Chapter - 1 : Function
Part - II

Answer the following questions

1. What is subroutine?

- Subroutines are the basic building blocks of computer programs.
- 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 the 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 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 sum x y:
     return x+y
ii) let disp:
      print 'welcome'
```

```
iii) let rec sum num:
       if (num!=0) then
         return num+sum(num-1)
       else
         return num
```

i) Normal Function ii) Normal Function iii) Recursive Function

Part - III

Answer the following questions

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 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.

- When a function depends on variables or functions outside of its definition block, you can never be sure that the function will behave the same every time it is called. Example: random() function

4. Differentiate pure and impure function.

Pure Function	Impure Function
With the same set of arguments will return the same values.	With the same set of arguments might get the different return values.
They do not have any side effects.	They cause side effects.
They do not modify the arguments which are passed to them	They may modify the arguments which are passed
Example: strlen()	Example: random()

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

- One of the most popular side effects is modifying the variable outside of function.
- Example

```
y:=0
let inc (x:int):int:=
   y:=y+x
   return (y)
```

- The result of inc() will change every time if the value of y get changed inside the function definition.
- The side effect of the inc() function is changing the data of the external variable y.

Part - IV

Answer the following questions

1. What are called parameters and write a note on

i) Parameter without Type

ii)Parameter with Type

Parameters

- Parameters are the variables in a function definition

i) Parameter without Type

- Let's see an example of a function defintion 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 passed to the variable 'b' is the argument.
- The precondition (requires) and postcondition (returns) of the function is given.
- We have not mentioned any datatype.
- In the above function definition the expression return int value and implicitly it means that the entire expression has type int.

ii) Parameter with Type

- Lets us write the same function defintion with types,

```
(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 a (b-1)
```

- When we write the type annotations for 'a' and 'b' the parenthesis are mandatory.
- There are times we may want to explicitly write down types.
- This is useful when you get a type error from the compiler that doesn't make sense.
- Explicitly annotating the types can help with debugging such an error message.

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
ii) Identify the argument which tells it is a recursive function
iii) Name of the argument variable
iv) Statement which invoke the function recursively
v) Statement which terminates the recursion

i) gcd
ii) let rec gcd a b :=
iii) a b
iv) if b<>0 then gcd b (a mod b)
v) return a

3. Explain with example Pure and Impure functions.

Pure Functions

- Pure functions are function which will give exact result when the same arguments are passed.
- A function can be a pure function provided that it should not have any external variable which will alter the behaviour of that variable.

```
let square x
  return: x * x
```

- The above function square is a pure function because it will not give different results for same input.
- Advantage is that if a function is pure, then if it is called several times with the same arguments, the compiler only needs to actually call the function once.

Impure Functions

- The variables used inside the function may cause side effects though the function which are not passed with any arguments is called impure function.
- When a function depends on variables or functions outside of its definition block, you can never be sure that the function will behave the same every time it's called.
- For example,
- random() function will give different outputs for the same function call.

```
let randomnumber:=
    a:=random()
    if a>10 then
       return: a
    else
       return: 10
```

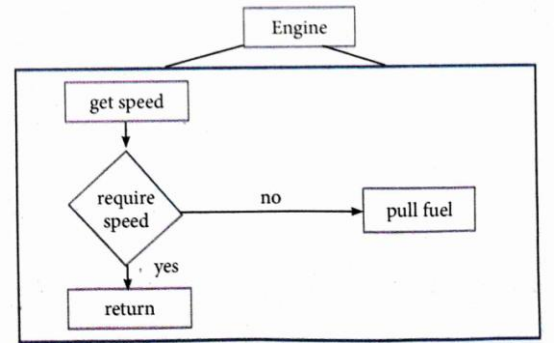
4. Explain with an example interface and implementation.

Interface

- An interface is a set of action that an object can do.
- In object oriented programming language, an interface is a description of all functions that a class must have in order to be a new interface.
- A class declaration combines the external interface (its local state) with an implementation of that interface. An object is an instance created from the class.
- The interface defines an object's visibility to the outside world.
- For example,
- Anything that "ACTS LIKE" a light, should have function definitions like turn_on() and a turn_off().

Implementation

- Implementation carries out the instructions defined in the interface.
- In object oriented programs classes are the interface and how the object is processed and executed is the implementation.



- 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 and the engine.
- In this case, the function call would be speed(70), this is the interface.
- Internally, the engine is doing all the things. fuel, air, pressure and electricity come together to create the power to move the vehicle.
- All of these actions are separated from the driver. Thus we separated interface from the implementation.

Chapter - 2 : Data Abstraction

Part - II

Answer the following questions

1. What is abstract data type?

- Abstract Data type (ADT) is a type (or class) for objects whose behaviour 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.

3. What is a Pair? Give an example.

- Any way of bundling two values together into one can be considered as a pair.
- Example: lst[(0, 10), (1, 20)]

4. What is a List? Give an example.

- List is constructed by placing expressions within square brackets separated by commas.
- List can store multiple values.
- Example: lst:=[10,20]
  x,y=lst
- In the above example x will become 10 and y will become 20.

5. What is a Tuple? Give an example.

- A tuple is a comma-separated sequence of values surrounded with parentheses
- Example: colour= ('red', 'blue', 'Green')

Part - III

Answer the following questions

1. Differentiate Concrete data type and abstract datatype.

Concrete data type	Abstract data type.
A concrete data type is a data type whose representation is known	In abstract data type the representation of a data type is unknown.
Direct implementations of a relatively simple concept.	A high level view of a concept independent of its implementation.

2. Which strategy is used for program designing? Define that Strategy.

- Wishful Thinking strategy is used for program designing
- 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) NI:=number() b) accetnum(nl) c) displaynum(nl) d) eval(a/b)
e) x,y: makeslope (m), makeslope(n) f) display()

a) Constructor
b) Selector
c) Selector
d) Selector
e) Constructor
f) Selector

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.
  - Multiple assignment
  - Element selection operator

Multiple assignment

- Multiple assignment, which unpacks a list into its elements and binds each element to a different name.

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

- In the above example x will become10 and y will become 20.

Element selection operator

- Element selection operator, also 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
```

5. Identify Which of the following are List, Tuple and class?

(a) arr [1, 2, 34] **(b) arr (1, 2, 34)**
(c) student [rno, name, mark] **(d) day:= ('sun, 'mon, 'tue, 'wed')**
(e) x [2, 5, 6.5, [5, 6], 8.2] **(f) employee [eno, ename, esal, eaddress]**

a) List
b) Tuple
c) Class
d) Tuple
e) List
f) Class

Part - IV

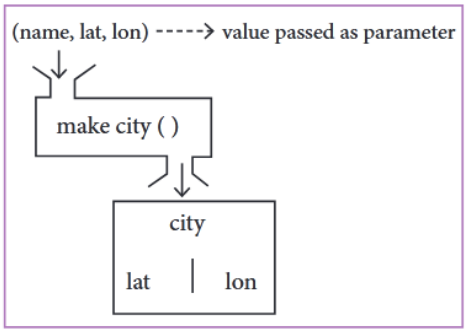
Answer the following questions

1. How will you facilitate data abstraction. Explain it with suitable example

- Data abstraction will be facilitate using Constructor and Selector

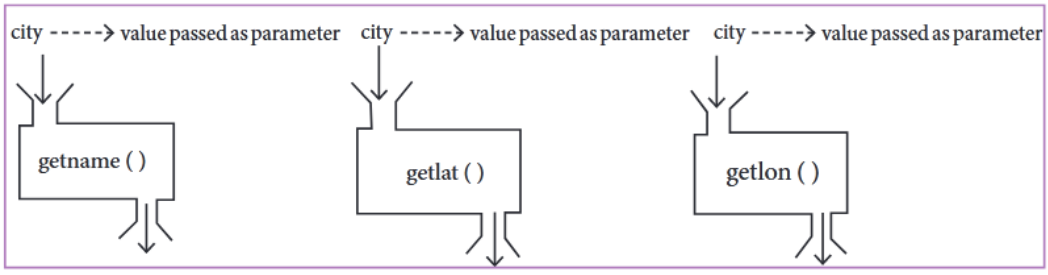
Constructors

- Constructors are functions that build the abstract data type.
- 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.



