# XII-STD

# COMPUTER SCIENCE

**NAME:** _____

**CLASS & SEC:** _____

**SCHOOL:** _____

# INDEX

# 1. FUNCTION

## Part - I

**Choose the best answer**          **(1 Mark)**

1. The small sections of code that are used to perform a particular task is called

   **(A) Subroutines**    (B) Files    (C) Pseudo code    (D) Modules

2. Which of the following is a unit of code that is often defined within a greater code structure?

   (A) Subroutines    **(B) Function**    (C) Files    (D) Modules

3. Which of the following is a distinct syntactic block?

   (A) Subroutines    (B) Function    **(C) Definition**    (D) Modules

4. The variables in a function definition are called as

   (A) Subroutines    (B) Function    (C) Definition    **(D) Parameters**

5. The values which are passed to a function definition are called

   **(A) Arguments**    (B) Subroutines    (C) Function    (D) Definition

6. Which of the following are mandatory to write the type annotations in the function definition?

   (A) Curly braces    **(B) Parentheses**    (C) Square brackets    (D) indentations

7. Which of the following defines what an object can do?

   (A) Operating System    (B) Compiler    **(C) Interface**    (D) Interpreter

8. Which of the following carries out the instructions defined in the interface?

   (A) Operating System    (B) Compiler    **(C) Implementation**    (D) Interpreter

9. The functions which will give exact result when same arguments are passed are called

   (A) Impure functions        (B) Partial Functions

   (C) Dynamic Functions        **(D) Pure functions**

10. The functions which cause side effects to the arguments passed are called

    **(A) impure function**        (B) Partial Functions

    (C) Dynamic Functions        (D) Pure functions

## Part - II

**Answer the following questions**          **(2 Marks)**

1. **What is a subroutine?**
   ➢ Subroutines are small sections of code that are used to perform a particular task that can be used repeatedly.

2. **Define Function with respect to Programming language.**
   ➢ A function is a unit of code that is often defined within a greater code structure.
   ➢ A function works on many kinds of inputs and produces a concrete output.

3. **Write the inference you get from X:=(78).**
   ➢ X:=(78) is a function definition.

- Definitions bind values to names.
- Hence, the value 78 bound to the name 'X'.

**4. Differentiate interface and implementation.**

| Interface | Implementation |
|---|---|
| Interface just defines what an object can do, but won't actually do it | Implementation carries out the instructions defined in the interface |

**5. Which of the following is a normal function definition and which is recursive function definition**

> **i) let rec sum x y:**
>
> > **return x + y**
> >
> > > **Answer:** Recursive function
>
> **ii) let disp :**
>
> > **print 'welcome'**
> >
> > > **Answer:** Normal function
>
> **iii) let rec sum num:**
>
> > **if (num!=0) then return num + sum (num-1)**
> >
> > > **else**
> >
> > **return num**
> >
> > > **Answer:** Recursive function

## Part - III

**Answer the following questions**                                 **(3 Marks)**

**1. Mention the characteristics of Interface.**
- The class template specifies the interfaces to enable an object to be created and operated properly.
- An object's attributes and behaviour is controlled by sending functions to the object.

**2. Why strlen is called pure function?**
- strlen is a pure function because the function takes one variable as a parameter, and accesses it to find its length.
- This function reads external memory but does not change it, and the value returned derives from the external memory accessed.

**3. What is the side effect of impure function? Give example.**
- **Impure function has the following side effecte,**
  - Function impure(has side effect) is that it doesn't take any arguments and it doesn't return any value.
  - Function depends on variable or function outside of its definition block.
  - It never assure you that the function will behave the same every time it's called.
  - Example
    > **let y: = 0**
    > **(int) inc (int) x**
    > **y: = y + x;**
    > **return (y)**
  - In the above example the value of y get changed inside the function definition due to which the result will change each time.

- The side effect of the inc () function is it is changing the data of the external visible variable *'y'*.

## 4. Differentiate pure and impure function.

| Pure function | Impure function |
|---|---|
| ➢ Pure functions are functions which will give exact result when the same arguments are passed. | ➢ Impure functions never assure you that the function will behave the same every time it's called |
| ➢ Pure function does not cause any side effects to its output. | ➢ Impure function causes side effects to its output |
| ➢ The return value of the pure functions solely depends on its arguments passed. | ➢ The return value of the impure functions does not solely depend on its arguments passed. |
| ➢ They do not modify the arguments which are passed to them. | ➢ They may modify the arguments which are passed. |
| ➢ Example: strlen(), sqrt() | ➢ Example: random(), date() |

## 5. What happens if you modify a variable outside the function? Give an example.

➢ Modify a variable outside the function causes sid effect.

➢ **Example:**

> **let y: = 0**
> **(int) inc (int) x**
> **y: = y + x;**
> **return (y)**

- Here, the result of inc() will change every time if the value of 'y' get changed inside the function definition.
- Hence, the side effect of the inc () function is it is changing the data of the external visible variable *'y'*.

## Part - IV

**Answer the following questions**                          **(5Marks)**

### 1. What are called Parameters and write a note on
### (i) Parameter without Type (ii) Parameter with Type

➢ Parameters are the variables in a function definition.

➢ Arguments are the values which are passed to a function definition.

➢ Two types of parameter passing are,

> 1.       Parameter without type
> 2.       Parameter with type

### i.   Parameter without Type

➢ Let us see an example of a function definition of Parameter Without type:

> **(requires: b>=0 )**
> **(returns: a to the power of b)**
> **let rec pow a b:=**
> **if b=0 then 1**
> **else a * pow a (b-1)**

- In the above function definition variable *'b'* is the parameter and the value which is passed to the variable *'b'* is the argument.
- The precondition *(requires)* and postcondition *(returns)* of the function is given.
- We have not mentioned any types: *(data types)*. This is called parameter without type.
- In the above function definition the expression has type'int', so the function's return type also be 'int' by implicit.

## ii. Parameter without Type

- Now let us write the same function definition with types for some reason:

> **(requires: b> 0 )**
> **(returns: a to the power of b )**
> **let rec pow (a: int) (b: int) : int :=**
> **if b=0 then 1**
> **else a * pow b (a-1)**

- in this example we have explicitly annotating the types of arguments and return type as 'int'.
- Here, When we write the type annotations for *'a'* and *'b'* the parentheses are mandatory.
- This is the way passing parameter with type which helps the compiler to easily infer them.

## 2. Identify in the following program

> **let rec gcd a b :=**
> **if b <> 0 then gcd b (a mod b) else return a**

### i) Name of the function
- gcd

### ii) Identify the statement which tells it is a recursive function
- **let rec gcd a b :=**
- "rec"keyword tells the compiler it is a recursive function

### iii) Name of the argument variable
- 'a'and 'b'

### iv) Statement which invoke the function recursively
- gcd 'a'and 'b'

### v) Statement which terminates the recursion
- Return a

## 3. Explain with example Pure and impure functions.

| Pure function | Impure function |
|---|---|
| - Pure functions are functions which will give exact result when the same arguments are passed. | - Impure functions never assure you that the function will behave the same every time it's called |
| - Pure function does not cause any side effects to its output. | - Impure function causes side effects to its output |
| - The return value of the pure functions solely depends on its arguments passed. | - The return value of the impure functions does not solely depend on its arguments passed. |

| | |
|---|---|
| ➢ They do not modify the arguments which are passed to them. | ➢ They may modify the arguments which are passed. |
| ➢ If we call pure functions with same set of arguments, we will always get the same return values. | ➢ If we call impure functions with same set of arguments, we might get the different return values. |
| ➢ **Example:**<br>let square x<br>    return: x * x | ➢ **Example:**<br>let Random number<br>let a := random()<br>if a > 10 then<br>return: a<br>else<br>return: 10 |

4. **Explain with an example interface and implementation.**
   **Interface:**
➢ An interface is a set of action that an object can do.
➢ Interface just defines what an object can do, but won't actually do it
➢ The interface defines an object's visibility to the outside world.
➢ In Object Oriented Programming language, an Interface is a description of all functions that a class must have.
➢ The purpose of interfaces is to allow the computer to enforce the properties of the class of *TYPE T (whatever the interface is)* must have functions called *X, Y, Z,* etc.
➢ For example when you press a light switch, the light goes on, you may not have cared how it splashed the light.
➢ In our example, anything that *"ACTS LIKE"* a light, should have function definitions like turn_on () and a turn_off ().
➢ An object "ACTS LIKE"is an instance created from th class "LIGHT". All the objects of class "light"will uses all its functions.
   **Characteristics of Interface:**
➢ The class template specifies the interfaces to enable an object to be created and operated properly.
➢ An object's attributes and behaviour is controlled by sending functions to the object.
   **Implementation:**
➢ Implementation carries out the instructions defined in the interface.
➢ How the object is processed and executed is the implementation.
➢ A class declaration combines the external interface *(its local state)* with an implementation of that interface *(the code that carries out the behaviour).*
➢ **Example:**
   Let's take the example of increasing a car's speed.
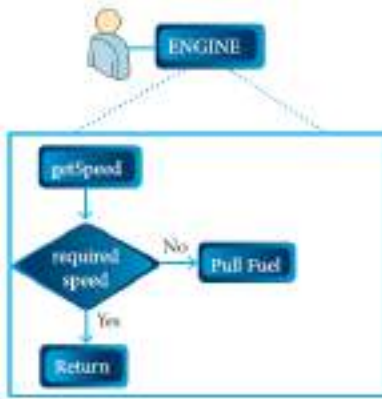
- ➢ The person who drives the car doesn't care about the internal working.
- ➢ To increase the speed of the car he just presses the accelerator to get the desired behaviour.
- ➢ Here the accelerator is the interface between the driver *(the calling / invoking object)* and the engine *(the called object)*.
- ➢ In this case, the function call would be Speed (70): This is the interface.
- ➢ Internally, the engine of the car is doing all the things.
- ➢ It's where fuel, air, pressure, and electricity come together to create the power to move the vehicle.
- ➢ All of these actions are separated from the driver, who just wants to go faster.
- ➢ Thus we separate interface from implementation.

# 2. DATA ABSTRACTION

## Part - I

**Choose the best answer**                                         **(1 Mark)**

1. Which of the following functions that build the abstract data type ?
   **(A) Constructors**    (B) Destructors    (C) recursive    (D)Nested
2. Which of the following functions that retrieve information from the data type?
   (A) Constructors    **(B) Selectors**      (C) recursive      (D)Nested
3. The data structure which is a mutable ordered sequence of elements is called
   (A) Built in       **(B) List**    (C) Tuple      (D) Derived data
4. A sequence of immutable objects is called
   (A) Built in     (B) List      **(C) Tuple**      (D) Derived data
5. The data type whose representation is known are called
   (A) Built in datatype           (B) Derived datatype
   **(C) Concrete datatype**         (D) Abstract datatype
6. The data type whose representation is unknown are called
   (A) Built in datatype           (B) Derived datatype
   (C) Concrete datatype        **(D) Abstract datatype**
7. Which of the following is a compound structure?
   **(A) Pair**     (B) Triplet        (C) single          (D) quadrat
8. Bundling two values together into one can be considered as
   **(A) Pair**     (B) Triplet          (C) single          (D) quadrat
9. 9. Which of the following allow to name the various parts of a multi-item object?
   (A) Tuples    (B) Lists       **(C) Classes**        (D) quadrats
10. 10. Which of the following is constructed by placing expressions within square brackets?
    (A) Tuples         **(B) Lists**     (C) Classes       (D) quadrats

## Part - II

**Answer the following questions**                                **(2 Marks)**

1. **What is abstract data type?**
   ➢ Abstract Data type (ADT) is a type (or class) for objects whose behavior is defined by a set of value and a set of operations.
2. **Differentiate constructors and selectors.**

| Constructors | Selectors |
|---|---|
| ▪ Constructors are functions that build the abstract data type. | ▪ Selectors are functions that retrieve information from the data type. |
| ▪ Constructors create an object, bundling together different pieces of information | ▪ selectors extract individual pieces of information from the object. |

3. **What is a Pair? Give an example.**
   ➢ Any way of bundling two values together into one can be considered as a pair.
   ➢ Lists are a common method to do so.
   ➢ Therefore List can be called as Pairs.
   ➢ Example: lst[(0, 10), (1, 20)]
4. **What is a List? Give an example.**

- ➢ List can store multiple values.
- ➢ List is constructed by placing expressions within square brackets separated by commas.
- ➢ Such an expression is called a list literal.
- ➢ Example: List is [10, 20]

**5. What is a Tuple? Give an example.**
- ➢ A tuple is a comma-separated sequence of values surrounded with parentheses.
- ➢ Tuple is similar to a list.
- ➢ Cannot change the elements of a tuple
- ➢ Example: colour= ('red', 'blue', 'Green')

## Part - III

**Answer the following questions** (3 Marks)

**1. Differentiate Concrete data type and abstract datatype.**

| Concrete Data Type | Abstract Data Type |
|---|---|
| ▪ Concrete data type or structures (CDT's) are direct implementation of a relatively simple concept. | ▪ Abstract data type (ADT's) offer a high level view (and use) of a concept independent of its implementation. |
| ▪ A concrete data type is a data type whose representation is known. | ▪ Abstract data type the representation of a data type is unknown. |

**2. Which strategy is used for program designing? Define that Strategy.**
- ➢ A powerful strategy for designing programs **'wishful thinking'**.
- ➢ Wishful Thinking is the formation of beliefs and making decisions according to what might be pleasing to imagine instead of by appealing to reality.

**3. Identify Which of the following are constructors and selectors?**
- (a) N1=number()                    - constructors
- (b) accetnum(n1)                   - selectors
- (c) displaynum(n1)                 - selectors
- (d) eval(a/b)                      - selectors
- (e) x,y= makeslope (m), makeslope(n)   - constructors
- (f) display()                      - selectors

**4. What are the different ways to access the elements of a list. Give example.**
- ➢ The elements of a list can be accessed in two ways.

i. **Multiple assignments**
- ➢ Which unpacks a list into its elements and binds each element to a different name.
- ➢ **Example:**

      lst := [10, 20]
      x, y := lst
- ➢ **x** will become10 and **y** will become 20.

ii. **Element selection operator:**
- ➢ It expressed using square brackets.
- ➢ Unlike a list literal, a square-brackets expression directly following another expression does not evaluate to a list value, but instead selects an element from the value of the preceding expression.
- ➢ **Example:**

      lst[0]
      10

8

lst[1]

20
5. **Identify Which of the following are List, Tuple and class?**
   (a) arr [1, 2, 34]                           - List
   (b) arr (1, 2, 34)                           - Tuple
   (c) student [rno, name, mark]                - Class
   (d) day= ('sun', 'mon', 'tue', 'wed')        - Tuple
   (e) x= [2, 5, 6.5, [5, 6], 8.2]              - List
   (f) employee [eno, ename, esal, eaddress]    - Class

## Part - IV

**Answer the following questions** (5Marks)

1. **How will you facilitate data abstraction. Explain it with suitable example**
➢ Data abstraction is supported by defining an abstract data type(ADT), which is a collection of constructors and selectors.
➢ To facilitate data abstraction, you will need to create two types of functions:
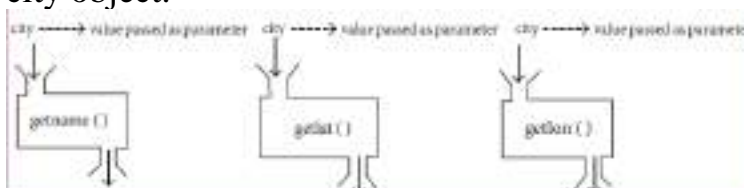  • Constructors
  • Selectors

**a) Constructors:**
➢ Constructors are functions that build the abstract data type.
➢ Constructors create an object, bundling together different pieces of information, while selectors extract individual pieces of information from the object.
➢ For example, say you have an abstract data type called city.
➢ This city object will hold the city's name, and its latitude and longitude.
➢ To create a city object, you'd use a function like **city = makecity (name, lat, lon).**
➢ Here makecity(name, lat, lon) is the constructor which creates the object city.
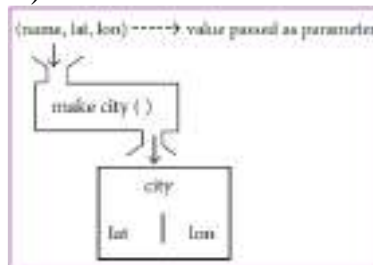


**b) Selectors:**
  ➢ Selectors are functions that retrieve information from the data type.
  ➢ Selectors extract individual pieces of information from the object.
  ➢ Selectors extract the information of a city object, you would use functions like
  • getname(city)
  • getlat(city)
  • getlon(city)
  ➢ These are the selectors because these functions extract the information of the city object.

2. **What is a List? Why List can be called as Pairs. Explain with suitable example**
   **LIST**
   - ➢ List is constructed by placing expressions within square brackets separated by commas.
   - ➢ Such an expression is called a list literal.
   - ➢ List can store multiple values.
   - ➢ Each value can be of any type and can even be another list.
   - ➢ The elements of a list can be accessed in two ways.

iii. **Multiple assignments**
   - ➢ Which unpacks a list into its elements and binds each element to a different name.
   - ➢ **Example:**

        lst := [10, 20]
        x, y := lst

   - ➢ **x** will become 10 and **y** will become 20.

iv. **Element selection operator:**
   - ➢ It expressed using square brackets.
   - ➢ Unlike a list literal, a square-brackets expression directly following another expression does not evaluate to a list value, but instead selects an element from the value of the preceding expression.
   - ➢ **Example:**

        lst[0]
        10
        lst[1]
        20

   **PAIR**
   - ➢ Any way of bundling two values together into one can be considered as a pair.
   - ➢ Lists are a common method to do so.
   - ➢ Therefore List can be called as Pairs.
   - ➢ Example: lst[(0, 10), (1, 20)]
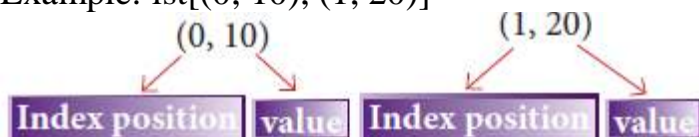


3. **How will you access the multi-item. Explain with example.**
   - ➢ The structure construct (In OOP languages it's called class construct) to represent multi-part objects where each part is named (given a name).
   - ➢ Consider the following pseudo code:

        class Person:
        creation( )
        firstName := " "
        lastName := " "
        id := " "
        email := " "

   - ➢ The new data type Person is pictorially represented as
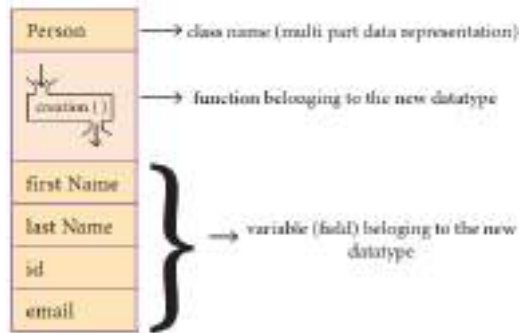
| Person | → class name (multi part data representation) |
| creation ( ) | → function belonging to the new datatype |
| first Name | |
| last Name | } → variable (field) belonging to the new datatype |
| id | |
| email | |

| Let main() contains | |
|---|---|
| p1:=Person() | statement creates the object. |
| firstName := " Padmashri " | setting a field called firstName with value Padmashri |
| lastName :="Baskar" | setting a field called lastName with value Baskar |
| id :="994-222-1234" | setting a field called id value 994-222-1234 |
| email="compsci@gmail.com" | setting a field called email with value compsci@gmail.com |
| - - output of firstName : Padmashri | |

➢ The class (structure) construct defines the form for multi-part objects that represent a person.

➢ Person is referred to as a class or a type, while p1 is referred to as an object or an instance.

➢ Using class you caqn create many objects.

➢ A class defines a data abstraction by grouping related data items.

➢ A class as bundled data and the functions that work on that data.

11

# 3. Scoping

## Part - I

**Choose the best answer** **(1 Mark)**

1. Which of the following refers to the visibility of variablesin one part of a program to another part of the same program.

    **(A) Scope**     (B) Memory     (C) Address     (D) Accessibility

2. The process of binding a variable name with an object is called

    (A) Scope     **(B) Mapping**     (C) late binding     (D) early binding

3. Which of the following is used in programming languages to map the variable and object?

    (A) ::     (B) :=     **(C) =**     (D) ==

4. Containers for mapping names of variables to objects is called

    (A) Scope     (B) Mapping     (C) Binding     **(D) Namespaces**

5. Which scope refers to variables defined in current function?

    **(A) Local Scope**     (B) Global scope

    (C) Module scope     (D) Function Scope

The process of subdividing a computer program into separate sub-programs is called

    (A) Procedural Programming     **(B) Modular programming**

    (C)Event Driven Programming     (D) Object oriented Programming

7. Which of the following security technique that regulates who canuse resources in a computing environment?

    (A) Password     (B)Authentication     **(C) Access control**     (D) Certification

8. Which of the following members of a class can be handled only from within the class?

    (A) Public members     (B)Protected members

    (C) Secured members     (D) Private members

9. Which members are accessible from outside the class?

    (A) Public members     (B)Protected members

    (C) Secured members     **(D) Private members**

10. The members that are accessible from within the class and are also available to its sub-classes is called

    (A) Public members     **(B)Protected members**

    (C) Secured members     (D) Private members

## Part - II

**Answer the following questions** **(2 Marks)**

1. **What is a scope?**
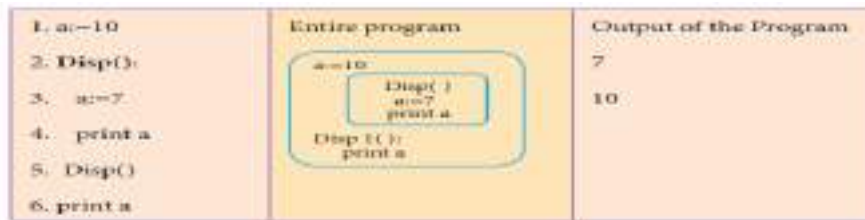
    Scope refers to the visibility of variables, parameters and functions in one part of a program to another part of the same program.