Selectors

- Selectors are functions that retrieve information from the data type.
- To extract the information of a city object, you would use functions like
  getname(city)
  getlat(city)
  getlon(city)
- The above code 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.
- List can store multiple values.
- Example: lst:=[10,20]
  x,y=lst

- In the above example x will become 10 and y will become 20.
- The elements of a list can be accessed in two ways.
    - Multiple assignment
    - Element selection operator

Multiple assignment

- Multiple assignment, which unpacks a list into its elements and binds each element to a different name.

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

- In the above example x will become10 and y will become 20.

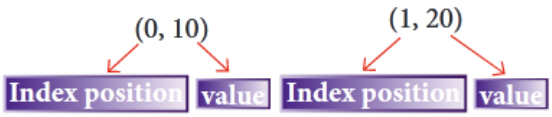
Element selection operator

- Element selection operator, also 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]	lst[1]
	10	20

- In both the example mentioned above mathematically we can represent list similar to a set.

lst[(0, 10), (1, 20)] - where



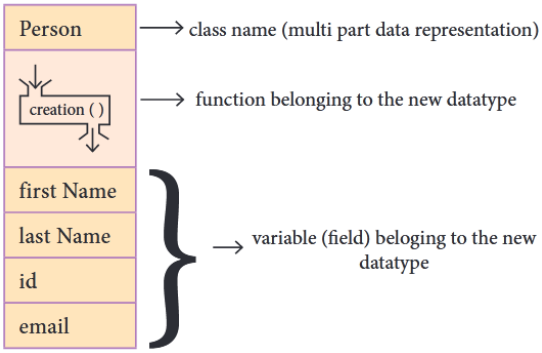
- Any way of bundling two values together into one can be considered as a pair.
- Lists are a common method to do so.
- Therefore List can be called as Pairs.

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.
- Consider the following pseudo code:

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

- The new data type Person is pictorially represented as



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.
- Its definition adds a new data type, in this case a type named Person.
- Once defined, we can create new variables (instances) of the type.
- In this example, Person is referred to as a class or a type, while p1 is referred to as an object or an instance.

Chapter - 3 : Scoping
Part - II

Answer the following questions

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.

3. What is Mapping?

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

4. What do you mean by Namespaces?

- Namespaces are containers for mapping names of variables to objects.

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

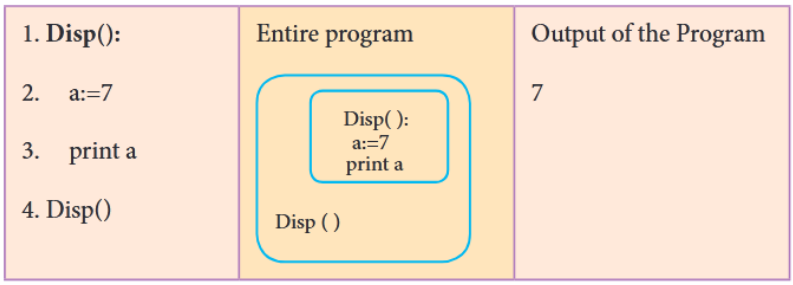
- Python prescribes a convention of prefixing the name of the variable or method with single or double underscore to emulate the behaviour of protected and private access specifiers.

Part - III

Answer the following questions

1. Define Local scope with an example.

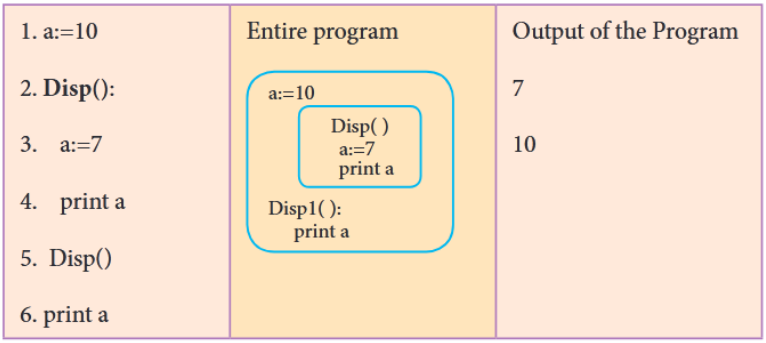
- Local scope refers to variables defined in current function. Always, a function will first look up for a variable name in its local scope. Only if it does not find it there, the outer scopes are checked.



- 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. This means, global variable can be accessed inside or outside of all the functions in a program.

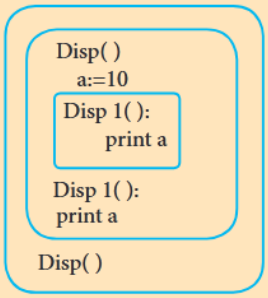


- On execution of the above code the variable a which is defined inside the function displays the value 7 for the function call Disp() and then it displays 10, because a is defined in global scope.

3. Define Enclosed scope with an example.

- All programming languages permit functions to be nested. A function (method) with in another function is called nested function.
- 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.

1. Disp(): 2. a:=10 3. Disp1(): 4. print a 5. Disp1() 6. print a 7. Disp()	Entire program 	Output of the Program 10 10

- 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.

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

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

Output

Red Blue Green
Red Blue
Red

Identify the scope of the variable

color:='Red' Global scope
b:='Blue' Enclosed scope
g:='Green' Local scope

Part - IV

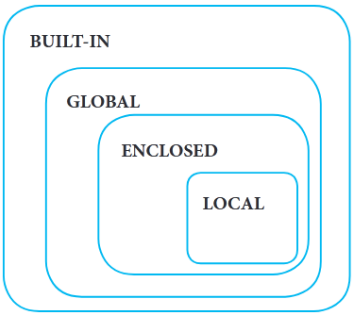
Answer the following questions

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

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).

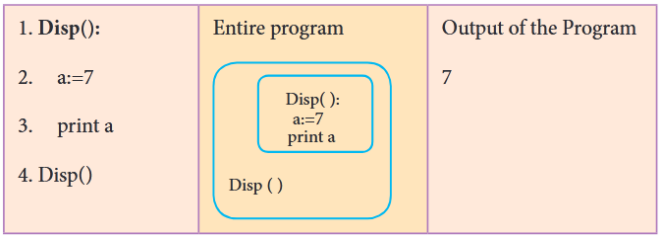
Local(L)	Defined inside function/class
Enclosed(E)	Defined inside enclosing functions (Nested function concept)
Global(G)	Defined at the uppermost level
Built-in (B)	Reserved names in built-in functions (modules)



<u>Types of scopes for variable</u>

Local Scope

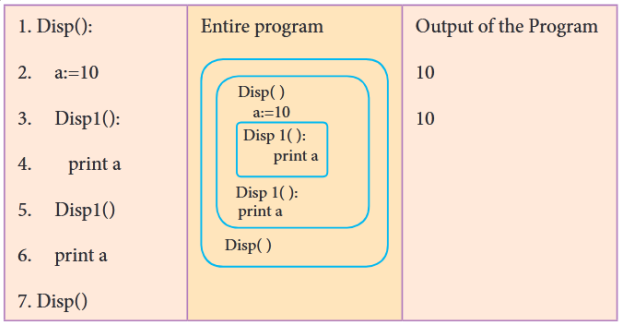
- Local scope refers to variables defined in current function. Always, a function will first look up for a variable name in its local scope. Only if it does not find it there, the outer scopes are checked.



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

Enclosed Scope

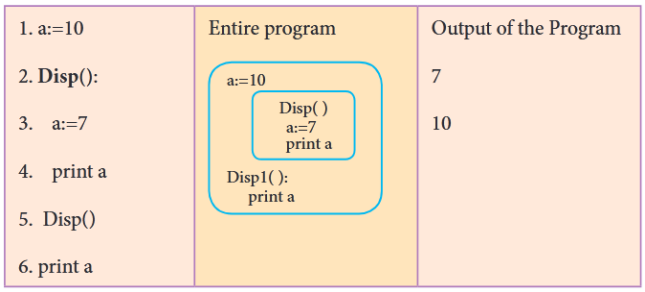
- All programming languages permit functions to be nested. A function (method) with in another function is called nested function.
- A variable which is declared inside a function which contains another function definition with in it, the inner function can also access the variable of the outer function.
- 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.



- 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()

Global Scope

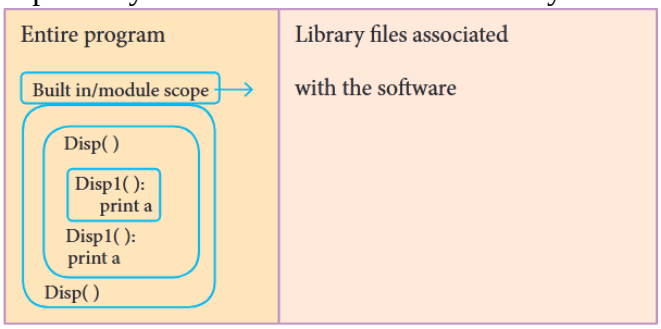
- A variable which is declared outside of all the functions in a program is known as global variable. This means, global variable can be accessed inside or outside of all the functions in a program.



- 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.

Built-in Scope

- Any variable or module which is defined in the library functions of a programming language has Built-in or module scope. They are loaded as soon as the library files are imported to the program



2. Write any Five Characteristics of Modules.

i) Modules contain instructions, processing logic, and data.
ii) Modules can be separately compiled and stored in a library.

iii) Modules can be included in a program.
iv) Module segments can be used by invoking a name and some parameters.
v) 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, 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.

Chapter - 4: Algorithmic Strategies

Part - II

Answer the following questions

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. Write the phases of performance evaluation of an algorithm.

- Performance evaluation can be divided into two different phases:
  - A Priori estimates: This is a theoretical performance analysis of an algorithm.
  - A Posteriori testing: This is called performance measurement.

3. What is Insertion sort?

- Insertion sort is a simple sorting algorithm that builds the final sorted array or list one item at a time. It always maintains a sorted sublist in the lower positions of the list.

4. What is Sorting?

- Sorting is the process of arranging items in a certain order using the methods such as bubble sort, insertion sort, selection sort, etc.

5. What is searching? Write its types.

- Searching is the process to locate specific data among a collection of data.
- **Types of Searching**
  - Linear Search
  - Binary Search

Part - III

Answer the following questions

1. List the characteristics of an algorithm.

- Input
- Output
- Finiteness
- Definiteness
- Effectiveness
- Correctness
- Simplicity
- Unambiguous
- Feasibility
- Portable
- Independent

2. Discuss about Algorithmic complexity and its types.

- 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.

Time Complexity

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

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 following two components:
  - Fixed part
  - Variable part

3. What are the factors that influence time and space complexity.

Time Factor

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

Space Factor

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

4. Write a note on Asymptotic notation.

- 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 Ω is used to describe the best-case of an algorithm.

(iii) Big Θ

- When an algorithm has a complexity with lower bound = upper bound, 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).
- The running time of that algorithm always falls in n log n in the best-case and worst-case.

5. What do you understand by Dynamic programming?

- Dynamic programming is an algorithmic design method that can be 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.
- Dynamic programming is used whenever problems can be divided into similar sub-problems.
- Dynamic programming approaches are used to find the solution in optimized way.

Part - IV

Answer the following questions

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.
Effectiveness	Every instruction must be carried out effectively.
Correctness.	The algorithms should be error free
Simplicity	Easy to implement.
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 Search

- 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.

Procedure

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

3. What is Binary search? Discuss with example.

Binary search

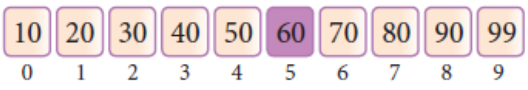
- Binary search also called half-interval search algorithm. It finds the position of a search element within a sorted array.
- The binary search algorithm can be done as divide-and-conquer search algorithm and executes in logarithmic time.

Procedure for Binary search

1. 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.
2. When a match is found, display success message with the index of the element matched.
3. If no match is found for all comparisons, then display unsuccessful message.

Binary Search Working principles

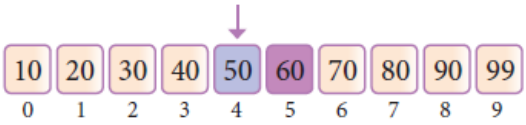
- List of elements in an array must be sorted first for Binary search.
- The array is being sorted so it enables to do the binary search algorithm.
- Let us assume that the search element is 60 and we need to search the location or index of search element 60 using binary search.



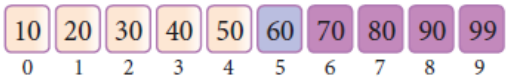
- First, we find index of middle element of the array by using this formula :

mid = low + (high - low) / 2

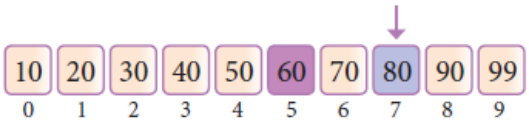
- Here it is, 0 + (9 - 0) / 2 = 4. So, 4 is the mid value of the array.



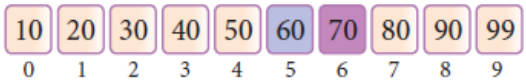
- Compare the value stored at location or index 4 is 50, which is not match with search element. As the search value 60 > 50.



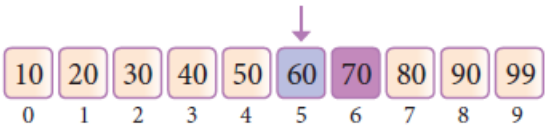
- Now we change **low = mid + 1** and find the new mid value again.
- low = 4 + 1 = 5.
- mid = 5 + (9 - 5) / 2 = 7. Now mid is 7.
- We compare the value stored at location 7 with our target value 60.



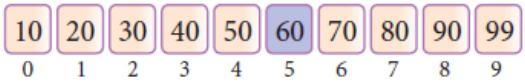
- The value stored at location or index 7 is not a match with search element, rather it is more than what we are looking for. (80 > 60)



- Now, we change **high = mid -1** and find the new mid again.
- high = 7 - 1 = 6
- mid = 5 + (6 - 5) / 2 = 5. Now the mid value is 5.



- Now we compare the value stored at location 5 with 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.