2. **Why scope should be used for variable. State the reason.**

    ➢ The scope should be used for variables because it limits a variable's scope to a single definition.

    ➢ That is the variables are visible only to that part of the code

    ➢ Example :

| 1. a:=10 | Entire program | Output of the Program |
|---|---|---|
| 2. Disp(): | a:=10 | 7 |
| 3. a:=7 | Disp( ) a:=7 print a | 10 |
| 4. print a | | |
| 5. Disp() | Disp 1( ): print a | |
| 6. print a | | |

## 3. What is Mapping?

➤ The process of binding a variable name with an object is called mapping.

➤ = (equal to sign) is used in programming languages to map the variable and object.

## 4. What do you mean by Namespaces?

➤ Namespaces are containers for mapping names of variables and object.

➤ Example: a:=5

➤ Here the variable 'a' is mapped to the value '5'.

## 5. How Python represents the private and protected Access specifiers?

➤ Python prescribes a convention of adding a prefix __(double underacore) results in a variable name or method becoming private.

➤ Example: self.__n2=n2

➤ Adding a prefix _ (single underscore) to a variable name or method makes it protected.

➤ Example: self._sal=sal

## Part - III

**Answer the following questions**             **(3 Marks)**

## 1. Define Local scope with an example.

**Local Scope**

➤ Local scope refers to variables defined in current function.

➤ Always, a function will first look up for a variable name in its local scope.

➤ Only if it does not find it there, the outer scopes are checked.

➤ Example:



| 1. Disp(): | Entire program | Output of the Program |
|---|---|---|
| 2. a:=7 | Disp( ): a:=7 print a | 7 |
| 3. print a | | |
| 4. Disp() | Disp ( ) | |

➤ On execution of the above code the variable *a* displays the value 7, because it is defined and available in the local scope.

## 2. Define Global scope with an example.

➤ A variable which is declared outside of all the functions in a program is known as global variable.

➤ Global variable can be accessed inside or outside of all the functions in a program.

➤ Example:

> On execution of the above code the variable *a* which is defined inside the function displays the value 7 for the function call Disp() and then it displays 10, because *a* is defined in global scope.

## 3. Define Enclosed scope with an example.

> A variable which is declared inside a function which contains another function definition with in it, the inner function can also access the variable of the outer function. This scope is called enclosed scope.

> When a compiler or interpreter search for a variable in a program, it first search Local, and then search Enclosing scopes.

> Example:



> In the above example Disp1() is defined with in Disp(). The variable 'a' defined in Disp() can be even used by Disp1() because it is also a member of Disp().

## 4. Why access control is required?

> Access control is a security technique that regulates who or what can view or use resources in a computing environment.

> It is a fundamental concept in security that minimizes risk to the object.

> In other words access control is a selective restriction of access to data.

> In Object oriented programming languages it is implemented through access modifiers.

## 5. Identify the scope of the variables in the following pseudo code and write its output

```
color:= Red
mycolor():
b:=Blue
myfavcolor():
g:=Green
printcolor, b, g
myfavcolor()
printcolor, b
mycolor()
print color
```

**OUTPUT:**

Red Blue Green

14

Red Blue
Red

**Scope of variables:**

| Variables | Scope |
|-----------|-------|
| Color.=Red | Global |
| b.=Blue | Enclosed |
| G.=Green | Local |

## Part - IV

**Answer the following questions**                                    **(5Marks)**

1.  **Explain the types of scopes for variable or LEGB rule with example.**

    **SCOPE**

    ➢ Scope refers to the visibility of variables, parameters and functions in one part of a program to another part of the same program.
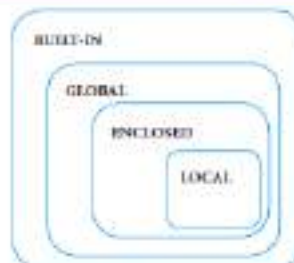
    **TYPES OF VARIABLE SCOPE:**

    ➢ Local Scope
    ➢ Enclosed Scope
    ➢ Global Scope
    ➢ Built-in Scope

    **LEGB RULE:**

    ➢ The *LEGB* rule is used to decide the order in which the scopes are to be searched for scope resolution.
    ➢ The scopes are listed below in terms of hierarchy (highest to lowest).

    | Local(L) | Defined inside function/class |
    |----------|-------------------------------|
    | Enclosed(E) | Defined inside enclosing functions (Nested function concept) |
    | Global(G) | Defined at the uppermost level |
    | Built-in (B) | Reserved names in built-in functions (modules) |

    

    **Local Scope**

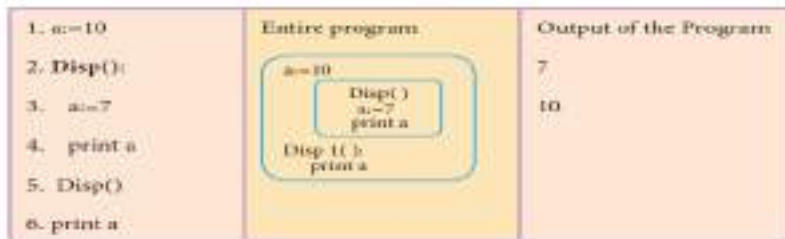    ➢ Local scope refers to variables defined in current function.
    ➢ Always, a function will first look up for a variable name in its local scope.
    ➢ Only if it does not find it there, the outer scopes are checked.
    ➢ Example:

    

    ➢ On execution of the above code the variable *a* displays the value 7, because it is defined and available in the local scope.

    **Global Scope**

15

- ➢ A variable which is declared outside of all the functions in a program is known as global variable.
- ➢ Global variable can be accessed inside or outside of all the functions in a program.
- ➢ Example:



| 1. a:=10 | Entire program | Output of the Program |
|---|---|---|
| 2. Disp(); | a=10 | 7 |
| 3. a:=7 | Disp()<br>a=7<br>print a | 10 |
| 4.  print a | Disp 1( );<br>print a | |
| 5. Disp() | | |
| 6. print a | | |

- ➢ On execution of the above code the variable *a* which is defined inside the function displays the value 7 for the function call Disp() and then it displays 10, because *a* is defined in global scope.
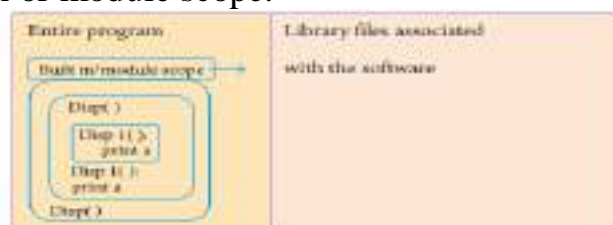
## Enclosed Scope
➢ A variable which is declared inside a function which contains another function definition with in it, the inner function can also access the variable of the outer function. This scope is called enclosed scope.

➢ When a compiler or interpreter search for a variable in a program, it first search Local, and then search Enclosing scopes.

➢ Example:



| 1. Disp(); | Entire program | Output of the Program |
|---|---|---|
| 2.  a:=10 | Disp( )<br>a=10<br>Disp 1( );<br>print a | 10 |
| 3.  Disp1() | | 10 |
| 4.   print a | Disp 1( )<br>print a | |
| 5.   Disp1() | Disp() | |
| 6. print a | | |
| 7. Disp() | | |

- ➢ In the above example Disp1() is defined with in Disp(). The variable 'a' defined in Disp() can be even used by Disp1() because it is also a member of Disp().

## Built-in Scope
➢ The built-in scope has all the names that are pre-loaded into the program scope when we start the compiler or interpreter.

➢ Any variable or module which is defined in the library functions of a programming language has Built-in or module scope.



| Entire program | Library files associated |
|---|---|
| Built in/module scope ⟶ | with the software |
| Disp( )<br>Disp 1( )<br>print a<br>Disp 1( )<br>print a<br>Disp() | |

## 2. Write any Five Characteristics of Modules.
The following are the desirable characteristics of a module.
1. Modules contain instructions, processing logic, and data.
2. Modules can be separately compiled and stored in a library.
3. Modules can be included in a program.
4. Module segments can be used by invoking a name and some parameters.

5. Module segments can be used by other modules.

## 3. **Write any five benefits in using modular programming.**

➢ Less code to be written.
➢ A single procedure can be developed for reuse, eliminating the need to retype the code many times.
➢ Programs can be designed more easily because a small team deals with only a small part of the entire code.
➢ Modular programming allows many programmers to collaborate on the same application.
➢ The code is stored across multiple files.
➢ Code is short, simple and easy to understand.
➢ Errors can easily be identified, as they are localized to a subroutine or function.
➢ The same code can be used in many applications.
➢ The scoping of variables can easily be controlled.

# 4. ALGORITHMIC STRATEGIES

## Part - I

**Choose the best answer:** (1 Marks)

1. The word comes from the name of a Persian mathematician Abu Ja'far Mohammed ibn-i Musa al Khowarizmi is called?
   (A) Flowchart    (B) Flow    **(C) Algorithm**    (D) Syntax

2. From the following sorting algorithms which algorithm needs the minimum number of swaps?
   (A) Bubble sort    (B) Quick sort    (C) Merge sort    **(D) Selection sort**

3. Two main measures for the efficiency of an algorithm are
   (A) Processor and memory    (B) Complexity and capacity
   **(C) Time and space**    (D) Data and space

4. The complexity of linear search algorithm is
   **(A) O(n)**    (B) O(log n)    (C) O(n2)    (D) O(n log n)

5. From the following sorting algorithms which has the lowest worst case complexity?
   (A) Bubble sort    (B) Quick sort    **(C) Merge sort**    (D) Selection sort

6. Which of the following is not a stable sorting algorithm?
   (A) Insertion sort    **(B) Selection sort**    (C) Bubble sort    (D) Merge sort

7. Time complexity of bubble sort in best case is
   **(A) θ (n)**    (B) θ (nlogn)    (C) θ (n2)    (D) θ (n(logn) 2)

8. The Θ notation in asymptotic evaluation represents
   (A) Base case    **(B) Average case**    (C) Worst case    (D) NULL case

9. If a problem can be broken into subproblems which are reused several times, the problem possesses which property?
   **(A) Overlapping subproblems**    (B) Optimal substructure
   (C) Memoization    (D) Greedy

10. In dynamic programming, the technique of storing the previously calculated values is called ?
    (A) Saving value property    (B) Storing value property
    **(C) Memoization**    (D) Mapping

## Part - II

**Answer the following questions:** (2 Marks)

1. **What is an Algorithm?**
   ➤ An algorithm is a finite set of instructions to accomplish a particular task.
   ➤ It is a step-by-step procedure for solving a given problem.

2. **Define Pseudo code.**
   ➤ Pseudo code is a methodology that allows the programmer to represent the implementation an algorithm.
   ➤ It has no syntax like programming languages and thus can't be compiled or interpreted by the computer.

3. **Who is an Algorist?**
   ➤ An Algorist is a person skilled in the design of algorithms.
   ➤ An algorithmic artist.

4. **What is Sorting?**
   ➤ Sorting is a process of arranging group of items in an ascending or descending order.

18

- Bubble Sort, Quick Sort, Heap Sort, Merge Sort, Selection Sort are the various sorting algorithms.

## 5. What is searching? Write its types.
- A search algorithm is the step-by-step procedure used to locate specific data among a collection of data.
- Example: Linear Search, Binary Search.

### Part - III
**Answer the following questions:**                                **(3 Marks)**

### 1. List the characteristics of an algorithm.
- Input
- Output
- Finiteness
- Definiteness
- Effectiveness
- Correctness
- Unambiguous
- Feasibility
- Portable
- Independent

### 2. Discuss about Algorithmic complexity and its types.
**Algorithmic complexity:**
- The complexity of an algorithm f (n) gives the running time and/or the storage space required by the algorithm in terms of n as the size of input data.

**Types of complexity:**

i. **Time Complexity**
- The Time complexity of an algorithm is given by the number of steps taken by the algorithm to complete the process.

ii. **Space Complexity**
- **Space complexity** of an algorithm is the amount of memory required to run to its completion.
- The space required by an algorithm is equal to the sum of the fixed part and variable part.

### 3. What are the factors that influence time and space complexity.
- The two main factors, which decide the efficiency of an algorithm are

i. **Time Factor**
- Time is measured by counting the number of key operations like comparisons in the sorting algorithm.

ii. **Space Factor**
- Space is measured by the maximum memory space required by the algorithm.

### 4. Write a note on Asymptotic notation.
**Asymptotic Notations**

- Asymptotic Notations are languages that uses meaningful statements about time and space complexity.
- The following three asymptotic notations are mostly used to represent time complexity of algorithms:

### i. Big O:
- Big O is often used to describe the worst-case of an algorithm.

### ii. Big Ω
- Big Omega is the reverse Big O, if Bi O is used to describe the upper bound (worst - case) of a asymptotic function, Big Omega is used to describe the lower bound (best-case).

### iii. Big Θ
- When an algorithm has a complexity with lower bound = upper bound, say that an algorithm has a complexity O (n log n) and Ω (n log n), it's actually has the complexity Θ (n log n),
- Time complexity is n log n in both best-case and worst-case.

## 5. What do you understand by Dynamic programming?
➢ Dynamic programming is used when the solution to a problem can be viewed as the result of a sequence of decisions.
➢ Dynamic programming approach is similar to divide and conquer (i.e.) the problem can be divided into smaller sub-problems.
➢ Results of the sub-problems can be re-used to complete the process.
➢ Dynamic programming approaches are used to find the solution in optimized way.

## Part - IV

**Answer the following questions:**                              **(5Marks)**

### 1. Explain the characteristics of an algorithm.

| | |
|---|---|
| **Input** | Zero or more quantities to be supplied. |
| **Output** | At least one quantity is produced. |
| **Finiteness** | Algorithms must terminate after finite number of steps. |
| **Definiteness** | All operations should be well defined. For example operations involving division by zero or taking square root for negative number are unacceptable. |
| **Effectiveness** | Every instruction must be carried out effectively. |
| **Correctness** | The algorithms should be error free. |
| **Simplicity** | Easy to implement. |
| **Unambiguous** | Algorithm should be clear and unambiguous. Each of its steps and their inputs/outputs should be clear and must lead to only one meaning. |
| **Feasibility** | Should be feasible with the available resources. |
| **Portable** | An algorithm should be generic, independent of any programming language or an operating system able to handle all range of inputs. |
| **Independent** | An algorithm should have step-by-step directions, which should be independent of any programming code. |

### 2. Discuss about Linear search algorithm.
**LINEAR SEACH:**
➢ Linear search also called sequential search is a sequential method for finding a particular value in a list.

- This method checks the search element with each element in sequence until the desired element is found or the list is exhausted.
- In this searching algorithm, list need not be ordered.

  **Pseudo code:**

1. Traverse the array using for loop
2. In every iteration, compare the target search key value with the current value of the list.
   - If the values match, display the current index and value of the array
   - If the values do not match, move on to the next array element.
3. If no match is found, display the search element not found.

   **EXAMPLE:**

- To search the number 25 in the array given below, linear search will go step by step in a sequential order starting from the first element in the given array if the search element is found that index is returned otherwise the search is continued till the last index of the array.
- In this example number 25 is found at index number 3.

| index | 0 | 1 | 2 | 3 | 4 |
|-------|-----|-----|-----|-----|-----|
| values | 10 | 12 | 20 | 25 | 30 |

   Input: values[] = {5, 34, 65, 12, 77, 35}

   Target = 25

   **Output:**

   3

**3. What is Binary search? Discuss with example.**

   **Binary Search**

- Binary search also called half-interval search algorithm. It finds the position of a search element within a sorted array.
- The binary search algorithm can be done as divide-and-conquer search algorithm and executes in logarithmic time.

**Pseudo code for Binary search**

i. Start with the middle element:

- If the search element is equal to the middle element of the array i.e., the middle value = number of elements in array/2, then return the index of the middle element.
- If not, then compare the middle element with the search value,
- If the search element is greater than the number in the middle index, then select the elements to the right side of the middle index, and go to Step-1.
- If the search element is less than the number in the middle index, then select the elements to the left side of the middle index, and start with Step-1.

ii. When a match is found, display success message with the index of the element matched.

iii. If no match is found for all comparisons, then display unsuccessful message.

**Binary Search Working principles**

- List of elements in an array must be sorted first for Binary search.
- The array is being sorted so it enables to do the binary search algorithm.

21

➤ Let us assume that the search element is 60 and we need to search the location or index of search element 60 using binary search.
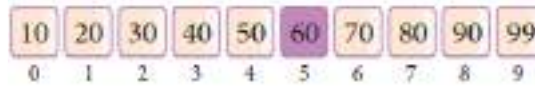


- First, we find index of middle element of the array by using this formula :
    **mid = low + (high - low) / 2**
- Here it is, 0 + (9 - 0 ) / 2 = 4 (fractional part ignored). So, 4 is the mid value of tharray.



- Compare the search element with the value stored at mid value location 4. The value stored at location or index 4 is 50, which is not match with search element. As the search value 60 is greater than 50.



- Now we change our low to mid + 1 and find the new mid value again using the formula.
- We compare the value stored at location 7 with our target value 31.



- Element not found because the value in index 7 is greater than search value(80>60)
- So, the search element must be in the lower part from the current mid value location



- Now we change our search range low to mid-1 and find the new mid value as index 5.
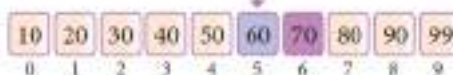


- Now we compare the value stored at location 5with our search element.
- We found that it is a match.



- We can conclude that the search element 60 is found at location or index 5.

**4. Explain the Bubble sort algorithm with example.**
➤ Bubble sort is a simple sorting algorithm; it starts at the beginning of the list of values stored in an array.
➤ It compares each pair of adjacent elements and swaps them if they are in the unsorted order.
➤ This comparison and passed to be continued until no swaps are needed, which indicates that the list of values stored in an array is sorted.
➤ It is named so because, the way smaller elements "bubble" to the top of the list.
➤ It is too slow and less efficient when compared to insertion sort and other sorting methods.

**Pseudo code**
1. Start with the first element i.e., index = 0, compare the current element with the next element of the array.
2. If the current element is greater than the next element of the array, swap them.

3. If the current element is less than the next or right side of the element, move to the next element.
4. Go to Step 1 and repeat until end of the index is reached.

**Example:**
➢ Let's consider an array with values {15, 11, 16, 12, 14, 13}
➢ Below, we have a pictorial representation of how bubble sort will sort the given array.
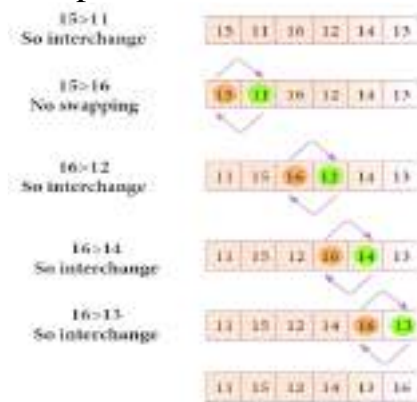


➢ The above pictorial example is for iteration-1.
➢ Similarly, remaining iteration can be done.
➢ The final iteration will give the sorted array.
➢ At the end of all the iterations we will get the sorted values in an array as given below:

| 11 | 12 | 13 | 14 | 15 | 16 |
|----|----|----|----|----|----|

**5. Explain the concept of Dynamic programming with suitable example.**
➢ Dynamic programming is used when the solution to a problem can be viewed as the result of a sequence of decisions.
➢ Dynamic programming approach is similar to divide and conquer (i.e.) the problem can be divided into smaller sub-problems.
➢ Results of the sub-problems can be re-used to complete the process.
➢ Dynamic programming approaches are used to find the solution in optimized way.

**Steps to do Dynamic programming**
➢ The given problem will be divided into smaller overlapping sub-problems.
➢ An optimum solution for the given problem can be achieved by using result of smaller sub-problem.
➢ Dynamic algorithms uses Memoization.