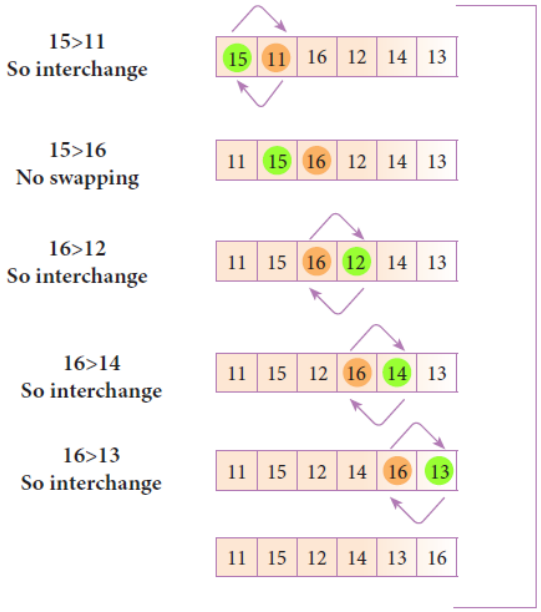
Bubble sort

- Bubble sort is a simple sorting algorithm. The algorithm 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 for 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.

Procedure

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. Go to Step 1 and repeat until end of the index is reached.

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

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. The given problem is divided into smaller and yet smaller possible 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.

Example

- Fibonacci series generates the subsequent number by adding two previous numbers. Fibonacci series starts from two numbers − Fib 0 & Fib 1.
- The initial values of Fib 0 & Fib 1 can be taken as 0 and 1.
- Fibonacci series satisfies the following conditions :

$$\mathbf{Fib_n = Fib_{n-1} + Fib_{n-2}}$$

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: Go to 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

Chapter - 5 : Python – Variables And Operators

II. Answer the following questions: (2 Marks)

1. What are the different modes that can be used to test Python Program?

- Interactive mode and Script mode are the two modes that can be used to test python program.

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 ?

- Arithmetic operators
- Relational or Comparative operators
- Logical operators
- Assignment operators
- Conditional Operator

4. What is a literal? Explain the types of literals ?

- Literal is a raw data given to a variable or constant.
- In Python, there are various types of literals.
  1) Numeric
  2) String
  3) Boolean

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

III. Answer the following questions: (3 Marks)

1. Write short notes on Arithmetic operator with examples.

- An arithmetic operator is a mathematical operator that takes two operands and performs a calculation on them.
- Arithmetic Operators:
  +, -, *, /, % (modulus), ** (exponent) and // (integer division)
- Example: If the value is a=100 and b=10
  (i) a + b = 110 (ii)a % 30 = 10 (iii) a // 30 = 3 (iv) a**2 = 10000

2. What are the assignment operators that can be used in Python?

- In Python, = is a simple assignment operator to assign values to variable.
- There are various compound operators in Python like +=, -=, *=, /=, %=, **= and //= are also available.
- Assignment Operators:
  =, +=, -=, *=, /=, %=,**= and //=
- Example: (i) x = 10 (ii) x += 20 # x = x + 20 (iii) x //= 3 # x = x // 3

3. Explain Ternary operator with examples.

- Ternary operator is also known as conditional operator that evaluate 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.
- Syntax: Variable_Name = [on_true] if [Test expression] else [on_false]
- Example: min = 50 if 49<50 else 70

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: "\t" is a tab, "\n" is a newline, and "\r" is a carriage return.

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\nLanguage")	Python Language
\t	Tab	>>>print("Hi \t Hello")	Hi Hello

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:
  s = ”Python”
  c = “P”
  m = ‘‘‘This is a multiline string with more than one lines’’’

IV. Answer the following questions: (5 Marks)

1. Describe in detail the procedure Script mode programming.

- Basically, 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.

Creating and Saving Scripts in Python

1. Choose File → New File or press Ctrl + N in Python shell window that appears an untitled blank script text editor.
2. Type the following code
   a =100
   b = 350
   c = a+b print ("The Sum=", c)
3. Choose File → Save or Press Ctrl + S. Now, Save As dialog box appears on the screen.
4. Type the file name in File Name box. Python files are by default saved with extension .py.

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.
3. To correct the errors, go back to Script editor, make corrections, save the file using Ctrl + S or File → Save and execute it again.
4. For all error free code, the output will appear in the IDLE window of Python.

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

print() function:

- In Python, the print() function is used to display result on the screen.
- It displays an entire statement which is specified within print ( ).
- **The syntax is,**
  print(“String’’)
  print(variable)
  print(“string”, variable)
  print(“string1”, varl, “string2”’, var2)
- Examples:
  print(“Welcome’’)
  print(x)

```
print( "The value of x = ",x)
print( " The sum of ",x," and ",y," is ", x+y )
```

- The print ( ) evaluates the expression before printing it on the monitor.
- 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 is, Variable = input ("prompt string")
- Where, prompt string is a statement or message to the user, to know what input can be given.
- **Example 1: input( ) with prompt string**
  city=input ("Enter Your City: ")
- 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 2:input( ) without prompt string**
  city=input()
- 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.

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

1) 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.
- Examples : Sum, total_marks, regno, num1

2) Keywords:

- Keywords are special words used by Python interpreter to recognize the structure of program.
- As these words have specific meaning for interpreter, they cannot be used for any other purpose.
- Few python's keywords are for, while, lamba, del, if, and, or,…

3) 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.

4) Delimiters:

- Python uses the symbols and symbol combinations as delimiters in expressions, lists, dictionaries and strings.
- The following are few delimiters: ( ) [ ] { } , : . ; "

5) Literals:

- Literal is a raw data given to a variable or constant.
- In Python, there are various types of literals.
  1) Numeric
  2) String
  3) Boolean

Chapter - 6 : Control Structures

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

1. List the control structures in Python.

- Sequential
- Alternative or Branching
- Iterative or Looping

2. Write note on break statement.

- The break statement terminates the loop containing it.
- When the break statement is executed, the control flow of the program comes out of the loop and starts executing the segment of code after the loop structure.
- If break statement is inside a nested loop, break will terminate the innermost loop.
- Syntax: break

3. Write is the syntax of if .. else statement.

```
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 is 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:
  (i) range(1, 10) → start from 1 and end at 9
  (ii) range(1, 10, 2) → returns 1 3 5 7 9

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

```
str="ABCDE"
i=1
while(i<=len(str)):
  print(str[0:i], sep=" ")
  i+=1
```

2. Write note on if .. else structure.

- The if .. else statement is used to choose between two alternatives based on a condition.
- Syntax:

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

- Example:

```
a = int(input("Enter any number :"))
if a%2==0:
    print (a, " is an even number")
else:
    print (a, " is an odd number")
```

- Output:

```
Enter any number :56
56 is an even number
```

3. Using if .. else .. elif statement write a suitable program to display largest of 3 numbers.

```
a=int(input("Enter the first number:"))
b=int(input("Enter the second number:"))
c=int(input("Enter the third number:"))
if (a>b) and (a>c):
    print(a, " is the largest number")
elif (b>c):
    print(b, " is the largest number ")
else:
    print(c, " is the largest number ")
```

4. Write the syntax of while loop.

```
while <condition>:
    statements block 1
[else:
    statements block2]
```

5. List the differences between break and continue statements.

- The break statement terminates the loop when it is executed.
- Continue statement is used to skip the remaining part of a loop and start with next iteration.

IV. Answer the following questions: (5 Marks)

1. Write a detail note on for loop.

- The for loop is usually known as a definite loop, because the programmer knows exactly how many times the loop will be executed.
- Syntax:

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

- The for .... in statement is a looping statement used in Python to iterate over a sequence of objects, i.e., it goes through each item in a sequence.
- Here the sequence is the collection of ordered or unordered values or even a string.
- Usually in Python, for loop uses the range() function in the sequence to specify the initial, final and increment values.
- The syntax of range() is

range (start, stop[,step]) where,

  - start – refers to the initial value
  - stop – refers to the final value
  - step – refers to increment value

- Example :

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

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'.
- Syntax:

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

- 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:

```
a=int(input("Enter the first number: "))
b=int(input("Enter the second number: "))
c=int(input("Enter the third number: "))
if (a>b) and (a>c):
    print(a, " is the largest number")
elif (b>c):
    print(b, " is the largest number ")
else:
    print(c, " is the largest number ")
```

3. Write a program to display all 3 digit odd numbers.