**Fibonacci Iterative Algorithm with Dynamic programming approach**
➢ The following example shows a simple Dynamic programming approach for the generation of Fibonacci series.
➢ Initialize f0=0, f1 =1
➢ step-1: Print the initial values of Fibonacci f0 and f1
➢ step-2: Calculate fibanocci fib ← f0 + f1
➢ step-3: Assign f0← f1, f1← fib
➢ step-4: Print the next consecutive value of fibanocci fib
➢ step-5: Goto step-2 and repeat until the specified number of terms generated
➢ For example if we generate fibobnacci series upto 10 digits, the algorithm will generate the series as shown below:
➢ The Fibonacci series is : 0 1 1 2 3 5 8 13 21 34 55

# 5. PYTHON-VARIABLES AND OPERATORS

## Part- I

**Choose the best answer** **(1 Marks)**

1. Who developed Python ?
   A) Ritche    **B) Guido Van Rossum**    C) Bill Gates    D) Sunder Pitchai
2. The Python prompt indicates that Interpreter is ready to accept instruction.
   **A) >>>**          B) <<<          C) #          D) <<
3. Which of the following shortcut is used to create new Python Program?
   A) Ctrl + C        B) Ctrl + F        C) Ctrl + B        **D) Ctrl + N**
4. Which of the following character is used to give comments in Python Program?
   **A) #**        B) &        C)        @ D) $
5. This symbol is used to print more than one item on a single line.
   A) Semicolon(;)      B) Dollor($)      **C) comma(,)**          D) Colon(:)
6. Which of the following is not a token?
   **A) Interpreter**      B) Identifiers      C) Keyword      D) Operators
7. Which of the following is not a Keyword in Python?
   A) break            B) while            C) continue      **D) operators**
8. Which operator is also called as Comparative operator?
   A) Arithmetic      **B) Relational**    C) Logical      **D) Assignment**
9. Which of the following is not Logical operator?
   A) and            B) or        C) not      **D) Assignment**
10. Which operator is also called as Conditional operator?
   **A) Ternary**      B) Relational      C) Logical    D) Assignment

## Part - II

**Answer the following questions:** **(2 Marks)**

1. **What are the different modes that can be used to test Python Program?**
   ➢ In Python, programs can be written in two ways namely **Interactive mode** and **Script mode.**
   ➢ The Interactive mode allows us to write codes in Python command prompt (**>>>**)
   ➢ Script mode is used to create and edit python source file.
2. **Write short notes on Tokens.**
   ➢ Python breaks each logical line into a sequence of elementary lexical components known as **Tokens**. The normal token types are
      1. Identifiers,
      2. Keywords,
      3. Operators,
      4. Delimiters and
      5. Literals.
3. **What are the different operators that can be used in Python?**
   ➢ Operators are special symbols which represent computations, conditional matching etc.
   ➢ Operators are categorized as Arithmetic, Relational, Logical, Assignment etc.
4. **What is a literal? Explain the types of literals?**
   ➢ Literal is a raw data given in a variable or constant.
   ➢ In Python, there are various types of literals.

24

- Numeric Literals consists of digits and are immutable.
- A string literal is a sequence of characters surrounded by quotes.
- A Boolean literal can have any of the two values: True or False.

### 5. Write short notes on Exponent data?
- An Exponent data contains decimal digit part, decimal point, exponent part followed by one or more digits.
- **Example :** 12.E04, 24.e04

## Part - III

**Answer the following questions:**                              **(3 Marks)**

### 1. Write short notes on Arithmetic operator with examples.
- An arithmetic operator is a mathematical operator used for simple arithmetic.
- It takes two operands and performs a calculation on them.
- Arithmetic operators used in Python:

| Operator - Operation | Examples | Result |
|---|---|---|
| Assume a=100 and b=10. Evaluate the following expressions | | |
| + (Addition) | >>> a + b | 110 |
| - (Subtraction) | >>>a – b | 90 |
| * (Multiplication) | >>> a*b | 1000 |
| / (Division) | >>> a / b | 10.0 |
| % (Modulus) | >>> a % 30 | 10 |
| ** (Exponent) | >>> a ** 2 | 10000 |
| // (Floor Division) | >>> a//30 (Integer Division) | 3 |

### 2. What are the assignment operators that can be used in Python?
- = is a simple assignment operator to assign values to variable.
- There are various compound operators in Python like +=, -=, *=, /=, %=, **= and //= are also available.
- **Example:**

      **a=5**              #assigns the value 5 to a
      **a,b=5, 10**       #assigns the value 5 to a and 10 to b
      **a+=2**            #a=a+2, add to the value of 'a' and stores the result in 'a' (left hand operator)

### 3. Explain Ternary operator with examples.
- Ternary operator is also known as conditional operator that evaluates something based on a condition being true or false.
- It simply allows testing a condition in a single line replacing the multiline if-else making the code compact.
- The Syntax conditional operator is,

      **Variable Name = [on_true] if [Test expression] else [on_false]**

- **Example :**

      min= 50 if 49<50 else 70 # min = 50

### 4. Write short notes on Escape sequences with examples.
- In Python strings, the backslash **"\"** is a special character, also called the **"escape"** character.
- It is used in representing certain whitespace characters
- Python supports the following escape sequence characters.

| Escape sequence character | Description | Example | Output |
|---|---|---|---|
| \\ | Backslash | >>> print("\\test") | \test |
| \' | Single-quote | >>> print("Doesn\'t") | Doesn't |
| \" | Double-quote | >>> print("\"Python\"") | "Python" |
| \n | New line | print("Python","\n","Lang..") | Python Lang.. |
| \t | Tab | print("Python","\t","Lang..") | Python Lang.. |

5. **What are string literals? Explain.**

➢ In Python a string literal is a sequence of characters surrounded by quotes.
➢ Python supports single, double and triple quotes for a string.
➢ A character literal is a single character surrounded by single or double quotes.
➢ The value with triple-quote "' '" is used to give multi-line string literal.
➢ **Example:**
>     strings = "This is Python"
>     char = "C"
>     multiline_str = "'This is a multiline string with more than one line code."'
>     print (strings)
>     print (char)
>     print (multiline_str)

**Output:**
>     This is Python
>     C
>     This is a multiline string with more than one line code.

**Part - IV**

**Answer the following questions:** (5 Marks)

1. **Describe in detail the procedure Script mode programming.**

**Script mode Programming**

➢ A script is a text file containing the Python statements.
➢ Python Scripts are reusable code.
➢ Once the script is created, it can be executed again and again without retyping.
➢ The Scripts are editable.

**(i) Creating Scripts in Python**

1. Choose **File → New File** or press **Ctrl + N** in Python shell window.
2. An untitled blank script text editor will be displayed on screen
3. Type the code in Script editor as given below,
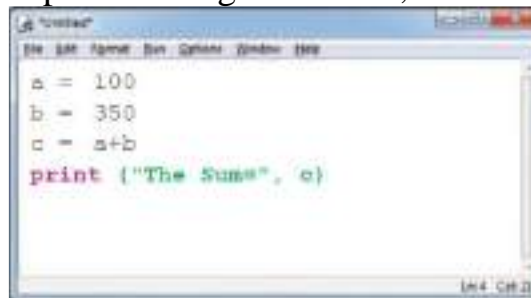


**(ii) Saving Python Script**

1. Choose **File → Save** or Press **Ctrl + S**
2. Now, **Save As** dialog box appears on the screen.

3. In the **Save As** dialog box,
   - select the location where you want to save your Python code, and
   - Type the file name in **File Name** box.
   - Python files are by default saved with extension **.py.**
   - So, while creating Python scripts using Python Script editor, no need to specify the file extension.
4. Finally, click **Save** button to save your Python script.

**(iii) Executing Python Script**

1. Choose **Run → Run Module** or Press **F5**
2. If your code has any error, it will be shown in red color in the IDLE window, and Python describes the type of error occurred. To correct the errors, go back to Script editor, make corrections, save the file using **Ctrl + S** or **File → Save** and execute it again.
3. For all error free code, the output will appear in the IDLE window of Python as shown in **Figure.**



## 2. Explain input() and print() functions with examples.

**Input and Output Functions**

- ➢ A program needs to interact with the user to accomplish the desired task; this can be achieved using **Input-Output functions**.
- ➢ The **input()** function helps to enter data at run time by the user
- ➢ The output function **print ()** is used to display the result of the program on the screen after execution.

**The print() function**

- ➢ In Python, the **print()** function is used to display result on the screen.
- ➢ The syntax for **print()** is as follows:

    print ("string to be displayed as output " )

    print (variable )

    print ("String to be displayed as output ", variable)

    print ("String1 ", variable, "String 2", variable, "String 3" ……)

- ➢ **Example**

    >>> print ("Welcome to Python Programming")
        Welcome to Python Programming
    >>> x = 5
    >>> y = 6
    >>> z = x + y
    >>> print (z)
        11
    >>> print ("The sum = ", z)

27

$$\text{The sum} = 11$$

>>> print ("The sum of ", x, " and ", y, " is ", z)
The sum of 5 and 6 is 11

- ➢ The **print ( )** evaluates the expression before printing it on the monitor.
- ➢ The print () displays an entire statement which is specified within print ( ).
- ➢ **Comma ( , )** is used as a separator in **print ( )** to print more than one item.

## input() function

- ➢ In Python, **input( )** function is used to accept data as input at run time.
- ➢  The syntax for **input()** function is,

**Variable = input ("prompt string")**

- ➢ **Prompt string** in the syntax is a statement or message to the user, to know what input can be given.
- ➢ If a prompt string is used, it is displayed on the monitor; the user can provide expected data from the input device.
- ➢ The **input( )** takes whatever is typed from the keyboard and stores the entered data in the given variable.
- ➢ If prompt string is not given in **input( )** no message is displayed on the screen, thus, the user will not know what is to be typed as input.
- ➢ **Example 1:input( ) with prompt string**

>>> city=input ("Enter Your City: ")
Enter Your City: Madurai

- ➢ **Example 2:input( ) without prompt string**

>>> city=input()

Rajarajan

## Input() using numerical values:

- ➢ The **input ( )** accepts all data as string or characters but not as numbers.
- ➢ The **int( )** function is used to convert string data as integer data explicitly.
- ➢ **Example**

```
x = int (input("Enter Number 1: "))
y = int (input("Enter Number 2: "))
print ("The sum = ", x+y)
```

**Output:**
Enter Number 1: 34
Enter Number 2: 56
The sum = 90

## 3.  Discuss in detail about Tokens in Python.

## TOKENS

- ➢ Python breaks each logical line into a sequence of elementary lexical components known as **Tokens**.
- ➢ The normal token types are
1) Identifiers,
2) Keywords,
3) Operators,
4) Delimiters and
5) Literals.

- ➢ Whitespace separation is necessary between tokens, identifiers or keywords.

## i. **Identifiers**
- ➢ An Identifier is a name used to identify a variable, function, class, module or object
- ➢ An identifier must start with an alphabet (A..Z or a..z) or underscore ( _ ).
- ➢ Identifiers may contain digits (0 .. 9)
- ➢ Python identifiers are case sensitive i.e. uppercase and lowercase letters are distinct.
- ➢ Identifiers must not be a **python** keyword.
- ➢ Python does not allow punctuation character such as %,$, @ etc., within identifiers.
- ➢ **Example of valid identifiers**
    - Sum, total_marks, regno, num1
- ➢ **Example of invalid identifiers**
    - 12Name, name$, total-mark, continue

## ii. **Keywords**
- ➢ Keywords are special words used by Python interpreter to recognize the structure of program.
- ➢ Keyword have specific meaning for interpreter, they cannot be used for any other purpose.
- ➢ **Python's Keywords:** false, class, if, elif, else, pass, break etc.

## iii. **Operators**
- ➢ In computer programming languages operators are special symbols which represent computations, conditional matching etc.
- ➢ Operators are categorized as Arithmetic, Relational, Logical, Assignment etc.
- ➢ Value and variables when used with operator are known as **operands**.

## iv. **Delimiters**
- ➢ Python uses the symbols and symbol combinations as delimiters in expressions, lists, dictionaries and strings.
- ➢ Following are the delimiters.

| ( | ) | [ | ] | { | } |
|---|---|---|---|---|---|
| , | : | . | ` | = | ; |
| += | -= | *= | /= | //= | %= |
| &= | |= | ^= | >>= | <<= | **= |

## v. **Literals**
- ➢ Literal is a raw data given in a variable or constant.
- ➢ In Python, there are various types of literals.
    - ▪ Numeric Literals consists of digits and are immutable.
    - ▪ A string literal is a sequence of characters surrounded by quotes.
    - ▪ A Boolean literal can have any of the two values: True or False

# 6. Control Structure
## Part - I

**Choose the best answer**                                                      **1 Marks**

1. How many important control structures are there in Python?
   **A) 3**              B) 4              C) 5              D) 6
2. elif can be considered to be abbreviation of
   A) nested if        B) if..else        **C) else if**        D) if..elif
3. What plays a vital role in Python programming?
   A) Statements      B) Control        C) Structure        **D) Indentation**
4. Which statement is generally used as a placeholder?
   A) Continue        B) break          **C) pass**          D) goto
5. The condition in the if statement should be in the form of
   A) Arithmetic or Relational expression
   B) Arithmetic or Logical expression
   **C) Relational or Logical expression**
   D) Arithmetic
6. Which is the most comfortable loop?
   A) do..while            B) while            **C) for**            D) if..elif
7. What is the output of the following snippet?
   ```
   i=1
   while True:
   if i%3 ==0:
   break
   print(i,end=")
   i +=1
   ```
   **A) 12**              B) 123              C) 1234              D) 124
8. What is the output of the following snippet?
   ```
   T=1
   while T:
   print(True)
   break
   ```
   A) False            **B) True**            C) 0            D) no output
9. Which amongst this is not a jump statement?
   **A) for**          B) goto      C) continue              D) break
10. Which punctuation should be used in the blank?
**if <condition>_**
**statements-block 1**
**else:**
**statements-block 2**
   A) ;            **B) :**          C) ::          D) !

## Part -II

**Answer the following questions**                                              **(2 Marks)**

**1. List the control structures in Python.**
   There are three important control structure.

2. **Write note on break statement.**
   - The **break** statement terminates the loop containing it.
   - Control of the program flows to the statement immediately after the body of the loop.
3. **Write the syntax of if..else statement**
   **Syntax:**

   if<condition>:
   Statements-block 1
   else:
   statements-block 2

4. **Define control structure.**
   A program statement that causes a jump of control from one part of the program to another is called **control structure** or **control statement.**
5. **Write note on range () in loop**
   - range() generates a list of values starting from **start** till **stop-1.**
   - The syntax of range() is as follows:

     range (start,stop,[step])

   Where,

       **start** – refers to the initial value

       **stop** – refers to the final value

       **step** – refers to increment value, this is optional part.

**Part –III**

**Answer the following questions**                            **(3 Marks)**

1. **Write a program to display**
   **A**
   **A B**
   **A B C**
   **A B C D**
   **A B C D E**
   **Code:**

   for i in range(65,70):
       for j in range(65,i+1):
           print(chr[j],end=' ')
       else:
           print("")

2. **Write note on if..else structure.**
   - The **if .. else** statement provides control to check the true block as well as the false block.
   - **if..else** statement thus provides two possibilities and the condition determines which BLOCK is to be executed.

31

**Syntax:**
if<condition>:
statements-block 1
else:
statements-block 2

3. **Using if..else..elif statement write a suitable program to display largest of 3 numbers.**
   **Code:**
   n1=int(input("Enter the first number:"))
   n2=int(input("Enter the second number:"))
   n3=int(input("Enter the third number:"))
   if(n1>=n2)and(n1>=n3):
           biggest=n1;
   elif(n2>=n1)and(n2>=n3):
           biggest=n2
   else:
           biggest=n3
   print("The biggest number between", "n1", "n2", "and", "n3", "is", biggest)
   **OUTPUT**
   Enter the first number: 1
   Enter the second number: 3
   Enter the third number: 5
   The biggest number between 1, 3 and 5 is 5

4. **Write the syntax of while loop.**
   **Syntax:**
           while<condition>:
                   statements block 1
           [else:
                   statements block 2]

5. **List the differences between break and continue statements.**

| break | continue |
|---|---|
| The **break** statement terminates the loop containing it. | Continue statement unlike the break statement is used to skip the remaining part of a loop |
| Control of the program flows to the statement immediately after the body of the loop. | Control of the program start with next iteration. |
| **Syntax:** <br> break | **Syntax:** <br> continue |

**Part -IV**

**Answer the following questions**                             **(5 Marks)**
   1. **Write a detail note on for loop**
   ➢ **for** loop is the most comfortable loop. It is also an entry check loop.
   ➢ The condition is checked in the beginning and the body of the loop(statements-block 1) is executed if it is only True otherwise the loop is not executed.

**Syntax:**

```
for counter_variable in sequence:
    statements-block 1
[else:
    statements-block 2]
```

➤ The *counter_variable* is the control variable.

➤ the *sequence* refers to the initial, final and increment value.

➤ **for** loop uses the *range()* function in the sequence to specify the initial, final and increment values.

➤ *range()* generates a list of values starting from **start** till **stop-1.**

    **The syntax of range() is as follows:**

    range (start,stop,[step])

    Where,

        start – refers to the initial value

        stop – refers to the final value

        step – refers to increment value, this is optional part.

    **Example:**

```
        for i in range (2,10,2):
                print (i, end=' ')

        else:
                print("\n End of the loop")
```

    **Output:**

    2 4 6 8

    End of the loop

**2. Write a detail note on if..else..elif statement with suitable example.**

➤ When we need to construct a chain of **if** statement(s) then **'elif'** clause can be used instead of **'else'.**

➤ **'elif'** clause combines **if..else-if..else** statements to one **if..elif…else.**

➤ **'elif'** can be considered to be abbreviation of **'else if'.**

➤ In an **'if'** statement there is no limit of **'elif'** clause that can be used, but an **'else'** clause if used should be placed at the end.

**Syntax:**

```
if <condition-1>:
    statements-block 1
elif <condition-2>:
    statements-block 2
else:
    statements-block n
```

➤ In the syntax of **if..elif..else** mentioned above, condition-1 is tested if it is true then statements-block1 is executed.

➤ otherwise the control checks condition-2, if it is true statements-block2 is executed and even if it fails statements-block n mentioned in **else** part is executed.

    **Example:**

```
m1=int (input("Enter mark in first subject : "))
m2=int (input("Enter mark in second subject : "))
```

```
avg= (m1+m2)/2
if avg>=80:
print ("Grade : A")
elif avg>=70 and avg<80:
print ("Grade : B")
elif avg>=60 and avg<70:
print ("Grade : C")
elif avg>=50 and avg<60:
print ("Grade : D")
else:
print ("Grade : E")
```

**Output 1:**

> Enter mark in first subject : 34
> Enter mark in second subject : 78
> Grade : D

3. **Write a program to display all 3 digit odd numbers.**
   **Code:**
```
lower=int(input("Enter the lower limit for the range:"))
upper=int(input("Enter the upper limit for the range:"))
for i in range(lower,upper+1):
        if(i%2!=0):
                print(i, end="")
```
   **Output:**

> Enter the lower limit for the range: 100
> Enter the upper limit for the range: 150
> 101 103 105 107 109 11 113 115 117 119 121 123 125 127 129 131 133 135
> 137 139 141 143 145 147 149

4. **Write a program to display multiplication table for a given number.**
   **Code:**
```
num=int(input("Display Multiplication Table of"))
for i in range(1,11):
        print(i, 'x', num, '=', num*i)
```
   **Output:**

Display Multiplication Table of 2
1 x 2 = 2
2 x 2 = 4
3 x 2 = 6
4 x 2 = 8
5 x 2 = 10
6 x 2 = 12
7 x 2 = 14
8 x 2 = 16
9 x 2 = 18
10 x 2 = 20

# 7. PYTHON FUNCTIONS

## Part - I

**Choose the best answer:**                                                    **(1 Mark)**

1. A named blocks of code that are designed to do one specific job is called as
   (a) Loop       (b) Branching     **(c) Function**     (d) Block
2. A Function which calls itself is called as
   (a) Built-in    **(b) Recursion**     (c) Lambda      (d) return
3. Which function is called anonymous un-named function
   **(a) Lambda**          (b) Recursion     (c) Function     (d) define
4. Which of the following keyword is used to begin the function block?
   (a) define         (b) for     (c) finally       **(d) def**
5. Which of the following keyword is used to exit a function block?
   (a) define         **(b) return**    (c) finally       (d) def
6. While defining a function which of the following symbol is used.
   (a) ; (semicolon)       (b) . (dot)        **(c) : (colon)**      (d) $ (dollar)
7. In which arguments the correct positional order is passed to a function?
   **(a) Required**     (b) Keyword       (c) Default       (d) Variable-length
8. Read the following statement and choose the correct statement(s).
   (I) In Python, you don't have to mention the specific data types while defining function.
   (II) Python keywords can be used as function name.
   > **(a) I is correct and II is wrong**
   > (b) Both are correct
   > (c) I is wrong and II is correct
   > (d) Both are wrong
9. Pick the correct one to execute the given statement successfully.
          if _____ : print(x, " is a leap year")
   (a) x%2=0       **(b) x%4==0**          (c) x/4=0     (d) x%4=0
10. Which of the following keyword is used to define the function testpython(): ?
    (a) define         (b) pass        **(c) def**       (d) while

## Part - II

**Answer the following questions:**                                         **(2 Marks)**

1. **What is function?**
   ➢ Functions are named blocks of code that are designed to do specific job.
   ➢ Types of function are User defined, Built-in, Lambda and recursion.
   ➢ Function blocks begin with the keyword **"def"** followed by function name and parenthesis ().
2. **Write the different types of function.**
   **Types of Functions**



3. **What are the main advantages of function?**
   **Main advantages of functions are**

- It avoids repetition and makes high degree of code reusing.
- It provides better modularity for your application.

**4. What is meant by scope of variable? Mention its types.**
- Scope of variable refers to the part of the program, where it is accessible, i.e., area where you can refer (use) it.
- Scope holds the current set of variables and their values. We will study
- The two types of scopes are- **local scope** and **global scope**.

**5. Define global scope.**
- A variable, with global scope can be used anywhere in the program.
- It can be created by defining a variable outside the scope of any function/block.

**6. What is base condition in recursive function**
- A recursive function calls itself.
- The condition that is applied in any recursive function is known as base condition.
- A base condition is must in every recursive function otherwise it will continue to execute like an infinite loop.

7. **How to set the limit for recursive function? Give an example.**
- Python stops calling recursive function after 1000 calls by default.
- So, it also allows you to change the limit using sys.setrecursionlimit (limit_value).
- **Example:**

```
import sys
sys.setrecursionlimit(3000)
def fact(n):
if n == 0:
return 1
else:
return n * fact(n-1)
print(fact (2000))
```

## Part - III

**Answer the following questions:**                                     **(3 Marks)**

**1. Write the rules of local variable.**
- A variable with local scope can be accessed only within the function/block that it is created in.
- When a variable is created inside the function/block, the variable becomes local to it.
- A local variable only exists while the function is executing.
- The formal arguments are also local to function.

**2. Write the basic rules for global keyword in python.**
- The basic rules for global keyword in Python are:
- When we define a variable outside a function, it's global by default. You don't have to use global keyword.
- We use global keyword to read and write a global variable inside a function.
- Use of global keyword outside a function has no effect

**3. What happens when we modify global variable inside the function?**

➢ If we modify the global variable, we can see the change on the global variable outside the function also.

➢ **EXAMPLE:**

```
x = 0 # global variable def add():
global x
x = x + 5 # increment by 2
print ("Inside add() function x value is :", x)
add()
print ("In main x value is :", x)
```

**Output:**

```
Inside add() function x value is : 5
In main x value is : 5
```

**4. Differentiate ceil() and floor() function?**

| Ceil() | Floor() |
|---|---|
| ➢ Returns the smallest integer greater than or equal to x | ➢ Returns the largest integer less than or equal to x |
| ➢ **math.ceil (x)** | ➢ **math.floor (x)** |

**5. Write a Python code to check whether a given year is leap year or not.**

**CODE:**

```
n=int(input("Enter the year"))
if(n%4==0)
    print("Leap Year")
else:
    print("Not a Leap Year")
```

**OUTPUT:**

```
Entre the year      2012
Leap Year
```

**6. What is composition in functions?**

➢ The value returned by a function may be used as an argument for another function in a nested manner.

➢ This is called composition.

➢ For example, if we wish to take a numeric value or an expression as a input from the user, we take the input string from the user using the function input() and apply eval() function to evaluate its value

**7. How recursive function works?**

➢ Recursive function is called by some external code.

➢ If the base condition is met then the program gives meaningful output and exits.

➢ Otherwise, function does some required processing and then calls itself to continue recursion.

**8. What are the points to be noted while defining a function?**

➢ Function blocks begin with the keyword **"def"** followed by function name and parenthesis ().

➢ Any input parameters or arguments should be placed within these parentheses when you define a function.

➢ The code block always comes after a colon (:) and is indented.

37

> The statement **"return [expression]"** exits a function, optionally passing back an expression to the caller. A **"return"** with no arguments is the same as return None.

<div align="center">

**Part - IV**

</div>

**Answer the following questions:**               **(5 Marks)**

1. **Explain the different types of function with an example.**

> Functions are named blocks of code that are designed to do specific job.
> **Types of function**
>> • User defined function
>> • Built-in function
>> • Lambda function
>> • Recursion function

**i. Built-in function:**
> Built-in function are functions that are inbuilt with in python
> print(), echo() are some built-function.

**ii. User defined function:**
> Function defined by the users themselves is called user defined function.
> Functions must be defined, to create and use certain functionality.
> Function blocks begin with the keyword **"def"** followed by function name and parenthesis ().
> When defining functions there are multiple things that need to be noted.
> Any input parameters or arguments should be placed within these parentheses.
> The code block always comes after a colon (:) and is indented.
> The statement **"return [expression]"** exits a function, and it is optional. **"return"** with no arguments is the same as return None.
> **Example:**

```
def area(w,h):
        return w * h
print (area (3,5))
```

**iii. Lambda function:**
> In Python, anonymous function is a function that is defined without a name.
> While normal functions are defined using the **def** keyword, in Python anonymous functions are defined using the **lambda** keyword.
> Hence, anonymous functions are also called as **lambda** functions.

**Use of lambda or anonymous function:**
> Lambda function is mostly used for creating small and one-time anonymous function.
> Lambda functions are mainly used in combination with the functions like filter(), map() and reduce().
> **Syntax:**

```
Lambda [arguments(s)] :expression
```

> **Example:**

```
sum = lambda arg1, arg2: arg1 + arg2
print ('The Sum is :', sum(30,40))
print ('The Sum is :', sum(-30,40))
```

> **Output:**

The Sum is : 70
The Sum is : 10

iv. **Recursive function:**
- ➢ Function that calls itself is known as recursive.

**Overview of how recursive function works**
- ➢ Recursive function is called by some external code.
- ➢ If the base condition is met then the program gives meaningful output and exits.
- ➢ Otherwise, function does some required processing and then calls itself to continue recursion.

2. **Explain the scope of variables with an example.**
- ➢ Scope of variable refers to the part of the program, where it is accessible, i.e., area where you can refer (use) it.
- ➢ Scope holds the current set of variables and their values.
- ➢ The two types of scopes - local scope and global scope.

**Local scope:**
- ➢ A variable declared inside the function's body or in the local scope is known as local variable.

**Rules of local variable**
- ➢ A variable with local scope can be accessed only within the function/block that it is created in.
- ➢ When a variable is created inside the function/block, the variable becomes local to it.
- ➢ A local variable only exists while the function is executing.
- ➢ The formal arguments are also local to function.
- ➢ **Example**

```
def loc():
        y=0 # local scope
print(y)
loc()
```

**Output:**
```
0
```

**Global scope:**
- ➢ A variable, with global scope can be used anywhere in the program.
- ➢ It can be created by defining a variable outside the scope of any function/block.

**Rules of global Keyword**
- ➢ The basic rules for global keyword in Python are:
- ➢ When we define a variable outside a function, it's global by default. You don't have to use global keyword.
- ➢ We use global keyword to read and write a global variable inside a function.
- ➢ Use of global keyword outside a function has no effect
- ➢ Use of global Keyword
- ➢ Without using the global keyword we cannot modify the global variable inside the function but we can only access the global variable.
- ➢ **EXAMPLE:**

```
x = 0 # global variable def add():
        global x
```

39

```
x = x + 5 # increment by 2
print ("Inside add() function x value is :", x)
add()
print ("In main x value is :", x)
```
**Output:**
```
Inside add() function x value is : 5
In main x value is : 5
```
**3. Explain the following built-in functions.**

**(a) id() (b) chr() (c) round() (d) type() (e) pow()**

| Function | Description | Syntax | Example |
|---|---|---|---|
| id ( ) | id( ) Return the "identity" of an object. i.e. the address of the object in memory. | id (object) | x=15<br>y='a'<br>print ('address of x is :',id (x))<br>print ('address of y is :',id (y))<br><br>**Output:**<br>address of x is : 1357486752<br>address of y is : 13480736 |
| chr ( ) | Returns the Unicode character for the given ASCII value. | chr (i) | c=65<br>print (chr (c))<br>**Output:**<br>A |
| round ( ) | Returns the nearest integer to its input.<br>1.      First argument (number) is used to specify the value to be rounded. | round (number [,ndigits]) | x= 17.9<br>print ('x value is rounded to', round (x))<br>**Output:**<br> X value is rounded to 18 |
| type ( ) | Returns the type of object for the given single object. | type (object) | x= 15.2<br>print (type (x))<br>**Output:**<br><class 'float'> |
| pow ( ) | Returns the computation of ab i.e. (a**b ) a raised to the power of b. | pow (a,b) | a= 5<br>print (pow (a,b))<br>**Output:**<br>25 |

**4. Write a Python code to find the L.C.M. of two numbers.**

   **CODE:**
```
x=int(input("Enter  first number:"))
y=int(input("Enter  second number:"))
if x>y:
```

```
                    min=x
            else
                    min=y
            while(1):
                    if((min%x==0)and(min%y==0)):
                            print("LCM is:",min)
                            break
                    min=min+1
```

**OUTPUT:**

Enter first number: 2
Enter second number: 3
LCM is: 6

## 5. Explain recursive function with an example.

- ➢ Function calls itself is known as recursion.
- ➢ Recursion works like loop but sometimes it makes more sense to use recursion than loop.
- ➢ Imagine a process would iterate indefinitely if not stopped by some condition! Such a process is known as infinite iteration.
- ➢ The condition that is applied in any recursive function is known as base condition.
- ➢ A base condition is must in every recursive function otherwise it will continue to execute like an infinite loop.
- ➢ Python stops calling recursive function after 1000 calls by default.
- ➢ So, it also allows you to change the limit using sys.setrecursionlimit(limit_value).

**Overview of how recursive function works**

1. Recursive function is called by some external code.
2. If the base condition is met then the program gives meaningful output and exits.
3. Otherwise, function does some required processing and then calls itself to continue recursion.

**EXAMPLE**

```
def fact(n):
        if n == 0:
                return 1
        else:
                return n * fact (n-1)
        print (fact (0))
        print (fact (5))
```

**Output:**
1
120

# CHAPTER 8
## STRINGS AND STRING MANIPULATION
### Part – I

**Choose the best answer**                                          **(1 Mark)**

1. Which of the following is the output of the following python code?

   ```
   str1="TamilNadu"
   print(str1[::-1])
   ```
   (a) Tamilnadu         (b) Tmlau    (c) udanlimaT    **d) udaNlimaT**

2. What will be the output of the following code?

   ```
   str1 = "Chennai Schools"
   str1[7] = "-"
   ```
   (a) Chennai-Schools     (b) Chenna-School     **(c) Type error**     (D) Chennai

3. Which of the following operator is used for concatenation?

   **(a) +**         (b) &         (c) *         d) =

4. Defining strings within triple quotes allows creating:

   (a) Single line Strings         **(b) Multiline Strings**
   (c) Double line Strings         (d) Multiple Strings

5. Strings in python:

   (a) Changeable     (b) Mutable     **(c) Immutable**     (d) flexible

6. Which of the following is the slicing operator?

   (a) { }         **(b) [ ]**         (c) < >     (d) ( )

7. What is stride?

   (a) index value of slide operation     (b) first argument of slice operation
   (c) second argument of slice operation     **(d) third argument of slice operation**

8. Which of the following formatting character is used to print exponential notation in upper case?

   (a) %e     **(b) %E**     (c) %g     (d) %n

9. Which of the following is used as placeholders or replacement fields which get replaced along with format( ) function?

   **(a) { }**         (b) < >     (c) ++         (d) ^^

10. The subscript of a string may be:

    (a) Positive         (b) Negative
    (c) Both (a) and (b)     **(d) Either (a) or (b)**

### Part –II

**Answer the following questions**                                   **(2 Marks)**

1. **What is String?**
   - String is a data type in python, which is used to handle array of characters.
   - String is a sequence of Unicode characters that may be a combination of letters, numbers, or special symbols enclosed within single, double or even triple quotes.

2. **Do you modify a string in Python?**
   - Yes we can modify the string by the following method
   - A new string value can be assign to the existing string variable.
   - When defining a new string value to the existing string variable.
   - Python completely overwrite new string on the existing string

3. **How will you delete a string in Python?**
   - ➢ Python will not allow deleting a particular character in a string.
   - ➢ Whereas you can remove entire string variable using **del** command.
   
   **Example:**
   del str1[2]

4. **What will be the output of the following python code?**
   
   **str1 = "School"**
   **print(str1*3)**
   
         **OUTPUT:**
   
                  School School School

5. **What is slicing?**
   - ➢ Slice is a substring of a main string.
   - ➢ A substring can be taken from the original string by using [ ] operator and index or subscript values.
   - ➢ Using slice operator, you have to slice one or more substrings from a main string.

**Part -III**

**Answer the following questions**              **(3 Marks)**

1. **Write a Python program to display the given pattern**
   
   C O M P U T E R
   C O M P U T E
   C O M P U T
   C O M P U
   C O M P
   C O M
   C O
   C

   **CODE**
   ```
   str1="COMPUTER"
   index=len(str1)
   for i in str1:
           print (str1[0:index])
        index-=1
   ```

2. **Write a short about the followings with suitable example:**
   (a) **capitalize( ) (b) swapcase( )**

| FUNCTION | PURPOSE | EXAMPLE |
|---|---|---|
| capitalize( ) | Used to capitalize the first character of the string | >>> city="chennai"<br>>>> print(city.capitalize())<br>**Output:**<br>**Chennai** |
| swapcase( ) | It will change case of every character to its opposite case vice-versa. | >>> str1="tAmiL NaDu"<br>>>> print(str1.swapcase())<br>**Output:**<br>*TaMIl nAdU* |

3. **What will be the output of the given python program?**
   str1 = "welcome"

str2 = "to school"
str3=str1[:2]+str2[len(str2)-2:]
print(str3)
**OUTPUT**

              **Weol**

4. **What is the use of format( )? Give an example.**
   ➢ The format( ) function used with strings is very versatile and powerful function used for formatting strings.
   ➢ The curly braces { } are used as placeholders or replacement fields which get replaced along with format( ) function.
   ➢ **Example**

num1=int (input("Number 1: "))
num2=int (input("Number 2: "))
print("The sum of { } and { } is { }".format(num1, num2,(num1+num2)))

  **Output**
       Number 1: 34
       Number 2: 54
       The sum of 34 and 54 is 88

5. **Write a note about count( ) function in python.**
   ➢ Returns the number of substrings occurs within the given range.
   ➢ Remember that substring may be a single character.
   ➢ Range (beg and end) arguments are optional.If it is not given, python searche
   ➢ Search is case sensitive.
   **SYNTAX:**
   Count(str,beg,end)
   **EXAMPLE:**
          >>>str1="Raja Raja Chozhan"
          >>>print(str1.count('Raja'))
   **OUTPUT:**
          2

## Part -IV

**Answer the following questions**                          **(5 Marks)**

1)Explain about string operators in python with suitable example.
   **STRING OPERATORS**
        Python provides the following string operators to manipulate string.

**(i) Concatenation (+)**
   ➢ Joining of two or more strings is called as Concatenation.
   ➢ The plus (+) operator is used to concatenate strings in python.
**Example:**
   >>> "welcome" + "Python"
**Output:**
   *'welcomePython'*
**(ii) Append (+ =)**
   ➢ Adding more strings at the end of an existing string is known as append.
   ➢ The operator += is used to append a new string with an existing string.
**Example**

44

```
>>> str1="Welcome to "
>>> str1+="Learn Python"
>>> print (str1)
```

**Output**

*Welcome to Learn Python*

**(iii) Repeating (*)**

The multiplication operator (*) is used to display a string in multiple number of times.

**Example**

```
>>> str1="Welcome "
>>> print (str1*4)
```

**Output**

Welcome Welcome Welcome Welcome

**(iv) String slicing**

> ➢ Slice is a substring of a main string.
> ➢ A substring can be taken from the original string by using [ ] operator and index or subscript values.
> ➢ Using slice operator, you have to slice one or more substrings from a main string.

**General format of slice operation:**

*str[start:end]*

➢ Where *start* is the beginning index and *end* is the last index value of a character in the string.

➢ Python takes the end value less than one from the actual index specified.

**Example**

```
>>> str1="THIRUKKURAL"
>>> print (str1[0])
```

**Output**

*T*

**(V)Stride when slicing string**

> ➢ When the slicing operation, you can specify a third argument as the stride, which refers to the number of characters to move forward after the first character is retrieved from the string.
> ➢ The default value of stride is 1.
> ➢ Python takes the last value as n-1
> ➢ You can also use negative value as stride, to prints data in reverse order.

**Example**

```
>>> str1 = "Welcome to learn Python"
>>> print(str1[::-2])
```

**Output**

**nhy re teolW**

# CHAPTER 9
## LISTS, TUPLES, SETS AND DICTIONARY
### Part – I

**Choose the best answer** **(1 Mark)**

1. Pick odd one in connection with collection data type
   (a) List     (b) Tuple     (c) Dictionary     **(d) Loop**
2. Let list1=[2,4,6,8,10], then print(List1[-2]) will result in
   (a) 10          **(b) 8**          (c) 4          (d) 6
3. Which of the following function is used to count the number of elements in a list?
   (a) count()          (b) find()          **(c)len()**          (d) index()
4. If List=[10,20,30,40,50] then List[2]=35 will result
   (a) [35,10,20,30,40,50]          (b) [10,20,30,40,50,35]
   **(c) [10,20,35,40,50]**          (d) [10,35,30,40,50]
5. If List=[17,23,41,10] then List.append(32) will result
   (a) [32,17,23,41,10]          **(b) [17,23,41,10,32]**
   (c) [10,17,23,32,41]          (d) [41,32,23,17,10]
6. Which of the following Python function can be used to add more than one element within an existing list?
   (a) append()          (b) append_more()          **(c)extend()**          (d) more()
7. What will be the result of the following Python code?
   S=[x**2 for x in range(5)]
   print(S)
   (a) [0,1,2,4,5]     **(b) [0,1,4,9,16]**     (c) [0,1,4,9,16,25]     (d) [1,4,9,16,25]
8. What is the use of type() function in python?
   (a) To create a Tuple
   (b) To know the type of an element in tuple.
   **(c) To know the data type of python object.**
   (d) To create a list.
9. Which of the following statement is not correct?
   (a) A list is mutable
   (b) A tuple is immutable.
   (c) The append() function is used to add an element.
   **(d) The extend() function is used in tuple to add elements in a list.**
10. Let setA={3,6,9}, setB={1,3,9}. What will be the result of the following snippet?
    print(setA|setB)
    (a) {3,6,9,1,3,9}          (b) {3,9}          (c) {1}          **(d) {1,3,6,9}**
11. Which of the following set operation includes all the elements that are in two sets but not the one that are common to two sets?
    **(a) Symmetric difference**          (b) Difference
    (c) Intersection          (d) Union
12. The keys in Python, dictionary is specified by
    (a) =          (b) ;          (c)+          **(d) :**

### Part - II

**Answer the following questions** **(2 Marks)**

**1. What is List in Python?**
   ➢ A list in Python is known as a **"sequence data type"** like strings.

46

- It is an ordered collection of values enclosed within square brackets [ ].
- Each value of a list is called as element.
- Element can be of any type such as numbers, characters, strings and even the nested lists as well.

## 2. How will you access the list elements in reverse order?

- Python enables reverse or negative indexing for the list elements.
- Thus, python lists index in opposite order.
- The python sets -1 as the index value for the last element in list and -2 for the preceding element and so on. This is called as **Reverse Indexing.**

## 3. What will be the value of x in following python code?

**List1=[2,4,6[1,3,5]]**

**x=len(List1)**

Answer: 4

## 4. Differentiate del with remove( ) function of List.

| del | remove( ) |
|---|---|
| **del** statement is used to delete known elements | **remove( )** function is used to delete elements of a list if its index is unknown. |
| The del statement can also be used to delete entire list. | The remove is used to delete a particular element |

## 5. Write the syntax of creating a Tuple with n number of elements.

# Tuple with n number elements

**Tuple_Name = (E1, E2, E2 ……. En)**

# Elements of a tuple without parenthesis

**Tuple_Name = E1, E2, E3 ….. En**

## 6. What is set in Python?

- In python, a set is another type of collection data type.
- A Set is a mutable and an unordered collection of elements without duplicates.
- This feature used to include membership testing and eliminating duplicate elements.

## Part - III

**Answer the following questions**                                    **(3 Marks)**

## 1. What are the advantages of Tuples over a list?

- The elements of a list are changeable (mutable) whereas the elements of a tuple are unchangeable (immutable), this is the key difference between tuples and list.
- The elements of a list are enclosed within square brackets. But, the elements of a tuple are enclosed by paranthesis.
- Iterating tuples is faster than list.

## 2. Write a shot note about sort( ).

Sort() function sorts the elements in list.

**Syntax:**

List.sort(reverse=True/False, Key=myFunc)

**Description of the syntax:**

Both arguments are optional

- If reverse is set as True, list sorting is in descending order.

> ➤ Ascending is default.
> ➤ Key=myFunc;"myFunc"-the name of the user defined function that specifies the sorting criteria.

## 3. What will be the output of the following code?

**list = [2\*\*x for x in range(5)]**
**print(list)**
> **OUTPUT:**
>> [1,2,4,8,16]

## 4. Explain the difference between del and clear( ) in dictionary with an example.

| Del | clear( ) |
|---|---|
| The **del** statement is used to delete known elements | The function **clear( )** is used to delete all the elements in list |
| The del statement can also be used to delete entire list. | It deletes only the elements and retains the list. |

## 5. List out the set operations supported by python.

**(i) Union:** It includes all elements from two or more sets
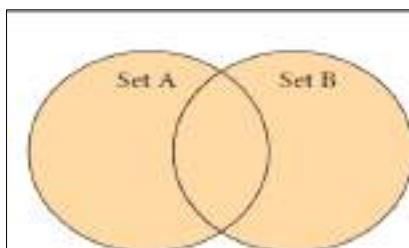


In python, the operator | is used to union of two sets. The function union( ) is also used to join two sets in python.

**(ii) Intersection:** It includes the common elements in two sets
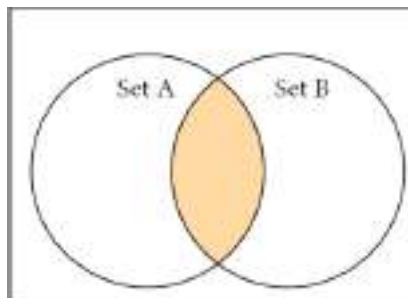


The operator & is used to intersect two sets in python. The function **intersection( )** is also used to intersect two sets in python.

**(iii) Difference** It includes all elements that are in first set (say set A) but not in the second set (say set B)
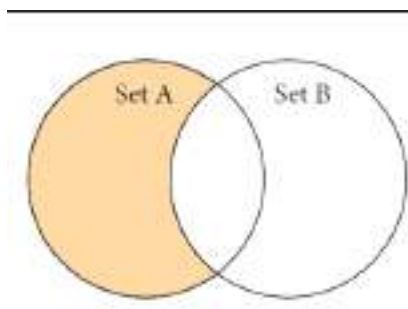
The minus (-) operator is used to difference set operation in python. The function **difference( )** is also used to difference operation.