```
for i in range(101,1000,2):
    print(i, end='\t')
```

4. Write a program to display multiplication table for a given number.

```
n = int(input("Enter a Number :"))
x = int(input("Enter number of terms :"))
for i in range(1, x+1):
    print(i, " x ", n, " = ", i*n)
```

Chapter - 7 : Python Functions

II. Answer the following questions: (2 Marks)

1. What is function?

- Functions are group of related statements that performs a specific task.

2. Write the different types of function?

- User defined functions
- Built in functions
- Lambda functions
- Recursion functions

3. What are the main advantages of function?

- 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. ie, area where we can refer it.
- We can say that scope holds the current set of variables and their values.
- There are two types of scopes
    - local scope
    - 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.

- 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.

- 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))
```

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

1.Write the rules of local variable.

- When a variable is created inside the function/block, the variable becomes local to it.
- A local variable only exits while the function is executing.
- The formal arguments are also local to function.

2. Write the basic rules for global keyword in Python.

- When we define a variable outside a function, it's global by default. We don't have to use global keyword.
- We use global keyword to read and write a global variable inside a function.
- Using global keyword word outside a function has no effect.

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

- Without using the global keyword we cannot modify the global variable inside the function but we can only access the global variable

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

ceil ()	**floor()**
Returns the largest integer less than or equal to x	Returns the smallest integer greater than or equal to x
Syntax: math.ceil (x)	Syntax: math.floor(s)
Example: x=26.7 print (math.ceil(x)) Output: 27	Example: x=26.7 print (math.floor(x)) Output: 26

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

```
y=int (input("Enter the year:"))
if y % 4 = = 0:
        print("LEAP YEAR")
else:
        print("NOT 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,

```
>>>n2 = eval (input ("Enter an arithmetic expression."))
Enter an arithmetic expression: 12.0+ 13.0*2
>>>n2
38.0
```

7. How recursive function works?

- When a function calls itself is known as recursion. Recursion works like loop but sometimes it makes more sense to use recursion than loop. We can convert any loop to recursion.
- Overview of how recursive function works:
    - Recursive function is called by 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 parenthesis when you define a function.

- The code block always comes after a column and is indented.
- The statement "return (expression)" exists a function optionally passing back an expression to the caller.
- A "return" with no arguments is the same as return none.

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

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

- Functions are a group of related statements that performs a specific task.
- There are four types of functions. They are
  - User-defined functions
  - Built-in functions
  - Lambda functions
  - Recursion function

User-defined functions:

- Functions defined by the users themselves. Functions must be defined, to create and use certain functionality
- Function blocks begin with the keyword "def" followed by function name and parenthesis ()
- Syntax:

```
def <function-name((parameter 1, parameter2.. }}>:
        <Block of statements>
        return expression/none>
```

- Example

```
def hello():
        print ("hello-Python")
        return
```

Built-in function:

- Functions that are inbuilt with Python.
- function → abs() Returns an absolute value of a number. The argument may be an integer or a floating point number.
- Syntax:

```
abs(x)
```

- Example:

```
x=-20
print('x=', abs(x))
```

Lambda function:

- Functions that are anonymous unnamed function.
- In Python anonymous functions are defined using the lambda keyword.
- Lambda function is mostly used for creating small and one-time anonymous function.
- Lambda function can take any number of arguments and must return one value in the forms of an expression
- Syntax:

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

- Example:

```
Sum = lambda arg1, arg 2: arg1 + arg 2
print (sum (30,40))
```

Output:

```
The sum is 70
```

Recursion Function:

- Function that calls itself is known as recursive.
- Recursion works like loop but sometimes it makes more sense to use recursion than loop.
- We can convert any loop to recursion.
- 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.
- Example:

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

2. Explain the scope of variables with an example.

- Scope of variable refers to the part of the program, where it is accessible, ie, area where we can refer it.
- We can say that scope holds the current set of variables and their values.
- There are two types of scopes. They are,
  - Local scope
  - Global scope

Local scope

- A variable declared inside the functions body or in the local scope is known as local variable.
- Rules
  - A variable with local scope can be accessed only within the function that it is created in.
  - When a variable is created inside the function, the variable becomes local it A local variable only exists while the function is executing.
  - The formal arguments are also local to function
- Example:

```
def loc:
    y=0
    print (y)
loc()
```

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.
- Rules:
  - When we define a variable outside a function, it's global by default.
  - We use global keyword to read and write a variable inside a function.
  - Use of global keyword outside a function has no effect.
- Example:

```
c=l
def add():
    print (c)
add ()
```

3. Explain the following built-in functions.

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

(a) id() :

- Return the "identity" of an object. i.e. the address of the object in memory.
- **Syntax:** id (object)
- **Example:**

```
x=15
print ('address of x is :',id (x))
```

Output: address of x is : 1357486752

(b) chr() :

- It returns the Unicode character for the given ASCII value.
- **Syntax:** chr (i)
- **Example:**

```
c=65
print (chr (c))
```

Output: A

(c) round()

- It returns the nearest integer to its input.
- First argument (number) is used to specify the value to be rounded.
- Second argument (n digits) is used to specify the number of decimal digits desired after rounding.
- **Syntax:** round (number [,ndigits])
- **Example:**

```
n1=17.89
print (round (n1,1))
```

Output: 17.9

(d) type() :

- It returns the type of object for the given single object.
- **Syntax:** type (object)
- **Example:**

```
x= 15.2
print (type (x))
```

Output: <class 'float'>

(e) pow() :

- It returns the computation of ab i.e. (a**b) a raised to the power of b.
- **Syntax:** pow (a,b)
- **Example:**

```
a= 5
b= 2
print (pow (a,b))
```

Output: 25

4. Write a Python code to find the LCM of two Numbers.

```
def lcm(x,y):
    if x>y:
        greater = x
    else:
        greater =y
    while (True):
        if ((greater % x == 0) and (greater % y==0)):
            lcm=greater
            break
        greater += 1
    return lcm
x=int (input("Enter the first number:"))
y=int (input ("Enter the second number:"))
print(lcm(x,y))
```

5. Explain recursive Function with an example.

- When a function calls itself is known as recursion.
- Recursion works like loop but sometimes it makes more sense to use recursion than loop. We can convert any loop to recursion.
- A process would iterate indefinitely if not stopped by some condition then that process is known as infinite iteration.
- The condition that is applied in any recursive function is known as base condition.
- 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 source required processing and there calls itself to continue recursion.

- Example:

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

Output:

120

Chapter - 8 : Strings and String Manipulation

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?

- No, strings in Python are immutable.

3. How will you delete a string in Python?

- Python will not allow deleting a string. Whereas you can remove entire string variable using del command.
- Example:

```
>>> str1="Hello"
>>> del str1
```

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. Thus, [ ] is also known as slicing operator.

III. Answer the following questions: (3 Marks)

1. Write a python program to display the given pattern.