**(iv) Symmetric difference**

It includes all the elements that are in two sets (say sets A and B) but not the one that are common to two sets.
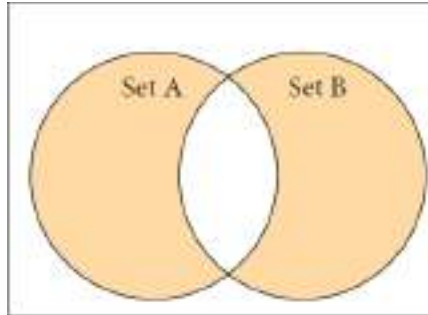


The caret (^) operator is used to symmetric difference set operation in python. The function **symmetric_difference( )** is also used to do the same operation.

## 6. What are the difference between List and Dictionary?

| List | Dictionary |
|------|-----------|
| • A list is an ordered collection of values or elements of any type . | • A dictionary is a mixed collection of elements and it stores a key along with its element. |
| • It is enclosed within square brackets [ ] | • The key value pairs are enclosed with curly braces { }. |
| • **Syntax:** <br> Variable = [element-1, element-2, element-3 ……. element-n] | • **Syntax of defining a dictionary:** <br> Dictionary_Name = { Key_1: Value_1, Key_2:Value_2, …….. Key_n:Value_n } |
| • The commas work as a separator for the elements. | • The keys in a Python dictionary is separated by a colon ( : ) while the commas work as a separator for the elements. |

**Part - IV**

**Answer the following questions** (5 Marks)

## 1. What the different ways to insert an element in a list. Explain with suitable example.

➢ The insert ( ) function is used to include an element at your desired position.

➢ The insert ( ) function is used to insert an element at any position of a list.

**Syntax:**

List.insert (position index, element)

**Example**

>>> MyList=[34,98,47,'Kannan', 'Gowrisankar', 'Lenin', 'Sreenivasan' ]

>>> MyList.insert(3, 'Ramakrishnan')

>>> print(MyList)

**Output**

[34, 98, 47, 'Ramakrishnan', 'Kannan', 'Gowrisankar', 'Lenin', 'Sreenivasan']

➤ In the above example, insert( ) function inserts a new element 'Ramakrishnan' at the index value 3, ie. at the 4<sup>th</sup> position.

➤ While inserting a new element in between the existing elements, at a particular location, the existing elements shift one position to the right .

## 2. What is the purpose of range ( )? Explain with an example.

➤ The range( ) is a function used to generate a series of values in Python.

➤ Using range( ) function, you can create list with series of values.

➤ The range( ) function has three arguments.

**Syntax of range ( ) function:**

range (start value, end value, step value)

where,

**start value** – beginning value of series. Zero is the default beginning value.

**end value** – upper limit of series. Python takes the ending value as upper limit – 1.

**step value** – It is an optional argument, which is used to generate different interval of values.

**Example : Generating whole numbers upto 10**

```
for x in range (1, 11):
print(x)
```

**Output**

```
1
2
3
4
5
6
7
8
9
10
```

## Creating a list with series of values

➤ Using the range( ) function, you can create a list with series of values.

➤ To convert the result of range( ) function into list, we need one more function called list( ).

➤ The list( ) function makes the result of range( ) as a list.

**Syntax:**

List_Varibale = list ( range ( ) )

**Example**

```
>>> Even_List = list(range(2,11,2))
>>> print(Even_List)
```

**Output**

[2, 4, 6, 8, 10]

➤ In the above code, list( ) function takes the result of range( ) as Even_List elements.

➤ Thus, Even_List list has the elements of first five even numbers.

## 3. What is nested tuple? Explain with an example.

**Tuples**

- Tuples consists of a number of values separated by comma and enclosed within parentheses.
- Tuple is similar to list, values in a list can be changed but not in a tuple.

## Nested Tuples

- In Python, a tuple can be defined inside another tuple; called Nested tuple.
- In a nested tuple, each tuple is considered as an element.
- The for loop will be useful to access all the elements in a nested tuple.

## Example

```
Toppers = (("Vinodini", "XII-F", 98.7), ("Soundarya", "XII-H", 97.5),
        ("Tharani", "XII-F", 95.3), ("Saisri", "XII-G", 93.8))
for i in Toppers:
        print(i)
```

**Output:**

('Vinodini', 'XII-F', 98.7)
('Soundarya', 'XII-H', 97.5)
('Tharani', 'XII-F', 95.3)
('Saisri', 'XII-G', 93.8)

## 4. Explain the different set operations supported by python with suitable example.

## Sets

A Set is a mutable and an unordered collection of elements without duplicates.

## Set Operations

### (i) Union:

- It includes all elements from two or more sets
- In python, the operator | is used to union of two sets.
- The function union( ) is also used to join two sets in python.
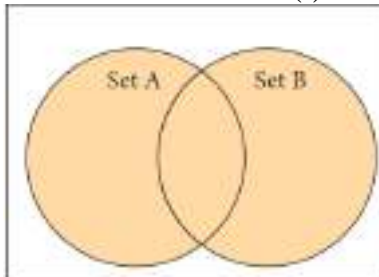


**Example:**

```
set_A={2,4,6,8}
set_B={'A', 'B', 'C', 'D'}
U_set=set_A|set_B
print(U_set)
```

**Output:**

{2, 4, 6, 8, 'A', 'D', 'C', 'B'}

### (ii) Intersection:

- It includes the common elements in two sets
- The operator & is used to intersect two sets in python.
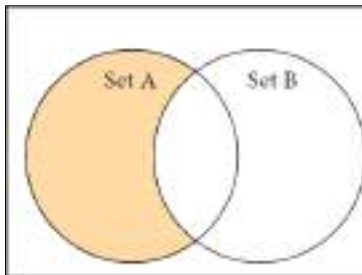- The function **intersection ( )** is also used to intersect two sets in python.
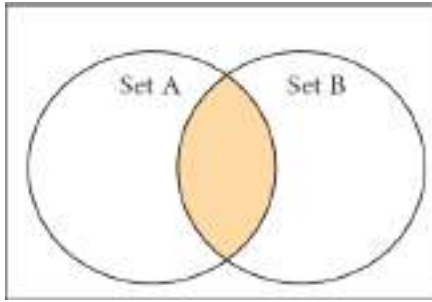
**Example:**

     set_A={'A', 2, 4, 'D'}

     set_B={'A', 'B', 'C', 'D'}

     **print(set_A & set_B)**

**Output:**    {'A', 'D'}

**(iii) Difference:**

> It includes all elements that are in first set (say set A) but not in the second set (say set B).
> The minus (-) operator is used to difference set operation in python.
> The function **difference( )** is also used to difference operation.



**Example:**

     set_A={'A', 2, 4, 'D'}

     set_B={'A', 'B', 'C', 'D'}

     **print(set_A - set_B)**

**Output:**    {2, 4}

**(iv)Symmetric difference**

> It includes all the elements that are in two sets (say sets A and B) but not the one that are common to two sets.
> The caret (^) operator is used to symmetric difference set operation in python.
> The function **symmetric_difference( )** is also used to do the same operation.
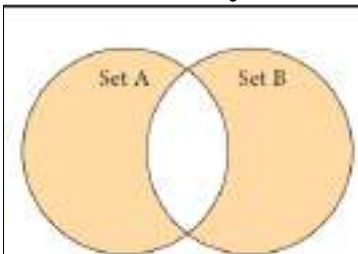


**Example:**

     set_A={'A', 2, 4, 'D'}

     set_B={'A', 'B', 'C', 'D'}

     **print(set_A ^ set_B)**

**Output:**

     {2, 4, 'B', 'C'}

52

# CHAPTER 10
## PYTHON CLASSES AND OBJECTS
### Part – I

**Choose the best answer** **(1 Mark)**

1. Which of the following are the key features of an Object Oriented Programming language?
   - (a) Constructor and Classes
   - (b) Constructor and Object
   - **(c) Classes and Objects**
   - (d) Constructor and Destructor

2. Functions defined inside a class:
   - (a) Functions
   - b) Module
   - **(c) Methods**
   - (d) section

3. Class members are accessed through which operator?
   - (a) &
   - **(b) .**
   - (c) #
   - (d) %

4. Which of the following method is automatically executed when an object is created?
   - (a) __object__( )
   - (b) __del__( )
   - (c) __func__( )
   - **(d) __init__( )**

5. A private class variable is prefixed with
   - **(a) __**
   - (b) &&
   - (c) ##
   - (d) **

6. Which of the following method is used as destructor?
   - (a) __init__( )
   - (b) __dest__( )
   - (c) __rem__( )
   - **(d) __del__( )**

7. Which of the following class declaration is correct?
   - (a) class class_name
   - (b) class class_name<>
   - **(c) class class_name:**
   - (d) class class_name[ ]

8. Which of the following is the output of the following program?
   ```
   class Student:
   def __init__(self, name):
   self.name=name
   S=Student("Tamil")
   ```
   - (a) Error
   - **(b) Tamil**
   - (c) name
   - (d) self

9. Which of the following is the private class variable?
   - **(a) __num**
   - (b) ##num
   - (c) $$num
   - (d) &&num

10. The process of creating an object is called as:
    - (a) Constructor
    - (b) Destructor
    - (c) Initialize
    - **(d) Instantiation**

### Part -II

**Answer the following questions** **(2 Marks)**

**1. What is class?**
- ➢ Class is the main building block in Python.
- ➢ Class is a template for the object.
- ➢ Object is a collection of data and function that act on those data.
- ➢ Objects are also called as instances of a class or class variable.

**2. What is instantiation?**
- ➢ Once a class is created, next you should create an object or instance of that class.
- ➢ The process of creating object is called as "Class Instantiation".

**Syntax:**
   Object_name = class_name( )

**3. What is the output of the following program?**
   **class Sample:**

53

```
        __num=10
        def disp(self):
                print(self.__num)
    S=Sample()
    S.disp()
    print(S.__num)
    Output:
        >>>
                10
                Line 7, in <module>
                        Print (S.__num)
                        Attribute Error:'Sample'object has no attribute '__num'
        >>>
```

**4. How will you create constructor in Python?**
   ➢ **"init"** is special function begin and end with double underscore in Python acts as constructor.
   ➢ Constructor function will automatically executed when an object of the class is created.

**5. What is the purpose of Destructor?**
   ➢ Destructor is also a special method gets executed automatically when an object exit from the scope. It is just opposite to constructor.
   ➢ In Python, __del__( ) method is used as destructor.

**Syntax:**
```
def __del__(self):
<statement>
```

# Part -III

**Answer the following questions**                                                    **(3 Marks)**

1. **What are class members? How do you define it?**
   ➢ Variables defined inside a class are called as "Class Variable" and functions are called as "Methods".
   ➢ Class variable and methods are together known as members of the class.
   ➢ The class members should be accessed through objects or instance of class.
   ➢ A class can be defined anywhere in a Python program.

**Syntax:**
```
class class_name:
statement_1
statement_2
…………..
 …………..
statement_n
```

2. **Write a class with two private class variables and print the sum using a method.**
   **CODE:**
```
Class Sample:
        def__init__(self,n1,n2):
                self.__n1=n1
                self.__n2=n2
```
54

```
            def sum(self):
                    print("Class Variable 1:",self.__n1)
                    print("Class Variable 2:",self.__n2)
                    print("Sum:",self.__n1+self.__n2)
            S=Sample(10,20)
            S.sum()
```
**OUTPUT:**
```
            >>>
            Class Variable 1: 10
            Class Variable 2: 20
            Sum: 30
            >>>
```
3. **Find the error in the following program to get the given output?**
   **class Fruits:**
   **def __init__(self, f1, f2):**
   **self.f1=f1**
   **self.f2=f2**
   **def display(self):**
   **print("Fruit 1 = %s, Fruit 2 = %s" %(self.f1, self.f2))**
   **F = Fruits ('Apple', 'Mango')**
   **del F.display**
   **F.display()**
   **Output**
   **Fruit 1 = Apple, Fruit 2 = Mango**
   **ERROR:**
   line 8, in <module>
           del F.display
   AttributeError: display

**CORRECT CODE:**
```
class Fruits:
    def __init__(self, f1, f2):
       self.f1=f1
       self.f2=f2
    def display(self):
       print("Fruit 1 = %s, Fruit 2 = %s" %(self.f1, self.f2))
    F = Fruits ('Apple', 'Mango')
    F.display()
```
**Output**
```
            Fruit 1 = Apple, Fruit 2 = Mango
```
4. **What is the output of the following program?**
   **class Greeting:**
   **def __init__(self, name):**
           **self.__name = name**
   **def display(self):**
           **print("Good Morning ", self.__name)**
   **obj=Greeting('Bindu Madhavan')**

55

**obj.display()**

Output:

>>>

  Good Morning Bindu Madhavan

>>>

5. **How do define constructor and destructor in Python?**
   **CONSTRUCTOR:**
   - ➤ **"init"** is special function begin and end with double underscore in Python acts as constructor.
   - ➤ Constructor function will automatically executed when an object of the class is created.

   **General format of constructor:**
   ```
   def __init__(self, [args ……..]):
           <statements>
   ```
   - ➤ Destructor is also a special method gets executed automatically when an object exit from the scope. It is just opposite to constructor.
   - ➤ In Python, __del__( ) method is used as destructor.

   **General format of destructor:**
   ```
   def __del__(self):
   <statement>
   ```

## Part -IV

**Answer the following questions**                        **(5 Marks)**

1. **Write a menu driven program to add or delete stationary items. You should use dictionary to store items and the brand.**
   **CODE:**
   ```
   stationary={ }
   print("\n1. Add Item\n2. Delete item n3. Exit")
   ch=int(input("\n Enter your choice:"))
   while(ch==1)or(ch==2):
      if(ch==1):
          n=int(input("\n Enter the Number of  Items to be added in the
                          Dictionary: "))
   for i in range(n):
       item=input("\nEnter an Item Name:")
       brand=input("\nEnter the Brand Name:")
       stationary[item]=brand
   print(stationary)
   elif(ch==2):
       ritem=input("\nEnter the item to be removed from the Dictionary:")
       stationary.pop(ritem)
       print(stationary)
   ch=int(input("\nEnter your choice:"))
   ```

# CHAPTER 11
## DATABASE CONCEPTS
### Part – I

**Choose the best answer**                                        **(1 Mark)**

1. What is the acronym of DBMS?
a) DataBase Management Symbol        b) Database Managing System
**c) DataBase Management System**       d) DataBasic Management System

2. A table is known as
a) tuple     b) attribute      **c) relation**       d) entity

3. Which database model represents parent-child relationship?
a) Relational       b) Network      **c) Hierarchical**    d) Object

4. Relational database model was first proposed by
**a) E F Codd**       b) E E Codd         c) E F Cadd      d) E F Codder

5. What type of relationship does hierarchical model represents?
a) one-to-one      **b) one-to-many**    c) many-to-one    d) many-to-many

6. Who is called Father of Relational Database from the following?
a) Chris Date    b) Hugh Darween      **c) Edgar Frank Codd**    d) Edgar Frank Cadd

7. Which of the following is an RDBMS?
a) Dbase         b) Foxpro       c) Microsoft Access        **d) SQLite**

8. What symbol is used for SELECT statement?
**a) σ**         b) Π      c) X       d) Ω

9. A tuple is also known as
a) table      **b) row**     c) attribute       d) field

10. Who developed ER model?
**a) Chen**      b) EF Codd      c) Chend    d) Chand

### Part - B

**Answer the following questions**                                **(2 Marks)**

**1. Mention few examples of a database.**
  ➢ Examples of popular DBMS: Dbase, FoxPro

**2. List some examples of RDBMS.**
➢ SQL server, Oracle, mysql, MariaDB, SQLite.

**3. What is data consistency?**
➢ Data consistency means that data values are the same at all instances of a database.
➢ On live data, it is being continuously updated and added, maintaining the consistency of data can become a challenge.
➢ But DBMS handles it by itself.

**4. What is the difference between Hierarchical and Network data model?**

| Hierarchical data model | Network data model |
|---|---|
| ➢ In hierarchical model, a child record has only one parent node | ➢ In a network model, a child may have many parent nodes. |
| ➢ It represents one-to-one relationship called Parent-child relationship in the form of tree structure. | ➢ It represents the data in many-to-many relationships. |

**5. What is normalization?**

> Normalization is an integral part of RDBMS in order to reduce data redundancy and improve data integrity.

## Part -C

**Answer the following questions** (3 Marks)

**1. What is the difference between Select and Project command?**

| Select command | Project command |
|---|---|
| The SELECT operation is used for selecting a subset with tuples according to a given condition. | The projection method defines a relation that contains a vertical subset of Relation. |
| (symbol : σ) | (symbol : Π) |
| General form σc ( R ) | Exa Πcourse (STUDENT) mple |

**2. What is the role of DBA?**
> Database Administrator or DBA is the one who manages the complete database management system.
> DBA takes care of the security of the DBMS, managing the license keys, managing user accounts and access etc.

**3. Explain Cartesian Product with a suitable example.**
> Cross product is a way of combining two relations.
> The resulting relation contains, both relations being combined.
> This type of operation is helpful to merge columns from two relations.
> **Example:** A x B means A times B, where the relation A and B have different attributes.

**4. Explain Object Model with example.**
> Object model stores the data in the form of objects, attributes and methods, classes and Inheritance.
> This model handles more complex applications, such as Geographic information System (GIS), scientific experiments, engineering design and manufacturing.
> It is used in file Management System.
> It represents real world objects, attributes and behaviors.

**5. Write a note on different types of DBMS users.**
**Database Administrators**
Database Administrator or DBA is the one who manages the complete database management system.
**Application Programmers or Software Developers**
This user group is involved in developing and designing the parts of DBMS.
**End User**
End users are the one who store, retrieve, update and delete data.
**Database designers**
They are responsible for identifying the data to be stored in the database for choosing appropriate structures to represent and store the data.

## Part -D

**Answer the following questions** (5 Marks)

**1. Explain the different types of data model.**
**Data Model**

A data model describes how the data can be represented and accessed from a software after complete implementation .

### Types of Data Model

The different types of a Data Model are

i. Hierarchical Model
ii. Relational Model
iii. Network Database Model
iv. Entity Relationship Model
v. Object Model

## Hierarchical Model

➢ In Hierarchical model, data is represented as a simple tree like structure form.
➢ This model represents a one-to-many relationship ie parent-child relationship.
➢ One child can have only one parent but one parent can have many children.
➢ This model is mainly used in IBM Main Frame computers.

### Example



## Relational Model

➢ The Relational Database model was first proposed by E.F. Codd in 1970 .
➢ The basic structure of data in relational model is tables (relations).
➢ All the information's related to a particular type is stored in rows of that table.
➢ Hence tables are also known as relations in a relational model.
➢ A relation key is an attribute which uniquely identifies a particular tuple (row in a relation (table)).



## Network Model

➢ Network database model is an extended form of hierarchical data model.
➢ In a Network model, a child may have many parent nodes.
➢ It represents the data in many-to-many relationships.
➢ This model is easier and faster to access the data.



59
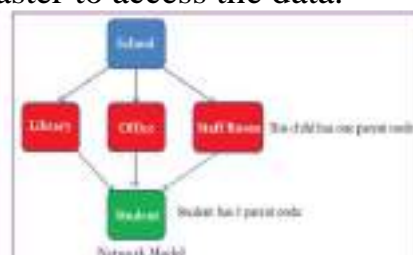
**Entity Relationship Model (ER model)**

> In this database model, relationship are created by dividing the object into entity and its characteristics into attributes.
> It was developed by Chen in 1976.
> ER model constructed by

- Rectangle represents the entities. **E.g. Doctor and Patient**
- Ellipse represents the attributes. **E.g. D-id, D-name, P-id, P-name.**
- Attributes describes the characteristics and each entity becomes a major part of the data stored in the database.
- Diamond represents the relationship in ER diagrams. **E.g. Doctor diagnosis the Patient**



ER model
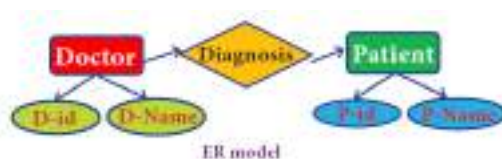
**Object Model**

> Object model stores the data in the form of objects, attributes and methods, classes and Inheritance.
> This model handles more complex applications, such as Geographic information System (GIS), scientific experiments, engineering design and manufacturing.



Object Model
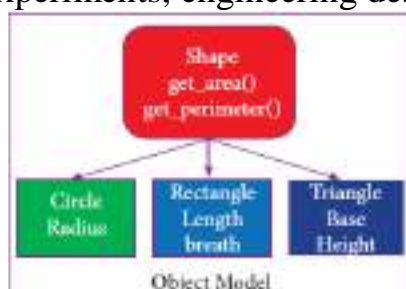
**2. Explain the different types of relationship mapping.**

**Types of Relationships**

Following are the types of relationships used in a database.
1. One-to-One Relationship
2. One-to-Many Relationship
3. Many-to-One Relationship
4. Many-to-Many Relationship

   **i. One-to-One Relationship**

> In One-to-One Relationship, one entity is related with only one other entity.
> One row in a table is linked with only one row in another table and vice versa.
> For example: A student can have only one exam number



One to one Relationship
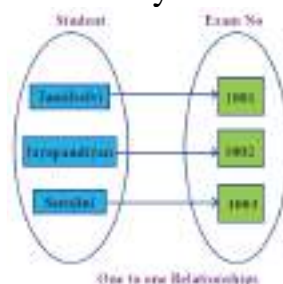
   **ii. One-to-Many Relationship**

> In One-to-Many relationship, one entity is related to many other entities.
> One row in a table A is linked to many rows in a table B, but one row in a table B is linked to only one row in table A.
> For example: One Department has many staff members.



One to Many Mapping

### iii. Many-to-One Relationship

> In Many-to-One Relationship, many entities can be related with only one in the other entity.
> For example: A number of staff members working in one Department.
> Multiple rows in staff members table is related with only one row in Department table.



Many to one Relationship

### iv. Many-to-Many Relationship

> A many-to-many relationship occurs when multiple records in a table are associated with multiple records in another table.
> **Example: Books and Student.** Many Books in a Library are issued to many students.



Many to Many Relationship
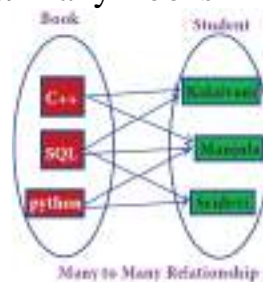
## 3. Differentiate DBMS and RDBMS.

| Basis of Comparison | DBMS | RDBMS |
|---|---|---|
| Expansion | Database Management Sytem | Relational DataBase Management System |
| Data storage | Navigational model ie data by linked records | Relational model (in tables).ie data in tables as row and column |
| Data redundancy | Exhibit | Not present |
| Normalization | Not performed | RDBMS uses normalization to reduce redundancy |

| Data access | Consumes more time | Faster, compared to DBMS. |
|---|---|---|
| Keys and indexes | Does not use. | Used to establish relationship. Keys are used in RDBMS. |
| Transaction management | Inefficient, error prone and insecure. | Efficient and secure. |
| Distributed databases | Not supported | Supported by RDBMS. |
| Example | Dbase, FoxPro. | SQL server, Oracle, musql, MariaDB, SQlite |

## 4. Explain the different operators in Relational algebra with suitable examples.

- It was used for modeling the data stored in relational databases and defining queries on it.
- Relational Algebra is divided into various groups

### 1. Unary Relational Operations

- SELECT ( symbol : $\sigma$)
- PROJECT ( symbol : $\Pi$)

### 2. Relational Algebra Operations from Set Theory

- UNION ($\cup$)
- INTERSECTION ($\cap$)
- DIFFERENCE ($-$)
- CARTESIAN PRODUCT (X)

### i. SELECT (symbol : $\sigma$)

- General form $\sigma_c$ ( R ) with a relation R and a condition C on the attributes of R.
- The SELECT operation is used for selecting a subset with tuples according to a given condition.
- Select filters out all tuples that do not satisfy C.
- $\sigma_{course}$ = "Big Data" **(STUDENT )**

### ii. PROJECT (symbol : $\Pi$)

- The projection eliminates all attributes of the input relation but those mentioned in the projection list. The projection method defines a relation that contains a vertical subset of Relation.
- **Example** $\Pi$course (STUDENT)

### iii. UNION (Symbol :$\cup$)

- It includes all tuples that are in tables A or in B.
- It also eliminates duplicates.
- Set A Union Set B would be expressed as A $\cup$ B

### iv. SET DIFFERENCE ( Symbol : - )

- The result of A – B, is a relation which includes all tuples that are in A but not in B.
- The attribute name of A has to match with the attribute name in B.

### v. INTERSECTION (symbol : $\cap$) A $\cap$ B

- Defines a relation consisting of a set of all tuple that are in both in A and B.
- However, A and B must be union-compatible.

### vi. PRODUCT OR CARTESIAN PRODUCT (Symbol : X )

62

- Cross product is a way of combining two relations.
- The resulting relation contains, both relations being combined.
- A x B means A times B, where the relation A and B have different attributes.
- This type of operation is helpful to merge columns from two relations.

**5.   Explain the characteristics of DBMS.**

| | |
|---|---|
| **Data stored into Tables** | Data is never directly stored into the database. Data is stored into tables, created inside the database. DBMS also allows to have relationship between tables which makes the data more meaningful and connected. |
| **Reduced Redundancy** | In the modern world hard drives are very cheap, but earlier when hard drives were too expensive, unnecessary repetition of data in database was a big problem But DBMS follows Normalisation which divides the data in such a way that repetition is minimum. |
| **Data Consistency** | On live data, it is being continuously updated and added, maintaining the consistency of data can become a challenge. But DBMS handles it by itself. |
| **Support Multiple user and Concurrent Access** | DBMS allows multiple users to work on it(update, insert, delete data) at the same time and still manages to maintain the data consistency. |
| **Query Language** | DBMS provides users with a simple query language, using which data can be easily fetched, inserted, deleted and updated in a database. |
| **Security** | The DBMS also takes care of the security of data, protecting the data from unauthorized access. In a typical DBMS, we can create user accounts with different access permissions, using which we can easily secure our data by restricting user access. |
| **DBMS Supports Transactions** | It allows us to better handle and manage data integrity in real world applications where multi-threading is extensively used. |

# CHAPTER 12
## STRUCTURED QUERY LANGUAGE

## Part - I

**Choose the best answer**                                          **(1 Marks)**

1. Which commands provide definitions for creating table structure, deleting relations, and modifying relation schemas?

   **a. DDL**         b. DML         c. DCL         d. DQL

2. Which command lets to change the structure of the table?

   a. SELECT    b. ORDER BY      c. MODIFY          **d. ALTER**

3. The command to delete a table is

   A) **DROP**    B) DELETE           C) DELETE ALL         D) ALTER TABLE

4. Queries can be generated using

   **a. SELECT**           b. ORDER BY     c. MODIFY         d. ALTER

5. The clause used to sort data in a database

   a. SORT BY                **b. ORDER BY**    c. GROUP BY      d. SELECT

## Part -II

**Answer the following questions**                                **(2 Marks)**

**1. Write a query that selects all students whose age is less than 18 in order wise.**

**Query:**      SELECT * FROM Student WHERE Age<=18 ORDER BY Name;

**2. Differentiate Unique and Primary Key constraint.**

| Unique key constraint | Primary key constraint |
|---|---|
| • This constraint ensures that no two rows have the same value in the specified columns. | • This constraint declares a field as a Primary key which helps to uniquely identify a record. |
| • The **UNIQUE** constraint can be applied only to fields that have also been declared as **NOT NULL**. | • The primary key does not allow **NULL** values and therefore a field declared as primary key must have the **NOT NULL** constraint. |

**3. Write the difference between table constraint and column constraint?**

| Table constraint | Column constraint |
|---|---|
| Table constraints apply to a group of one or more columns | Column constraints apply only to individual column |

**4. Which component of SQL lets insert values in tables and which lets to create a table?**

| Command | Description | Component |
|---|---|---|
| **Insert** | Insert data into a table | DML |
| **Create** | To create tables in the database | DDL |

**5. What is the difference between SQL and MySQL?**

| SQL | MySQL |
|---|---|
| • Structured Query Language is a language used for accessing databases. | • MySQL is a database management system, like SQL Server, Oracle, |

| | Informix, Postgres, etc. |
|---|---|
| • SQL is a DBMS | • MySQL is a RDBMS. |

## Part -III

**Answer the following questions** (3 Marks)

### 1. What is a constraint? Write short note on Primary key constraint.

➢ Constraint is a condition applicable on a field or set of fields.

➢ Primary constraint declares a field as a Primary key which helps to uniquely identify a record.

➢ It is similar to unique constraint except that only one field of a table can be set as primary key.

➢ The primary key does not allow **NULL** values and therefore a field declared as primary key must have the **NOT NULL** constraint.

### 2. Write a SQL statement to modify the student table structure by adding a new field.

*ALTER TABLE <table-name> ADD <column-name><data type><size>;*

To add a new column "Address" of type 'char' to the Student table, the command is used as

**ALTER TABLE Student ADD Address char;**

### 3. Write any three DDL commands.

**Data Definition Language**

**Create Command:** To create tables in the database.

    CREATE TABLE Student (Admno integer,Name char(20),Gender char(1), Age integer,);

**Alter Command: Alters the structure of the database.**

    **ALTER TABLE Student ADD Address char;**

**Drop Command: Delete tables from database.**

    DROP TABLE Students;

### 4. Write the use of Savepoint command with an example.

➢ The **SAVEPOINT** command is used to temporarily save a transaction so that you can roll back to the point whenever required.

**Syntax:**    SAVEPOINT savepoint_name;

**Example:**  SAVEPOINT A;

### 5. Write a SQL statement using DISTINCT keyword.

➢ The **DISTINCT** keyword is used along with the **SELECT** command to eliminate duplicate rows in the table.

➢ This helps to eliminate redundant data.

➢ For Example: **SELECT DISTINCT Place FROM Student;**

## Part -IV

**Answer the following questions** (5 Marks)

1. Write the different types of constraints and their functions.

➢ Constraint is a condition applicable on a field or set of fields.

> **Type of Constraints**
> ➢ **Unique Constraint**
> ➢ **Primary Key Constraint**
> ➢ **Default Constraint**
> ➢ **Check Constraint**
> ➢ **Table Constraint**

65

i. **Unique Constraint**
- This constraint ensures that no two rows have the same value in the specified columns.
- For example **UNIQUE** constraint applied on Admno of student table ensures that no two students have the same admission number and the constraint can be used as:

  **CREATE TABLE Student**
  (
  Admno integer NOT NULL UNIQUE, → **Unique constraint**
  Name char (20) NOT NULL,
  Gender char (1),
  Age integer,
  Place char (10),
  );
- The **UNIQUE** constraint can be applied only to fields that have also been declared as **NOT NULL**.
- When two constraints are applied on a single field, it is known as multiple constraints.
- In the above Multiple constraints **NOT NULL** and **UNIQUE** are applied on a single field Admno.

ii. **Primary Key Constraint**
- Primary constraint declares a field as a Primary key which helps to uniquely identify a record.
- It is similar to unique constraint except that only one field of a table can be set as primary key.

The primary key does not allow **NULL** values and therefore a field declared as primary key must have the **NOT NULL** constraint.
- **Example**
  **CREATE TABLE Student**
  (
  Admno integer NOT NULL PRIMARY KEY, → Primary Key constraint
  Name char(20)NOT NULL,
  Gender char(1),
  Age integer,
  Place char(10),
  );

iii. **Default Constraint**
- The **DEFAULT** constraint is used to assign a default value for the field.
- When no value is given for the specified field having **DEFAULT** constraint, automatically the default value will be assigned to the field.
- Example:
  **CREATE TABLE Student**
  (
  Admno integer NOT NULL PRIMARY KEY,
  Name char(20)NOT NULL,
  Gender char(1),
  Age integer DEFAULT = "17", → **Default Constraint**

66

Place char(10),
);
- In the above example the "Age" field is assigned a default value of 17, therefore when no value is entered in age by the user, it automatically assigns 17 to Age.

### iv. Check Constraint
- This constraint helps to set a limit value placed for a field.
- When we define a check constraint on a single column, it allows only the restricted values on that field.
- Example
  **CREATE TABLE Student**
  (
  Admno integer NOT NULL PRIMARY KEY
  Name char(20)NOT NULL,
  Gender char(1),
  Age integer (CHECK<=19), → **Check Constraint**
  Place char(10),
  );
- In the above example the check constraint is set to Age field where the value of Age must be less than or equal to 19.

### v. Table Constraint
- ➢ When the constraint is applied to a group of fields of the table, it is known as Table constraint.
- ➢ The table constraint is normally given at the end of the table definition. Let us take a new table namely Student1 with the following fields Admno, Firstname, Lastname, Gender, Age, Place:

**2. Consider the following employee table. Write SQL commands for the qtns.(i) to (v).**

| EMP CODE | NAME | DESIG | PAY | ALLO WANCE |
|----------|------|-------|-----|------------|
| S1001 | Hariharan | Supervisor | 29000 | 12000 |
| P1002 | Shaji | Operator | 10000 | 5500 |
| P1003 | Prasad | Operator | 12000 | 6500 |
| C1004 | Manjima | Clerk | 8000 | 4500 |
| M1005 | Ratheesh | Mechanic | 20000 | 7000 |

i. **To display the details of all employees in descending order of pay.**
SELECT * FROM employee ORDER BY DESC;
ii. **To display all employees whose allowance is between 5000 and 7000.**
SELECT * FROM employee WHERE allowance BETWEEN 5000 AND 7000;
iii. **To remove the employees who are mechanic.**
DELETE FROM employee WHERE desig='Mechanic';
iv. **To add a new row.**
INSERT INTO employee
(empcode,name,desig,pay,allowance)VALUES(S1002,Baskaran,Superisor,29000,12000);
v. **To display the details of all employees who are operators.**
SELECT *FROM employee WHERE design='Operator';
3. **What are the components of SQL? Write the commands in each.**

**Components of SQL**



## DATA MANIPULATION LANGUAGE

**A Data Manipulation Language (DML)** is a computer programming language used for adding (inserting), removing (deleting), and modifying (updating) data in a database.

**By Data Manipulation we mean,**
- Insertion of new information into the database
- Retrieval of information stored in a database.
- Deletion of information from the database.
- Modification of data stored in the database.

## DATA DEFINITION LANGUAGE

➢ The Data Definition Language (DDL) consists of SQL statements used to define the database structure or schema.

➢ It simply deals with descriptions of the database schema and is used to create and modify the structure of database objects in databases.

➢ The DDL provides a set of definitions to specify the storage structure and access methods used by the database system.

**SQL commands which comes under Data Definition Language are:**

| Create | To create tables in the database. |
|---|---|
| Alter | Alters the structure of the database. |
| Drop | Delete tables from database. |
| Truncate | Remove all records from a table, also release the space occupied by those records. |

## DATA CONTROL LANGUAGE

➢ A **Data Control Language (DCL)** is a programming language used to control the access of data stored in a database.

➢ It is used for controlling privileges in the database (Authorization).

➢ The privileges are required for performing all the database operations such as creating sequences, views of tables etc.

**SQL commands which come under Data Control Language are:**

| Grant | Grants permission to one or more users to perform specific tasks. |
|---|---|
| Revoke | Withdraws the access permission given by the GRANT statement. |

## TRANSACTIONAL CONTROL LANGUAGE

➢ **Transactional control language (TCL)** commands are used to manage transactions in the database.

➢ These are used to manage the changes made to the data in a table by DML statements.

**SQL command which come under Transfer Control Language are:**

| Commit | Saves any transaction into the database permanently. |
|---|---|
| **Roll back** | Restores the database to last commit state. |
| **Save point** | Temporarily save a transaction so that you can rollback. |

# DATA QUERY LANGUAGE

➢ The Data Query Language consists of commands used to query or retrieve data from a database. One such SQL command in Data Query Language is

| Select : | It displays the records from the table. |
|---|---|

4. **Construct the following SQL statements in the student table-**
   i. **SELECT statement using GROUP BY clause.**

   SELECT Gender FROM Student GROUP BY Gender;
   **OUTPUT:**

   | Gender |
   |---|
   | Male |
   | Female |

   SELECT Gender, count(*) FROM Student GROUP BY male;
   **OUPTPUT:**

   | Gender | Count(*) |
   |---|---|
   | Male | 5 |
   | Female | 3 |

   ii. **SELECT statement using ORDER BY clause.**

   SELECT * FROM student WHERE Age>=18 ORDER BY Name DESC;
   **OUTPUT:**

   

5. **Write a SQL statement to create a table for employee having any five fields and create a table constraint for the employee table.**

   ```
   CREATE TABLE employee
   (
   Empno integer NOT NULL,
   Name char(20),
   Desig char(20),
   Pay integer,
   Allowance integer,
   PRIMARY KEY (empno)
   );
   ```

# CHAPTER 13
## PYTHON AND CSV FILES

**Choose the best answer**                                 **(1 Marks)**

1. A CSV file is also known as a ….
   **(A) Flat File**      (B) 3D File      (C) String File      (D) Random File
2. The expansion of CRLF is
   (A) Control Return and Line Feed
   (B) Carriage Return and Form Feed
   (C) Control Router and Line Feed
   **(D) Carriage Return and Line Feed**
3. Which of the following module is provided by Python to do several operations on the CSV files?
   (A) py      (B) xls      **(C) csv**      (D) os
4. Which of the following mode is used when dealing with non-text files like image or exe files?
   (A) Text mode      **(B) Binary mode**      (C) xls mode      (D) csv mode
5. The command used to skip a row in a CSV file is
   **(A) next()**      (B) skip()      (C) omit()      (D) bounce()
6. Which of the following is a string used to terminate lines produced by writer()method of csv module?
   **(A) Line Terminator**      (B) Enter key      (C) Form feed      (D) Data Terminator
7. What is the output of the following program? import csv

d=csv.reader(open('c:\PYPRG\ch13\city.csv'))

next(d)

for row in d:

print(row)

if the file called "city.csv" contain the following details

   chennai,mylapore

  mumbai,andheri

 A) chennai,mylapore **(B) mumbai,andheri** (C) chennai (D) chennai,mylapore

 mumba mumbai,andheri

8. Which of the following creates an object which maps data to a dictionary?
   (A) listreader()`      (B) reader()      (C) tuplereader()      **(D) DicReader ()**
9. Making some changes in the data of the existing file or adding more data is called
   (A)Editing      (B) Appending      **(C)Modification**      (D) Alteration
10. What will be written inside the file test.csv using the following program
    ```
    import csv
    D = [['Exam'],['Quarterly'],['Halfyearly']]
    csv.register_dialect('M',lineterminator = '\n')
    with open('c:\pyprg\ch13\line2.csv', 'w') as f:
    wr = csv.writer(f,dialect='M')
    wr.writerows(D)
    f.close()
    ```

| (A) Exam Quarterly Halfyearly | (B) Exam Quarterly Halfyearly |
|---|---|

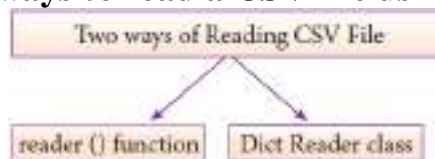| (C) E | (D) Exam, |
|---|---|
| Q | Quarterly, |
| H | Halfyearly |

## Part - II

**Answer the following questions** (2 Marks)

1. **What is CSV File?**

➢ A CSV file is a human readable text file where each line has a number of fields, separated by commas or some other delimiter.

➢ A CSV file is also known as a Flat File. Files in the CSV format can be imported to and exported from programs that store data in tables, such as **Microsoft Excel** or **OpenOfficeCalc**

2. **Mention the two ways to read a CSV file using Python.**



3. **Mention the default modes of the File.**

➢ **The default is reading in text mode.**

➢ In this mode, while reading from the file the data would be in the format of **strings.**

4. **What is use of next() function?**

➢ **"next()" command** is used to avoid or skip the first row or row heading.

➢ **Example:** While sorting the row heading is also get sorted, to avoid that the first is skipped using next().

➢ Then the list is sorted and displayed.

5. **How will you sort more than one column from a csv file?Give an example statement**.

➢ To sort by more than one column you can use **itemgetter** with muliptle indices.

**Syntax: operator.itemgetter(col_no)**

Example: sortedlist = sorted (data, key=operator.itemgetter(1))

## Part - III

**Answer the following questions** (3 Marks)

1. **Write a note on open() function of python. What is the difference between the two methods?**

➢ **Python has a built-in function open() to open a file.**

➢ This function returns a **file object,** also called a **handle,** as it is used to read or modify the file accordingly.

➢ **The default is reading in text mode.** In this mode, while reading from the file the data would be in the format of **strings.**

➢ On the other hand, **binary mode returns bytes and this is the mode to be used when dealing with non-text files like image or exe files.**

2. **Write a Python program to modify an existing file.**

➢ In this program, the third row of "student.csv" is modified and saved.

➢ First the "student.csv" file is read by using csv.reader() function.

➢ Then, the list() stores each row of the file.

➢ The statement "lines[3] = row", changed the third row of the file with the new content in "row".

71

➢ The file object writer using write rows (lines) writes the values of the list to "student.csv" file.

➢ **PROGRAM:**

```
import csv
row = ['3', 'Meena','Bangalore']
with open('student.csv', 'r') as readFile:
reader = csv.reader(readFile)
lines = list(reader) # list()- to store each row of data as a list
lines[3] = row
with open('student.csv', 'w') as writeFile:
# returns the writer object which converts the user data with delimiter
writer = csv.writer(writeFile)
#writerows()method writes multiple rows to a csv file
writer.writerows(lines)
readFile.close()
writeFile.close()
```

3. **Write a Python program to read a CSV file with default delimiter comma (,)**

```
#importing csv
import csv
#opening the csv file which is in different location with read mode
with open('c:\\pyprg\\sample1.csv', 'r') as F:
#other way to open the file is f= ('c:\\pyprg\\sample1.csv', 'r')
reader = csv.reader(F)
# printing each line of the Data row by row
print(row)
F.close()
```

**OUTPUT**
['SNO', 'NAME', 'CITY']
['12101', 'RAM', 'CHENNAI']
['12102', 'LAVANYA', 'TIRUCHY']
['12103', 'LAKSHMAN', 'MADURAI']

4. **What is the difference between the write mode and append mode.**

| WRITE MODE | APPEND MODE |
|---|---|
| • 'w' | • 'a' |
| • Open a file for writing. | • Open for appending at the end of the file without truncating it. |
| • Creates a new file if it does not exist or truncates the file if it exists. | • Creates a new file if it does not exist. |

5. **What is the difference between reader() and DictReader() function?**

**reader()**

➢ The reader function is designed to take each line of the file and make a list of all columns.

➢ Using this method one can read data from csv files of different formats like quotes (" "), pipe (|) and comma (,).

72

> Csv.Reader work with list/tuple.
> Syntax: sv.reader(fileobject,delimiter,fmtparams)

**DictReader()**

> **DictReader** works by reading the first line of the CSV and using each comma separated value in this line as a **dictionary key.**
> **DictReader** class of csv module is used to read a CSV file into a dictionary.
> It creates an object which maps data to a dictionary.
> csv.DictReader work with dictionary.