```
COMPUTER
COMPUTE
COMPUT
COMPU
COMP
COM
CO
C
```

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

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

(a) capitalize():

- Syntax: capitalize()
- Description: Used to capitalize the first character of the string
- Example:

```
>>> city="Chennai
>>> print (city, capitalize())
```

- Output: Chennai

b) swapcase()

- Syntax: swapcase()
- Description: It will change case of every character to its opposite case vice-versa.
- Example: >>Strl-AmilNaDu

```
>>> print (str1.Swapcase())
```

- Output: TaMilnAdU

3. What will be the output of the given Python program?

strl="Welcome" str2="to school" str3= str1[1:2]+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:

```
numl=int (input ("Number1:"))
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.

- Syntax:

```
Count (str, beg, end)
```

- Description:
  - 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 searched in whole string. Search is case sensitive.
- Example:

```
>>> Stri="Raja Raja Chozhan"
>>>print (str 1.count ("Raja'))
    2
```

IV. Answer the following questions: (5 Marks)

1. Explain about string operators in python with Suitable example.

- String operators Python provides the operators for String operations.
- These operators are useful 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"
    "Welcome Python"
```

(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:

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

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.
- It is an ordered collection of values enclosed within square brackets[]. Each value of a list is called as element.

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

- Python enables reverse or negative index 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)
```

Output: 4

4. Differentiate del with remove() function of List

- del statement is used to delete known elements whereas remove () function is used to delete elements of a list if it's the index is unknown, del can also be used to delete entire list.

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

- tuple_Name=(E1, E2, E3……,En)

6. What is set in Python?

- A Set is another type of collection data type.
- A Set is a mutable and unordered collection of elements without duplicates.
- That means the elements within a set cannot be repeated**.**

III. Answer the following questions: (3 Marks)

1. What are the difference between list and Tuples?

- The elements of a list are changeable whereas the elements of a tuple are unchangeable, 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 parenthesis().
- Iterating tuples is faster than list.

2. Write a short note about sort().

- sort() function arranges a list value in ascending order by default.
- Syntax: list.sort([reverse=True|False, key=myFunc])
- Both arguments are optional.
- If reverse is set as True, list sorting is in descending order.

Example:

```
MyList=[5,2,3,4,1]
MyList.sort( )
print(MyList)
MyList.sort(reverse=True)
print(MyList)
```

Output:

```
[1,2,3,4,5]
[5,4,3,2,1]
```

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.

- In Python dictionary, del keyword is used to delete a particular element.
- The clear() function is used to delete all the elements in a dictionary.
- To remove the dictionary, we can use del keyword with dictionary name.
- Example

```
Dict = {'Mark1' : 98, 'Marl2' : 86}
del Dict['Mark1']
```

```
print(Dict)
Dict.clear()
print(Dict)
del Dict
print(Dict)
```

- Output:

```
{'Mark2': 86}
{ }
NameError: name 'Dict' is not defined
```

5. List out the set operations supported by python.

- Python supports the set operations such as

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

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

(iii) Difference (-):
  - It includes all elements that are in first set (say set A) but not in the second set (say set B)

(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.

6. What are the difference between List and Dictionary?

List	Dictionary
List is an ordered set of elements.	Dictionary is a data structure that is used for matching one element (Key) with another (Value).
The index values can be used to access a particular element.	In dictionary key represents index.
Lists are used to look up a value.	It is used to take one value and look up another value.

IV. Answer the following questions: (5 Marks)

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

- There are three methods are used to insert the elements in a list.

(i) append():

- In Python, append( ) function is used to add a single element to an existing list.
- Syntax: List.append(element to be added)
- Example:

```
>>>Marks = [10, 23, 41, 75]
>>>Marks.append(60)
>>>print(Marks)
[10, 23, 41, 75, 60]
```

(ii) extend():

- In python, extend( ) function is used to add more than one element to an existing list.
- Syntax: List.extend([elements to be added])
- Example:

```
>>>Marks = [10, 23, 41, 75]
>>>Marks.extend([60,80,55])
>>>print(Marks)
[10, 23, 41, 75, 60,80,55]
```

(iii) insert():

- The insert( ) function is used to insert an element at any position of a list.
- Syntax: List.insert (position index, element)
- Example:

```
>>>Marks = [10, 23, 41, 75]
>>>Marks.insert(2,60)
>>>print(Marks)
[10, 23, 60, 41, 75]
```

- While inserting a new element in between the existing elements, at a particular location, the existing elements shifts 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.
- Syntax: range (start value, end value, step value)

  where,

  start value – beginning value of series
  end value – final value of series.
  step value – It is an optional argument, which refers to increment value.
- Example:

```
for x in range (2, 11, 2):
print(x, end=' ')
```

Output: 2 4 6 8 10

Creating a list with range() function:

- Using the range( ) function, you can create a list with series of values.
- 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)
    [2, 4, 6, 8, 10]
```

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

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

- 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:

```
voters=(("Aruna", "F", 36), ('Ram", "F", 47), ("Suriya", "M", 35), ("Kamal", "M", 23))
for i in voters:
    print(i)
```

Output:

```
('Aruna', 'F', 36)
('Rani', 'F', 47)
('Suriya', 'M', 35)
('Kamal', 'M', 23)
```

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

- Python supports the set operations such as Union, Intersection, Difference and Symmetric difference.

(i) Union:

- It includes all elements from two or more sets.
- In python, the operator | and the function union( ) are used to join two sets in python.
- Example:

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

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

(ii) Intersection:

- It includes the common elements in two sets.
- The operator & and the function intersection( ) are 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) (OR) print(set_A.intersection(set_B))
```

Output: {'A', 'D'}

(iii) Difference:

- It includes all elements that are in first set but not in the second set.
- The minus (-) operator and the function difference( ) are used to difference operation.
- Example:

```
set_A={'A', 2, 4, 'D'} .
set_B={'A', 'B', 'C', 'D'}
print(set_A - set_B) (OR) print(set_A.diff erence(set_B))
```

Output: {2, 4}

(iv) Symmetric difference:

- It includes all the elements that are in two sets but not the one that are common to two sets.
- The caret (^) operator and the function symmetric_difference() are used to symmetric difference set operation in python.
- Example:

```
set_A={'A', 2, 4, 'D'}
set_B={'A', 'B', 'C', 'D'}
print(set_A ^ set_B) (OR) print(set_A.symmetric_difference(set_B))
```

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

Chapter - 10 : Python Classes and Objects

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

1. What is class?

- class is a template for the object. It is the main building block in Python.

2. What is instantiation?

- The process of creating object is called as "Class Instantiation".

3. What is the output of the following program?

```
class Sample:
    _ _num=10
    def disp(self):
        print(self._ _num)
S=Sample()
S.disp()
print(S._ _num)
```

Output:
10
Attribute Error: 'Sample' object has no attribute '_ _num'

4. How will you create constructor in Python?

- Constructor is the special function called "init" that is automatically executed when an object of a class is created.
- It must begin and end with double underscore.
- Syntax:

```
def __init__(self, [args ……..]):
    <statements>
```

- Example:

```
class Sample:
    def __init__(self):
        print("Constructor of class Sample...")
S=Sample()
```

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.
- _ _del_ _( ) method is used as destructor.

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

1. What are class members? How do you define it?

- Variables and functions defined inside a class are called as "Class Variable" and "Methods" respectively.

- Class variable and methods are together known as members of the class.
- Example:

```
class Sample:
    x = 10
    def disp(self):
        print(Sample.x)
s = Sample()
s.disp()
```

2. Write a class with two private class variables and print the sum using a method.

```
class sample:
    def _ _init_ _(self,n1,n2):
        self._ _n1=n1
        self._ _n2=n2
    def sum(self):
        print(self._ _n1 + self._ _n2)
s=sample(12,14)
s.sum()
```

Output:

26

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

Answer: The statement "del F.display" should be removed to get the given output.