## Part - IV

**Answer the following questions** (5 Marks)

### 1. Differentiate Excel file and CSV file.

| Excel | CSV |
|---|---|
| Excel is a binary file that holds information about all the worksheets in a file, including both content and formatting | CSV format is a plain text format with a series of values separated by commas. |
| XLS files can only be read by applications that have been especially written to read their format, and can only be written in the same way. | CSV can be opened with any text editor in Windows like notepad, MS Excel, OpenOffice, etc. |
| Excel is a spreadsheet that saves files into its own proprietary format viz. xls or xlsx | CSV is a format for saving tabular information into a delimited text file with extension .csv |
| Excel consumes more memory while importing data | Importing CSV files can be much faster, and it also consumes less memory |

### 2. Tabulate the different mode with its meaning.

| Mode | Description |
|---|---|
| 'r' | Open a file for reading. (default) |
| 'w' | Open a file for writing. Creates a new file if it does not exist or truncates the file if it exists. |
| 'x' | Open a file for exclusive creation. If the file already exists, the operation fails. |
| 'a' | Open for appending at the end of the file without truncating it. Creates a new file if it does not exist. |
| 't' | Open in text mode. (default) |
| 'b' | Open in binary mode. |
| '+' | Open a file for updating (reading and writing) |

### 3. Write the different methods to read a File in Python.

The contents of CSV file with the help of csv.reader() method.

The reader function is designed to take each line of the file and make a list of all columns.

Using this method one can read data from csv files of different formats like

1. CSV file - data with default delimiter comma (,)
2. CSV file - data with Space at the beginning
3. CSV file - data with quotes
4. CSV file - data with custom Delimiters

73

The syntax for csv.reader() is csv.reader(fileobject,delimiter,fmtparams)

**i)CSV file with default delimiter comma (,)**

The following program read a file called "sample1.csv" with default delimiter comma (,) and print row by row.

```
#importing csv
import csv
#opening the csv file which is in different location with read mode
with open('c:\\pyprg\\sample1.csv', 'r') as F:
#other way to open the file is f= ('c:\\pyprg\\sample1.csv', 'r')
reader = csv.reader(F)
# printing each line of the Data row by row
print(row)
F.close()
```

**OUTPUT**

```
['SNO', 'NAME', 'CITY']
['12101', 'RAM', 'CHENNAI']
['12102', 'LAVANYA', 'TIRUCHY']
['12103', 'LAKSHMAN', 'MADURAI']
```

**ii)CSV files- data with Spaces at the beginning**

Consider the following file "sample2.csv" containing the following data when opened through notepad



The following program read the file through Python using "csv.reader()".

```
import csv
csv.register_dialect('myDialect',delimiter = ',',skipinitialspace=True)
F=open('c:\\pyprg\\sample2.csv','r')
reader = csv.reader(F, dialect='myDialect')
for row in reader:
print(row)
F.close()
```

**OUTPUT**

```
['Topic1', 'Topic2', 'Topic3']
['one', 'two', 'three']
['Example1', 'Example2', 'Example3']
```

➤ These whitespaces can be removed, by registering new dialects using csv.register_dialect() class of csv module.
➤ A dialect describes the format of the csv file that is to be read.
➤ In dialects the parameter "skipinitialspace" is used for removing whitespaces after the delimiter.

**iii)CSV File-Data With Quotes**

➤ You can read the csv file with quotes, by registering new dialects using csv.register_dialect() class of csv module.
➤ Here, we have quotes.csv file with following data.

74

SNO,Quotes
1, "The secret to getting ahead is getting started."
2, "Excellence is a continuous process and not an accident."

The following Program read "quotes.csv" file, where delimiter is comma (,) but the quotes are within quotes (" ").

```
import csv
csv.register_dialect('myDialect',delimiter = ',',quoting=csv.QUOTE_ALL,
skipinitialspace=True)
f=open('c:\\pyprg\\quotes.csv','r')
reader = csv.reader(f, dialect='myDialect')
for row in reader:
print(row)
```

**OUTPUT**

```
['SNO', 'Quotes']
['1', 'The secret to getting ahead is getting started.']
['2', 'Excellence is a continuous process and not an accident.']
```

In the above program, register a dialect with name myDialect.

Then, we used csv. QUOTE_ALL to display all the characters after double quotes.

## iv)CSV files with Custom Delimiters

You can read CSV file having custom delimiter by registering a new dialect with the help of csv.register_dialect().

| Roll No | Name | City |
|---------|------|------|
| 12101 | Arun | Chennai |
| 12102 | Meena | Kovai |
| 12103 | Ram | Nellai |
| 103 | Ayush | M |
| 104 | Abinandh | M |

The following program read the file "sample4.csv" with user defined delimiter "|"



```
import csv
csv.register_dialect('myDialect', delimiter = '|')
with open('c:\\pyprg\\sample4.csv', 'r') as f:
    reader = csv.reader(f, dialect='myDialect')
    for row in reader:
        print(row)
f.close()
```

```
OUTPUT
['RollNo', 'Name', 'City']
['12101', 'Arun', 'Chennai']
['12102', 'Meena', 'Kovai']
['12103', 'Ram', 'Nellai']
```

**4. Write a Python program to write a CSV File with custom quotes.**

```
import csv
info = [['SNO', 'Person', 'DOB'],
['1', 'Madhu', '18/12/2001'],
['2', 'Sowmya','19/2/1998'],
['3', 'Sangeetha','20/3/1999'],
['4', 'Eshwar', '21/4/2000'],
['5', 'Anand', '22/5/2001']]
csv.register_dialect('myDialect',quoting=csv.QUOTE_ALL)
with open('c:\\pyprg\\ch13\\person.csv', 'w') as f:
writer = csv.writer(f, dialect='myDialect')
for row in info:
writer.writerow(row)
f.close()
```

75

**OUTPUT**
"SNO","Person","DOB" "1","Madhu","18/12/2001"
"2","Sowmya","19/2/1998" "3","Sangeetha","20/3/1999"
"4","Eshwar","21/4/2000"
"5","Anand","22/5/2001"

## 5. Write the rules to be followed to format the data in a CSV file.

1. Each record (row of data) is to be located on a separate line, delimited by a line break by pressing enter key.

**For example:**

    xxx,yyy ↵

  ↵ denotes enter Key to be pressed

2. The last record in the file may or may not have an ending line break.

**For example:**

ppp, qqq ↵

yyy, xxx

3. There may be an optional header line appearing as the first line of the file with the same format as normal record lines.

The header will contain names corresponding to the fields in the file and should contain the same number of fields as the records in the rest of the

file.

**For example:** field_name1,field_name2,field_name3

aaa,bbb,ccc ↵

zzz,yyy,xxx CRLF( Carriage Return and Line Feed)

4. Within the header and each record, there may be one or more fields, separated by commas. Spaces are considered part of a field and should not be ignored. The last field in the record must not be followed by a comma.

**For example:** Red , Blue

5. Each field may or may not be enclosed in double quotes. If fields are not enclosed with double quotes, then double quotes may not appear inside the fields.

**For example:**

```
"Red","Blue","Green"↵      #Field data with doule quotes

Black,White,Yellow         #Field data without doule quotes
```

6. Fields containing line breaks (CRLF), double quotes, and commas should be enclosed in double-quotes.

**For example:**

```
Red, ",", Blue CRLF   # comma itself is a field value.so it is enclosed with double quotes

Red, Blue , Green
```

7. If double-quotes are used to enclose fields, then a double-quote appearing inside a field must be preceded with another double quote.

**For example:**

```
"Red, " "Blue", "Green",     # since double quotes is a field value it is enclosed with another double quotes

,, White
```

# CHAPTER 14
## IMPORTING C++ PROGRAMS IN PYTHON
### Part - I

**Choose the best answer**                                     **(1 Marks)**

1. Which of the following is not a scripting language?
   (A) JavaScript     (B) PHP     (C) Perl    **(D) HTML**

2. Importing C++ program in a Python program is called
   **(A) wrapping**           (B) Downloading
   (C) Interconnecting       (D) Parsing

3. The expansion of API is
   (A) Application Programming Interpreter
   **(B) Application Programming Interface**
   (C) Application Performing Interface
   (D) Application Programming Interlink

4. A framework for interfacing Python and C++ is
   (A) Ctypes      (B) SWIG      (C) Cython      **(D) Boost**

5. Which of the following is a software design technique to split your code into separate parts?
   (A) Object oriented Programming
   **(B) Modular programming**
   (C) Low Level Programming
   (D) Procedure oriented Programming

6. The module which allows you to interface with the Windows operating system is
   **(A) OS module**     (B) sys module     (c) csv module     (d) getopt module

7. getopt() will return an empty array if there is no error in splitting strings to
   (A) argv variable     (B) opt variable    **(c)args variable**    (d) ifile variable

8. Identify the function call statement in the following snippet.
   if __name__ =='__main__':
   main(sys.argv[1:])
   (A) main(sys.argv[1:])    **(B) __name__**     (C) __main__     (D) argv

9. Which of the following can be used for processing text, numbers, images, and scientific data?
   (A) HTML     (B) C     (C) C++     **(D) PYTHON**

10. What does __name__ contains ?
    (A) c++ filename    (B) main() name    **(C) python filename**    (D) os module name

### Part - II

**Answer the following questions**                           **(2 Marks)**

1. **What is the theoretical difference between Scripting language and other programming language?**

| Scripting language | Programming language |
|---|---|
| A scripting language requires an interpreter. | A programming language requires a compiler. |
| A scripting language need not be compiled. | A programming language needs to be compiled before running. |
| Example: JavaScript, VBScript, PHP, Perl, | Example: C,C++,Java, C# etc. |

| Python, Ruby, ASP and Tel. | |
|---|---|

2. **Differentiate compiler and interpreter.**

| Compiler | Interpreter |
|---|---|
| Complier generates an Intermediate Code. | Interpreter generates Machine Code. |
| Complier reads entire program for compilation. | Interpreter reads single statement at a time for interpretation. |
| Error deduction is difficult | Error deduction is easy |
| Comparatively faster | Slower |
| Example: gcc, g++. Borland TurboC | Example: Python, Basic, Java |

3. **Write the expansion of (i) SWIG (ii) MinGW**
   **SWIG-** Simplified Wrapper Interface Generator- Both C and C++
   **MinGW-**Minimalist GNU for Windows

4. **What is the use of modules?**
   ➢ Modules are used to break down large programs into small manageable and organized files.
   ➢ Modules provide reusability of code.
   ➢ We can define our most used functions in a module and import it, instead of copying their definitions into different programs.

5. **What is the use of cd command. Give an example.**
   **Syntax:** cd <absolute path>

## Part - III

**Answer the following questions** (3 Marks)

1. **Differentiate PYTHON and C++ .**

| PYTHON | C++ |
|---|---|
| Python is typically an "interpreted" language | C++ is typically a "compiled" language |
| Python is a dynamic-typed language | C++ is compiled statically typed language |
| Data type is not required while declaring variable | Data type is required while declaring variable |
| It can act both as scripting and general purpose language | It is a general purpose language |

2. **What are the applications of scripting language?**
   1.To automate certain tasks in a program
   2. Extracting information from a data set
   3. Less code intensive as compared to traditional programming language
   4. can bring new functions to applications and glue complex systems together

3. **What is MinGW? What is its use?**
   • MinGW refers to a set of runtime header files.
   • It is used in compiling and linking the code of C, C++ and FORTRAN to be run on Windows Operating System.
   • MinGW allows to compile and execute C++ program dynamically through Python program using g++.

4. **Identify the module ,operator, definition name for the following welcome.display()**

Welcome -> module name

. -> Dot operator

Display() -> function call

**5. What is sys.argv? What does it contain?**

- sys.argv is the list of command-line arguments passed to the Python program.
- **argv contains** all the items that come along via the command-line input,      it's basically an array holding the command-line arguments of the program.
- To use sys.argv, import sys should be used.
- The first argument, sys.argv[0] contains the name of the python program and sys.argv[1]is the next argument passed to the program.
- Main(sys.argv[1])
    - ➢ The input file (c++ file) is send along with its path as a list (array) using argv[1].
    - ➢ Argv[0] contains the python program which is need not to be passed because by default_main_contains source code reference.

## Part - IV

**Answer the following questions                                                                      (5 Marks)**

**1. Write any 5 features of Python.**

- ➢ Python uses Automatic Garbage Collection whereas C++ does not.
- ➢ C++ is a statically typed language, while Python is a dynamically typed language.
- ➢ Python runs through an interpreter, while C++ is pre-compiled.
- ➢ Python code tends to be 5 to 10 times shorter than that written in C++.
- ➢ In Python, there is no need to declare types explicitly where as it should be done in C++.
- ➢ In Python, a function may accept an argument of any type, and return multiple values without any kind of declaration beforehand. Whereas in C++ return statement can return only one value.

**2. Explain each word of the following command.**

**Python <filename.py> -<i> <C++ filename without cpp extension>**

| Python | keyword to execute the Python program from command-line |
| --- | --- |
| filename.py | Name of the Python program to executed |
| - i | input mode |
| C++ filename without cpp extension | name of C++ file to be compiled and executed |

**3. What is the purpose of sys,os,getopt module in Python.Explain**

**i) Python's sys module**

This module provides access to some variables used by the interpreter and to functions that interact strongly with the interpreter.

- sys.argv is the list of command-line arguments passed to the Python program.
- **argv contains** all the items that come along via the command-line input,      it's basically an array holding the command-line arguments of the program.
- To use sys.argv, import sys should be used.

- The first argument, sys.argv[0] contains the name of the python program and sys.argv[1]is the next argument passed to the program.
- Main(sys.argv[1])
  - ➢ The input file (c++ file) is send along with its path as a list (array) using argv[1].
  - ➢ Argv[0] contains the python program which is need not to be passed because by default_main_contains source code reference.

## ii) Python's OS Module

The *OS* module in Python provides a way of using operating system dependent functionality.

The functions that the *OS* module allows you to interface with the Windows operating system where Python is running on.

**os.system():** Execute the C++ compiling command in the shell.

For Example to compile C++ program **g++** *compiler* should be invoked.

**Command:**

 **os.system ('g++' + <varaiable_name1> '-<mode>' +      <variable_name2>**

| os.system :- | function system() defined in os module |
|---|---|
| g++ :- | General compiler to compile C++ program under Windows Operating system. |
| variable_name1:- | Name of the C++ file without extension .cpp in string format |
| mode :- | To specify input or output mode. Here it is o prefixed with hyphen. |
| variable_name2 :- | Name of the executable file without extension .exe in string format |

**Example:**

    os.system('g++' + cpp_file + '-o' + exe_file)

    g++ complier complies the file cpp_file and –o (output) send to exe_file.

## iii) Python getopt module

The getopt module of Python helps you to parse (split) command-line options and arguments.

This module provides two functions to enable command-line argument parsing.

**getopt.getopt method**

This method pases command-line options and parameter list.

**Syntax: <opts>,<args>=getopt.getopt(argv, options, [long_options])**

Here is the detail of the parameters

- **argv** − This is the argument list of values to be parsed (splited). In our program the complete command will be passed as a list.
- **options** − This is string of option letters that the Python program recognize as, for input or for output, with options (like 'i' or 'o') that followed by a colon (:). Here colon is used to denote the mode.
- **long_options** −This parameter is passed with a list of strings. Argument of Long options should be followed by an equal sign ('='). In our program

the C++ file name will be passed as string and 'i' also will be passed along with to indicate it as the input file.

- ➢ **getopt() method returns value consisting of two elements**.
- ➢ Each of these values are stored separately in two different list (arrays) **opts** and **args.** Opts contains list of splitted strings like mode, path and args contains any string if at all not splitted because of wrong path or mode.
- ➢ args will be an empty array if there is no error in splitting strings by getopt().

**Example:**

**opts, args = getopt.getopt (argv, "i:",['ifile='])**

| where **opts** contains | [('-i', 'c:\\pyprg\\p4')] |
|---|---|
| -i :- | **option** nothing but **mode should be followed by :** |
| 'c:\\pyprg\\p4' | **value** nothing but the **absolute path of C++ file.** |

- ➢ In our examples since the entire command line commands are parsed and no leftover argument, the **second argument args** will be empty [].
- ➢ If args is displayed using print() command it displays the output as [].

**Example**:

>>>print(args)

[ ]

4. **Write the syntax for getopt() and explain its arguments and return values.**
   **Python getopt module**

   The getopt module of Python helps you to parse (split) command-line options and arguments.

   This module provides two functions to enable command-line argument parsing.
   **getopt.getopt method**

   This method pases command-line options and parameter list.
   **Syntax: <opts>,<args>=getopt.getopt(argv, options, [long_options])**
   Here is the detail of the parameters

   - ▪ **argv** − This is the argument list of values to be parsed (splited). In our program the complete command will be passed as a list.
   - ▪ **options** − This is string of option letters that the Python program recognize as, for input or for output, with options (like 'i' or 'o') that followed by a colon (:). Here colon is used to denote the mode.
   - ▪ **long_options** −This parameter is passed with a list of strings. Argument of Long options should be followed by an equal sign ('='). In our program the C++ file name will be passed as string and 'i' also will be passed along with to indicate it as the input file.

- ➢ **getopt() method returns value consisting of two elements**.
- ➢ Each of these values are stored separately in two different list (arrays) **opts** and **args.** Opts contains list of splitted strings like mode, path and args contains any string if at all not splitted because of wrong path or mode.
- ➢ args will be an empty array if there is no error in splitting strings by getopt().

**Example:**

**opts, args = getopt.getopt (argv, "i:",['ifile='])**

| where **opts** contains | [('-i', 'c:\\pyprg\\p4')] |
|---|---|
| -i :- | **option** nothing but **mode should be followed by :** |
| 'c:\\pyprg\\p4' | **value** nothing but the **absolute path of C++ file.** |

> ➢ In our examples since the entire command line commands are parsed and no leftover argument, the **second argument args** will be empty [].
> ➢ If args is displayed using print() command it displays the output as [].

**Example**:
```
>>>print(args)
[ ]
```

**5. Write a Python program to execute the following c++ coding**
```
#include <iostream>
using namespace std;
int main()
{ cout<<"WELCOME";
return(0);
}
```
**The above C++ program is saved in a file welcome.cpp**
```
import sys, os, getopt
def main(argv):
    opts, args = getopt.getopt(argv, "i:",['ifile='])
    for o, a in opts:
    if o in "-i":
        run(a)
def run(a):
    inp_file=a+'.cpp'
    exe_file=a+'.exe'
    os.system('g++'+ inp_file+'-o'+exe_file)
    print("----------------------")
    print
    os.system(exe_file)
if __name__ =='__main__':
    print
    main(sys.argv[1:])
```
**STEPS TO IMPORT CPP CODE INTO PYTHON CODE:**
> ➢ Select **File→New** in Notepad and type the Python program
> ➢ Save the File as **welcome.py.**
> ➢ Click the Run Terminal and open the command window.
> ➢ Go to the folder of Python using cd command.
> ➢ Type the command:**Python c:\pyprg\welcome.py –i c:pyprg\welcome_cpp**

**Output:**
```
-----------------------
    WELCOME
-----------------------
```

# CHAPTER 15
# DATA MANIPULATION THROUGH SQL

## Part - I
**Choose the best answer**                                            **(1 Marks)**

1. Which of the following is an organized collection of data?
   **(A) Database**        (B) DBMS        (C) Information        (D) Records

2. SQLite falls under which database system?
   (A) Flat file database system        **(B) Relational Database system**
   (C) Hierarchical database system        (D) Object oriented Database system

3. Which of the following is a control structure used to traverse and fetch the records of the database?
   (A) Pointer        (B) Key        **(C) Cursor**        (D) Insertion point

4. Any changes made in the values of the record should be saved by the command
   (A) Save        (B) Save As        **(C) Commit**        (D) Oblige

5. Which of the following executes the SQL command to perform some action?
   **(A) Execute()**        (B) Key()        (C) Cursor()        (D) run()

6. Which of the following function retrieves the average of a selected column of rows in a table?
   (A) Add()        (B) SUM()        **(C) AVG()**        (D) AVERAGE()

7. The function that returns the largest value of the selected column is
   **(A) MAX()**        (B) LARGE()        (C) HIGH()    (D) MAXIMUM()

8. Which of the following is called the master table?
   **(A) sqlite_master**        (B) sql_master        (C) main_master        (D) master_main

9. The most commonly used statement in SQL is
   (A) cursor        **(B) select**        (C) execute        (D) commit

10. Which of the following clause avoide the duplicate?
    **(A) Distinct**        (B) Remove        (C) Where        (D) GroupBy

## Part - II
**Answer the following questions**                                   **(2 Marks)**

1. **Mention the users who use the Database.**
   ➢ Users of database can be human users, other program or applications.

2. **Which method is used to connect a database? Give an example.**
   ➢ Create a connection using connect () method and pass the name of the database File
   ➢ **Example:**
   ```
   import sqlite3
   # connecting to the database
   connection = sqlite3.connect ("Academy.db")
   # cursor
   cursor = connection.cursor()
   ```

3. **What is the advantage of declaring a column as "INTEGER PRIMARY KEY"**
   ➢ If a column of a table is declared to be an INTEGER PRIMARY KEY, then whenever a NULL will be used as an input for this column, the NULL will be automatically converted into an integer which will one larger than the highest value so far used in that column.
   ➢ If the table is empty, the value 1 will be used.

4. **Write the command to populate record in a table. Give an example.**
➢ To populate (add record) the table "INSERT" command is passed to SQLite. "execute" method executes the SQL command to perform some action.
➢ **Example:**
        sql_command = """INSERT INTO Student (Rollno, Sname, Grade, gender, Average, birth_date)     VALUES (NULL, "Akshay", "B", "M","87.8", "2001-12-12");"""
            cursor.execute(sql_ command)

5. **Which method is used to fetch all rows from the database table?**
➢ The fetchall() method is used to fetch all rows from the database table.
➢ **Example:**
        result = cursor.fetchall()

## Part - III

**Answer the following questions**                        **(3 Marks)**

1. **What is SQLite?What is it advantage?**
➢ SQLite is a simple relational database system, which saves its data in regular data fi les or even in the internal memory of the computer.
   **Advantages:**
➢ SQLite is fast, rigorously tested, and flexible, making it easier to work.
➢ Python has a native library for SQLite.

2. **Mention the difference between fetchone() and fetchmany()**

| fetchone() | fetchmany() |
|---|---|
| ➢ The fetchone() method returns the next row of a query result set or None in case there is no row left. | ➢ The fetchmany() method returns the next number of rows (n) of the result set. |
| ➢ Using while loop and fetchone() method we can display all the records from a table. | ➢ Displaying specified number of records is done by using fetchmany(). |

3. **What is the use of Where Clause.Give a python statement Using the where clause.**
➢ The WHERE clause is used to extract only those records that fulfill a specified condition.
➢ **Example:** To display the different grades scored by male students from "student table"
        import sqlite3
        connection = sqlite3.connect("Academy.db")
        cursor = connection.cursor()
        cursor.execute("SELECT DISTINCT (Grade) FROM student where gender='M'")
        result = cursor.fetchall()
        print(*result,sep="\n")
   **OUTPUT**
        ('B',)
        ('A',)
        ('C',)
        ('D',)

4. **Read the following details.Based on that write a python script to display department wise records**

              **database name**                 **:- organization.db**

              **Table name**                      **:- Employee**

              **Columns in the table**    **:- Eno, EmpName, Esal, Dept**

## PYTHON SCRIPT:

```
import sqlite3
connection = sqlite3.connect("organization.db")
c=conn.execute("SELECT * FROM Employee GROUP BY Dept")
for row in c:
        print(row)
conn.close()
```

5. **Read the following details.Based on that write a python script to display records in desending order of Eno.**

    **database name**                     **:- organization.db**

    **Table name**                           **:- Employee**

    **Columns in the table**         **:- Eno, EmpName, Esal, Dept**

## PYTHON SCRIPT:

```
import sqlite3
connection = sqlite3.connect("organization.db")
cursor=connection.cursor()
cursor.execute("SELECT * FROM Employee ORDER BY Eno DESC")
result=ccursor.fetchall()
print(result)
```

## Part - IV

**Answer the following questions**                             **(5 Marks)**

1. **Write in brief about SQLite and the steps used to use it.**

➢ SQLite is a simple relational database system, which saves its data in regular data files or even in the internal memory of the computer.

➢ It is designed to be embedded in applications, instead of using a separate database server program such as MySQL or Oracle.

   **ADVANTAGES:**

➢ SQLite is fast, rigorously tested, and flexible, making it easier to work.

➢ Python has a native library for SQLite.

   **STEPS TO USE SQLITE:**

   **Step 1: import sqlite3**

   **Step 2: Create a connection using connect () method and pass the name of the database File.**

➢ Connecting to a database in step2 means passing the name of the database to be accessed.

➢ If the database already exists the connection will open the same.

➢ Otherwise, Python will open a new database file with the specified name.

   **Step 3: Set the cursor object cursor = connection.cursor()**

➢ Cursor in step 3: is a control structure used to traverse and fetch the records of the database.

- Cursor has a major role in working with Python. All the commands will be executed using cursor object only.
- To create a table in the database, create an object and write the SQL command in it.

**Example:-** sql_comm = "SQL statement"

- For executing the command use the cursor method and pass the required sql command as a parameter.
- Many numbers of commands can be stored in the sql_comm and can be executed one after other.
- Any changes made in the values of the record should be saved by the commend "Commit" before closing the "Table connection".

2. **Write the Python script to display all the records of the following table using fetchmany()**

| Icode | ItemName | Rate |
|-------|----------|-------|
| 1003 | Scanner | 10500 |
| 1004 | Speaker | 3000 |
| 1005 | Printer | 8000 |
| 1008 | Monitor | 15000 |
| 1010 | Mouse | 700 |

**PYTHON SCRIPT:**
```
import sqlite3
connection= sqlite3.connect('Materials.db')
cursor=connection.cursor()
cursor.execute("SELECT* FROM Materials")
print("Displaying All The Records")
result=cursor.fetchmany(5)
print(result, Sep="\n")
```

**OUTPUT:**

**Displaying All The Records**
**(1003, 'Scanner', 10500)**
**(1004, 'Speaker', 3000)**
**(1005, 'Printer', 8000)**
**(1008, 'Monitor', 15000)**
**(1010, 'Mouse', 700)**

3. **What is the use of HAVING clause. Give an example python script.**
- Having clause is used to filter data based on the group functions.
- This is similar to WHERE condition but can be used only with group functions.
- Group functions cannot be used in WHERE Clause but can be used in HAVING clause.

**Example**
```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT GENDER,COUNT(GENDER) FROM Student
        GROUP BY GENDER HAVING COUNT(GENDER)>3")
result = cursor.fetchall()
```

86

```
co = [i[0] for i in cursor.description]
print(co)
print(result)
```
**OUTPUT**
```
['gender', 'COUNT(GENDER)']
[('M', 5)]
```

4. **Write a Python script to create a table called ITEM with following specification. Add one record to the table.**

**Name of the database :- ABC**

**Name of the table :- Item**

**Column name and specification :-**

| Icode | :- | integer and act as primary key |
|---|---|---|
| Item Name | :- | Character with length 25 |
| Rate | :- | Integer |
| Record to be added | :- | 1008, Monitor,15000 |

**PYTHON SCRIPT:**
```
import sqlite3
connection=sqlite3.connect("ABC.db")
cursor=connection.cursor()
sql_command_"""CREATE TABLE Item(Icode INTEGER PRIMARY
KEY,ItemName VARCHAR(25),Rate INTEGRE);"""
cursor.execute(sql_command)
connection.commit()
connection.close()
print("TABLE CREATED")
```
**OUTPUT:**
```
TABLE CREATED
```

5. **Consider the following table Supplier and item .Write a python script for (i) to (ii)**

| SUPPLIER | | | | |
|---|---|---|---|---|
| **Suppno** | **Name** | **City** | **Icode** | **SuppQty** |
| S001 | Prasad | Delhi | 1008 | 100 |
| S002 | Anu | Bangalore | 1010 | 200 |
| S003 | Shahid | Bangalore | 1008 | 175 |
| S004 | Akila | Hydrabad | 1005 | 195 |
| S005 | Girish | Hydrabad | 1003 | 25 |
| S006 | Shylaja | Chennai | 1008 | 180 |
| S007 | Lavanya | Mumbai | 1005 | 325 |

i. **Display Name, City and Itemname of suppliers who do not reside in Delhi.**
ii. **Increment the SuppQty of Akila by 40**

**PYTHON SCRIPT:**

i. **Display Name, City and Itemname of suppliers who do not reside in Delhi.**

87

```
import sqlite3
connection=sqlite3.connect("ABC.db")
cursor.execute("SELECT Supplier.Name, Supplier.City, Item.ItemName FROM
Supplier,Item WHERE Supplier.Icode=Item.Icode AND Supplier.City NOT In
Delhi")
s=[i[0] for I in cursor.description]
    print(s)
result=cursor.fetchhall()
for r in result:
    print r
```

**OUTPUT:**
**['Name',     'City',     'ItemName']**
**['Anu',      'Bangalore', 'Scanner']**
**['Shahid',   'Bangalore', 'Speaker']**
**['Akila',    'Hydrabad', 'Printer']**
**['Girish',   'Hydrabad', 'Monitor']**
**['Shylaja',  'Chennai', 'Mouse']**
**['Lavanya', 'Mumbai',  'CPU']**

  ii.  **Increment the SuppQty of Akila by 40**
```
import sqlite3
connection=sqlite3.connect("ABC.db")
cursor.execute("UPDATE     Supplier     ST     SuppQty=SuppQty+40     WHERE
Name='Akila'")
cursor.commit()
result=cursor.fetchall()
print(result)
connection.close()
```

# CHAPTER 16
## DATA VISUALIZATION USING PYPLOT:
## LINE CHART, PIE CHART AND BAR CHART
### Part - I

**Choose the best answer** (1 Marks)

1. Which is a python package used for 2D graphics?
   **a. matplotlib.pyplot**    b. matplotlib.pip   c. matplotlib.numpy    d. matplotlib.plt

2. Identify the package manager for Python packages, or modules.
   a. Matplotlib    **b. PIP**    c. plt.show()    d. python package

3. Read the following code: Identify the purpose of this code and choose the right option from the following.

C:\Users\YourName\AppData\Local\Programs\Python\Python36-32\Scripts>pip – version
   a. Check if PIP is Installed    b. Install PIP    c. Download a Package  **d. Check PIP version**

4. Read the following code: Identify the purpose of this code and choose the right option from the following.

C:\Users\Your Name\AppData\Local\Programs\Python\Python36-32\Scripts>pip list
   **a. List installed packages**    b. list command
        c.Install PIP    d. packages installed

5. To install matplotlib, the following function will be typed in your command prompt. What does "-U"represents?

Python –m pip install –U pip
   a. downloading pip to the latest version
   **b. upgrading pip to the latest version**
   c. removing pip
   d. upgrading matplotlib to the latest version

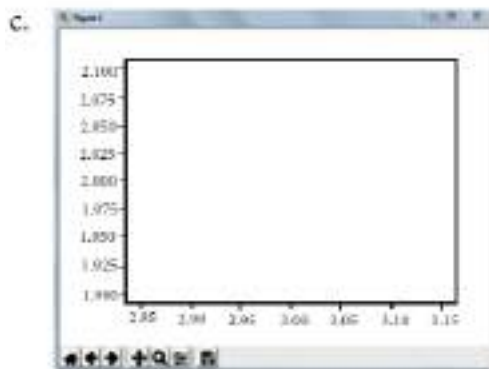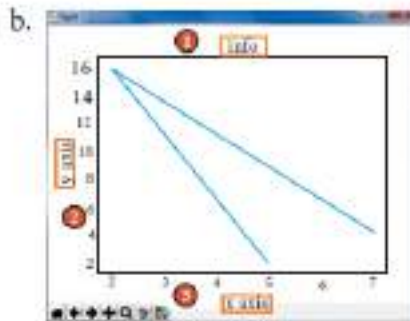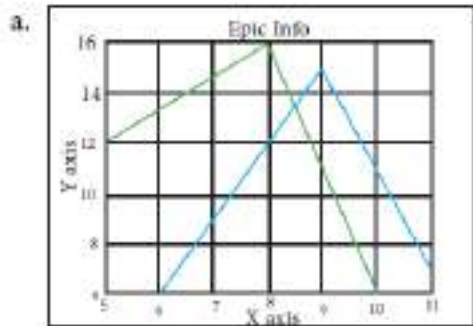6. Observe the output figure. Identify the coding for obtaining this output.



**a.import matplotlib.pyplot as plt**
**plt.plot([1,2,3],[4,5,1])**
**plt.show()**
b. import matplotlib.pyplot as plt
plt.plot([1,2],[4,5])
plt.show()
c. import matplotlib.pyplot as plt
plt.plot([2,3],[5,1])
plt.show()
d. import matplotlib.pyplot as plt

```
plt.plot([1,3],[4,1])
plt.show()
```
7.  Read the code:
a. import matplotlib.pyplot as plt
b. plt.plot(3,2)
c. plt.show()
Identify the output for the above coding.



**Ans: option c**
   8.   Which key is used to run the module?
   a. F6              b. F4           c. F3        **d. F5**
   9. Identify the right type of chart using the following hints.
Hint 1: This chart is often used to visualize a trend in data over intervals of time.
Hint 2: The line in this type of chart is often drawn chronologically.
   **a. Line chart**          b. Bar chart          c. Pie chart          d. Scatter plot
 10.    Read the statements given below. Identify the right option from the following for pie
        chart.
Statement A: To make a pie chart with Matplotlib, we can use the plt.pie() function.
Statement B: The autopct parameter allows us to display the percentage value using the
     Python string formatting.
        a. Statement A is correct                b. Statement B is correct
        **c. Both the statements are correct**    d. Both the statements are wrong
**Part - II**
**Answer the following questions**                                              **(2 Marks)**
   **1.  Define: Data Visualization.**
   ➢ Data Visualization is the graphical representation of information and data.
   ➢ The objective of Data Visualization is to communicate information visually to users.
   **2.  List the general types of data visualization.**
   ➢ Charts

- ➢ Tables
- ➢ Graphs
- ➢ Maps
- ➢ Infographics
- ➢ Dashboards

**3. List the types of Visualizations in Matplotlib.**
- ➢ Line plot
- ➢ Scatter plot
- ➢ Histogram
- ➢ Box plot
- ➢ Bar chart and
- ➢ Pie chart

**4. How will you install Matplotlib?**
- ➢ **Matplotlib** can be installed using pip software.
- ➢ Pip is a management software for installing python packages.
- ➢ importing Matplotlib using the command: **import matplotlib.pyplot as plt**
- ➢ MAtplotlib can be imported in the workplace.

**5. Write the difference between the following functions: plt.plot([1,2,3,4]), plt. plot([1,2,3,4], [1,4,9,16]).**

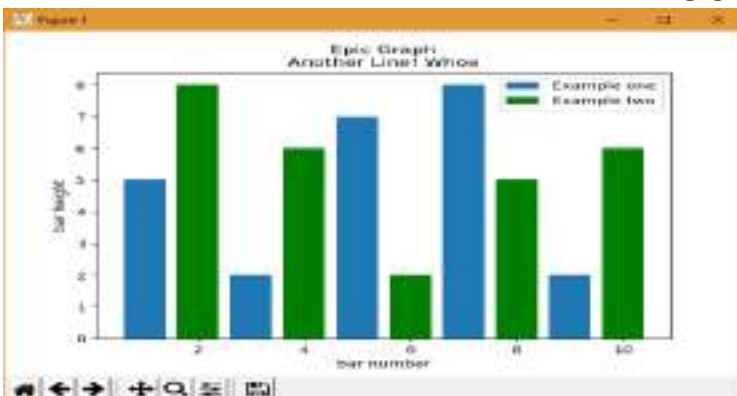| plt.plot([1,2,3,4]) | plt. plot([1,2,3,4], [1,4,9,16]) |
|---|---|
| It refers value as [1,2,3,4] | It refers x and y values as ([1,2,3,4],[1,4,9,16]) |
| Indirectly it refers x values as [0,1,2,3] (0,1) (1,1) (2,3) (3,4) | Directly x and y values are given as (1,1) (2,4) (3,9) (4,16) |

## Part - III

**Answer the following questions** **(3 Marks)**

**1. Draw the output for the following data visualization plot.**

```
import matplotlib.pyplot as plt
plt.bar([1,3,5,7,9],[5,2,7,8,2], label="Example one")
plt.bar([2,4,6,8,10],[8,6,2,5,6], label="Example two", color='g')
plt.legend()
plt.xlabel('bar number')
plt.ylabel('bar height')
plt.title('Epic Graph\nAnother Line! Whoa')
plt.show()
```

**OUTPUT**



91

**2. Write any three uses of data visualization.**
  - Data Visualization help users to analyze and interpret the data easily.
  - It makes complex data understandable and usable.
  - Various Charts in Data Visualization helps to show relationship in the data for one or more variables.

**3. Write the coding for the following:**
**a. To check if PIP is Installed in your PC.**
  - In command prompt type pip-version
  - If it is installed already, you will get the version.
  - **Command:** Python – m pip install – U pip

**b. To Check the version of PIP installed in your PC.**
  - C:\Users\YourName\AppData\Local\Programs\Python\Python36-32\Scripts>pip-version

**c. To list the packages in matplotlib.**
  - C:\Users\YourName\AppData\Local\Programs\Python\Python36-32\Scripts>pip list

**4. Write the plot for the following pie chart output.**



```
import matplotlib.pyplot as plt
slices = [7,2,2,13]
activities = ['sleeping', 'eating',' working', 'playing']
plt.pie(slices, labels=activites, atopct = 'y.1.1 f%%')
plt.title('Interesting Graph Check It Out')
plt.show()
```

**Part - IV**

**Answer the following questions**                                **(5 Marks)**

**1. Explain in detail the types of pyplots using Matplotlib.**
**Line Chart**
  - A Line Chart or Line Graph is a type of chart which displays information as a series of data points called 'markers' connected by straight line segments.
  - A Line Chart is often used to visualize a trend in data over intervals of time – a time series – thus the line is often drawn chronologically.

**Example: Line plot**
```
import matplotlib.pyplot as plt
years = [2014, 2015, 2016, 2017, 2018]
total_populations = [8939007, 8954518, 8960387, 8956741, 8943721]
plt.plot (years, total_populations)
plt.title ("Year vs Population in India")
```

92

```
plt.xlabel ("Year")
plt.ylabel ("Total Population")
plt.show()
```
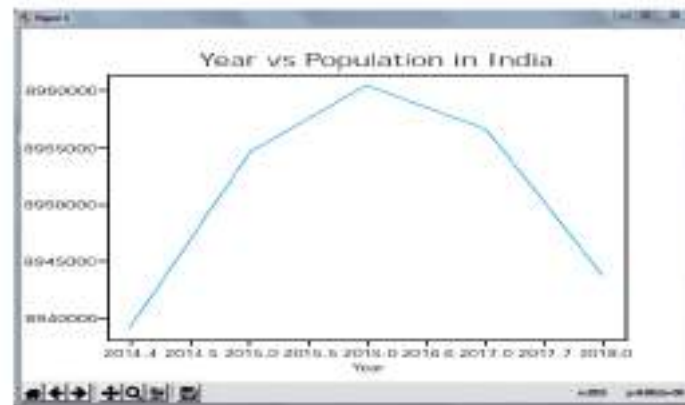In this program,

Plt.title() → specifies title to the graph

Plt.xlabel() → specifies label for X-axis

Plt.ylabel() → specifies label for Y-axis

**Output**



### Bar Chart

➢ A BarPlot (or BarChart) is one of the most common type of plot.

➢ It shows the relationship between a numerical variable and a categorical variable.

➢ Bar chart represents categorical data with rectangular bars.

➢ Each bar has a height corresponds to the value it represents.

➢ The bars can be plotted vertically or horizontally.

➢ It's useful when we want to compare a given numeric value on different categories.

➢ To make a bar chart with Matplotlib, we can use the plt.bar() function.

**Example**

```
import matplotlib.pyplot as plt
# Our data
labels = ["TAMIL", "ENGLISH", "MATHS", "PHYSICS", "CHEMISTRY", "CS"]
usage = [79.8, 67.3, 77.8, 68.4, 70.2, 88.5]
# Generating the y positions. Later, we'll use them to replace them with labels.
y_positions = range (len(labels))
# Creating our bar plot
plt.bar (y_positions, usage)
plt.xticks (y_positions, labels)
plt.ylabel ("RANGE")
plt.title ("MARKS")
plt.show()
```
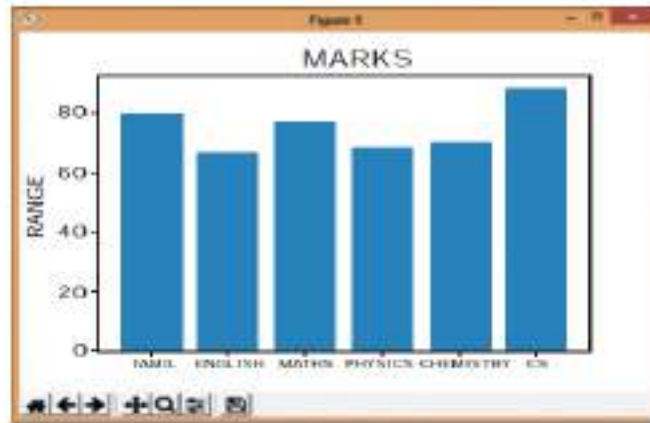**Output**

Labels → Specifies labels for the bars.

Usgae → Assign values to the labels specified.

Xticks → Display the tick marks along the x-axis at the values represented. Then specify the label for each tick mark.

Range → Create sequence of numbers.

**Pie Chart**

➢ Pie Chart is probably one of the most common type of chart.

➢ It is a circular graphic which is divided into slices to illustrate numerical proportion.

➢ The point of a pie chart is to show the relationship of parts out of a whole.

➢ To make a Pie Chart with Matplotlib, we can use the ***plt.pie()*** function.

➢ The autopct parameter allows us to display the percentage value using the Python string formatting.

**Example**

```
import matplotlib.pyplot as plt
sizes = [89, 80, 90, 100, 75]
labels = ["Tamil", "English", "Maths", "Science", "Social"]
plt.pie (sizes, labels = labels, autopct = "%.2f ")
plt.axes().set_aspect ("equal")
plt.show()
```

**Output**



2. **Explain the various buttons in a matplotlib window.**

**Home Button**

➢ The Home Button will help once you have begun navigating your chart.

➢ If you ever want to return back to the original view, you can click on this.

**Forward/Back buttons**

➢ These buttons can be used like the Forward and Back buttons in your browser.

➢ You can click these to move back to the previous point you were at, or forward again.

94

**Pan Axis**

This cross-looking button allows you to click it, and then click and drag your graph around.

**Zoom**

➢ The Zoom button lets you click on it, then click and drag a square that you would like to zoom into specifically.

➢ Zooming in will require a left click and drag.

➢ You can alternatively zoom out with a right click and drag.

**Configure Subplots**

➢ This button allows you to configure various spacing options with your figure and plot.

**Save Figure**

➢ This button will allow you to save your in various forms.

3. **Explain the purpose of the following functions:**

**a. plt.xlabel**

plt.xlabel() -> specifies label for X- axis

**b. plt.ylabel**

plt.ylabel() -> specifies label for Y- axis

**c. plt.title**

plt.title()-> specifies title to the graph

**d. plt.legend()**

Calling legend() with no arguments automatically fetches the legend handles and their associated labels.

**e. plt.show()**

Display a figure. When running in Python with its Pylab mode, display all figures and return to the Python prompt.