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')
obj.display()
```

Output: Good Morning Bindu Madhavan

5. How do define constructor and destructor in Python?

Syntax of Constructor:

```
def _ _init_ _(self, [args ……..]):
    <statements>
```

Syntax of Destructor:

```
def _ _del_ _(self):
    <statements>
```

Example:

```
class sample:
    def _ _init_ _(self,n1):
        self._ _n1=n1
        print("Constructor of class Sample...")
    def _ _del_ _(self):
        print("Destructor of class Sample...")
s=sample(5)
```

Output:

Constructor of class Sample...

Destructor of class Sample...

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

1. Explain about constructor and destructor with suitable example.

Constructor

- Constructor is the special function that is automatically executed when an object of a class is created.
- In Python, there is a special function called "init" which act as a Constructor.
- It must begin and end with double underscore.
- It is executed automatically when the object is created.
- This constructor function can be defined with or without arguments.
- This method is used to initialize the class variables.
- General format:

```
def _ _init_ _(self, [args……]):
    <statements>
```

- Example:

```
class Sample:
    def __init__(self, num):
        print("Constructor of class Sample...")
        self.num=num
        print("The value is :", num)
S=Sample(10)
```

Destructor

- Destructor is also a special method to destroy the objects.
- In Python, _ _del_ _( ) method is used as destructor. It is just opposite to constructor.
- The __del__ method gets called automatically when we deleted the object reference using the del.
- Example Program

```
class Sample:
    num=0
    def __init__(self, var):
        Sample.num+=1
        self.var=var
        print("The object value is = ", self.var)
        print("The value of class variable is= ", Sample.num)
    def __del__(self):
        Sample.num-=1
        print("Object with value %d is exit from the scope"%self.var)
S1=Sample(15)
S2=Sample(35)
S3=Sample(45)
```

Chapter - 11 : Database Concepts

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

1. Mention few examples of a DBMS.

- Dbase
- FoxPro

2. List some examples of RDBMS.

- SQL Server
- Oracle
- MySQL
- SQLite
- MariaDB
- MS Access

3. What is data consistency?

- Data Consistency means that data values are the same at all instances of a database

4. What is the difference between Hierarchical and 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 the data in many-to-many relationships.
- Network model is easier and faster to access the data.

5. What is normalization?

- Database normalization is the process to reduce data redundancy and improve data integrity.

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

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

Select	Project
The SELECT operation is used for selecting a subset with tuples according to a given condition.	The projection eliminates all attributes of the input relation but those mentioned in the projection list.
Select filters out all tuples that do not satisfy the condition.	The projection method defines a relation that contains a vertical subset of Relation.
symbol : σ	symbol : Π

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.
- A x B means A times B, where the relation A and B have different attributes.

Student

Rollno	Name
01	Ram
102	Raj

Course

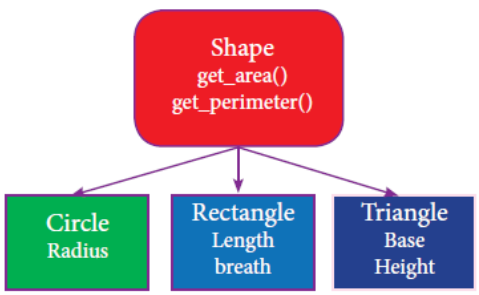
Crollno	Cno	CName
101	C120	JAVA
102	C121	C++

Student X Course

Rollno	Name	Crollno	Cno	CName
101	Ram	101	C120	JAVA
101	Ram	102	C121	C++
102	Raj	101	C120	JAVA
102	Raj	102	C121	C++

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 behaviours.



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. DBA takes care of the security of the DBMS, managing the license keys, managing user accounts and access etc.

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:

- Are responsible for identifying the data to be stored in the database for choosing appropriate structures to represent and store the data.

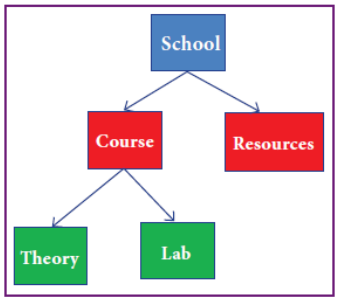
IV. Answer the following questions: **(5 Marks)**

1. Explain the different types of data model.

- A data model describes how the data can be represented and accessed from a software after complete implementation
- Following are the different types of a Data Model
  - Hierarchical Model
  - Relational Model
  - Network Database Model
  - Entity Relationship Model
  - 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 i.e 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.



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).

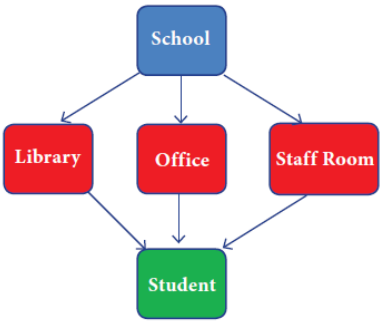
Stu_id	Name	Age
1	Malar	17
2	Suncar	16
3	Velu	16

Subj_id	Name	Teacher
1	C++	Kannan
2	Php	Ramakrishnan
3	Python	Vidhya

Stu_id	Subj_id	Marks
1	1	92
1	2	89
3	2	96

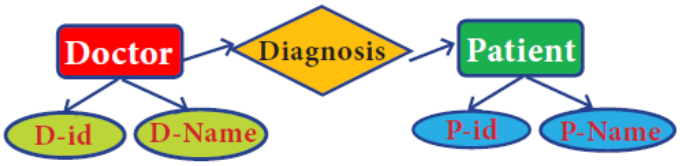
Network Database Model

- Network database model is an extended form of hierarchical data model.
- The difference between hierarchical and Network data model is :
    - In hierarchical model, a child record has only one parent node
    - 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.



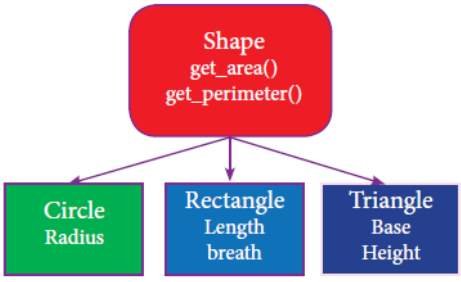
Entity Relationship Model

- In this database model, relationship are created by dividing the object into entity and its characteristics into attributes.
- This model is useful in developing a conceptual design for the database.
- It is very simple and easy to design logical view of data.
- Rectangle represents the entities.
- Ellipse represents the attributes.
- Diamond represents the relationship in ER diagrams



Object Model

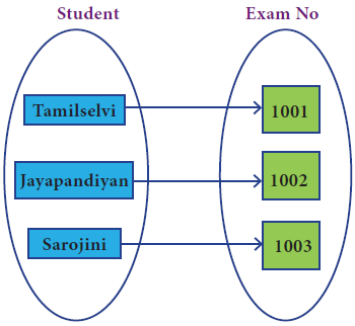
- Object model stores the data in the form of objects, attributes and methods, classes and Inheritance.
- This model handles more complex applications, such as Geographic information System (GIS), scientific experiments, engineering design and manufacturing.
- It is used in file Management System.
- It represents real world objects, attributes and behaviors.
- It provides a clear modular structure.
- It is easy to maintain and modify the existing code



2. Explain the different types of relationship mapping.

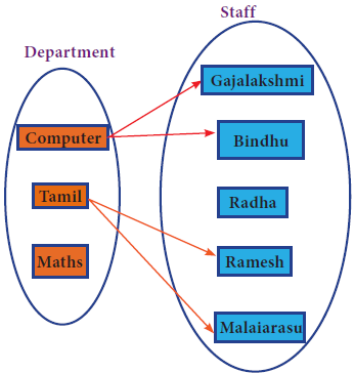
One-to-One Relationship:

- 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.



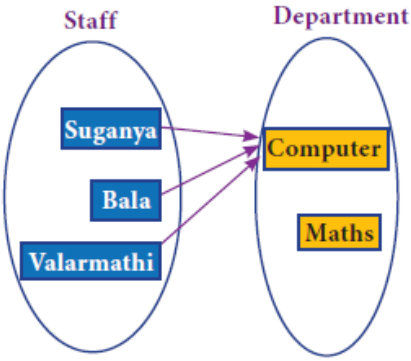
One-to-Many Relationship:

- 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.



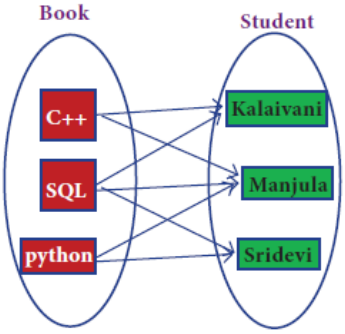
Many-to-One Relationship:

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



Many-to-Many Relationship:

- A many-to-many relationship occurs when multiple records in a table are associated with multiple records in another table.
- For example:
    - Many Books in a Library are issued to many students.



3. Differentiate DBMS and RDBMS.

Basis of Comparison	DBMS	RDBMS
Expansion	Database Management System	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	Present	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, mysql, MariaDB, SQLite, MS Access.

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

- Relational algebra operations are performed recursively on a relation (table) to yield an output.
- The output of these operations is a new relation, which might be formed by one or more input relations.
- Relational Algebra is divided into various groups
    - Unary Relational Operations
        - SELECT (symbol : σ)
        - PROJECT (symbol : Π)
    - Relational Algebra Operations from Set Theory
        - UNION (∪)
        - INTERSECTION (∩)
        - DIFFERENCE (−)
        - CARTESIAN PRODUCT (X)

SELECT (symbol: σ)

- General form σc (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.

STUDENT

Studno	Name	Course
cs1	Kannan	Big Data
cs2	Gowri Shankar	R language
cs3	Lenin	Big Data
cs4	Padmaja	Python Programming

σ_{course} = "Big Data" (STUDENT)

Studno	Name	Course
cs1	Kannan	Big Data
cs3	Lenin	Big Data

PROJECT (symbol: Π)

- 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.

Π_{course} (STUDENT)

Result

Course
Big Data
R language
Python Programming

UNION (Symbol: ∪)

- 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 ∪ B

Table A	
Studno	Name
cs1	Kannan
cs3	Lenin
cs4	Padmaja

Table B	
Studno	Name
cs1	Kannan
cs2	GowriShankar
cs3	Lenin

Table A ∪ B	
Studno	Name
cs1	Kannan
cs2	GowriShankar
cs3	Lenin
cs4	Padmaja

INTERSECTION (symbol: ∩) A ∩ 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.

A ∩ B	
cs1	Kannan
cs3	Lenin

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.

Table A – B	
cs4	Padmaja

PRODUCT OR CARTESIAN PRODUCT (Symbol: X)

- 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.

Student

Rollno	Name
01	Ram
102	Raj

Course

Crollno	Cno	CName
101	C120	JAVA
102	C121	C++

Student X Course

Rollno	Name	Crollno	Cno	CName
101	Ram	101	C120	JAVA
101	Ram	102	C121	C++
102	Raj	101	C120	JAVA
102	Raj	102	C121	C++

5. Explain the characteristics of RDBMS.

Ability to manipulate data	• RDBMS provides the facility to manipulate data in a database.
Reduced Redundancy	• RDBMS follows Normalisation which divides the data in such a way that repetition is minimum.
Data Consistency	• Data Consistency means that data values are the same at all instances of a database
Support Multiple user and Concurrent Access	• RDBMS allows multiple users to work on it at the same time and still manages to maintain the data consistency.
Query Language	• RDBMS provides users with a simple query language, using which data can be easily fetched, inserted, deleted and updated in a database.
Security	• The RDBMS also takes care of the security of data, protecting the data from unauthorized access. • It 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 (SQL)

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

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

SELECT name FROM student WHERE age<18 ORDER BY age;

2. Differentiate Unique and Primary Key constraint.

Unique Constraint	Primary Key Constraint
It ensures that no two rows have the same value in the specified columns.	It declares a field as a Primary key which helps to uniquely identify a record.
More than one fields of a table can be set as primary key.	Only one field of a table can be set as primary key.

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

- **Table constraint:** Table constraint apply to a group of one or more columns.
- **Column constraint:** Column constraint apply only to individual column.

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

Component	Command	Description
DDL (Data Definition Language)	CREATE TABLE	To create tables in the databases
DML (Data Manipulation Language)	INSERT	To insert values in to tables

5. What is the difference between SQL and MySQL?

- SQL: Structured Query Language is a language used for accessing database.
- MySQL: MySQL is a Database Management System like SQL Server, Oracle. MySQL is a RDBMS.

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 Key Constraint:

- This constraint declares a field as a Primary key which helps to uniquely identify a record.
- 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 student ADD address char(20);

3. Write any three DDL commands.

- 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.

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

- The SAVEPOINT command is used to temporarily save a transaction so that you can rollback to the point whenever required.
- The different states of our table can be saved at anytime using different names and the rollback to that state can be done using the ROLLBACK command.
- Syntax: SAVEPOINT savepoint_name;
- Example

Admno	Name	Age
105	Sumitha	21
106	Selvam	48
107	Aakash	14

UPDATE Student SET Name = 'Mini' WHERE Admno=105;
SAVEPOINT A;
SELECT * FROM STUDENT;

Admno	Name	Age
105	***Mini***	21
106	Selvam	48
107	Aakash	14

ROLLBACK TO A;
SELECT * FROM STUDENT;

Admno	Name	Age
105	***Sumitha***	21
106	Selvam	48
107	Aakash	14

5. Write a SQL statement using DISTINCT keyword.

SELECT DISTINCT place FROM student;

IV. Answer the following questions: (5 Marks)

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

- Constraints ensure database integrity, therefore known as database integrity constraints.
- The different types of constraints are as follows,

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
);
```

PRIMARY KEY Constraint:

- It is similar to unique constraint except that only one field of a table can be set as primary key which helps to uniquely identify a record.
- 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 PRIMARY KEY, → Primary Key Constraint
```

```
Name char(20) NOT NULL
);
```

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 PRIMARY KEY,
Name char(20) NOT NULL,
Age integer DEFAULT 17 → Default Constraint
);
```

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 PRIMARY KEY
Name char(20) NOT NULL,
Age integer (CHECK Age <=19), → Check Constraint
);
```

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.
- Example,

```
CREATE TABLE Student1
(
Admno integer NOT NULL,
Firstname char(20),
Lastname char(20),
PRIMARY KEY (Firstname, Lastname) → Table constraint
);
```

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.
ii) To display all employees whose allowance is between 5000 and 7000.
iii) To remove the employees who are mechanic.
iv) To add a new row.
v) To display the details of all employees who are operators.

i) SELECT * FROM EMPLOYEE ORDER BY PAY DESC ;
ii) SELECT * FROM EMPLOYEE WHERE ALLOWANCE BETWEEN 5000 AND 7000 ;
iii) DELETE FROM EMPLOYEE WHERE DESIG = 'Mechanic' ;
iv) INSERT INTO EMPLOYEE VALUES ('S1006', 'Rajesh', 'Mechanic', 20000, 7000) ;
v) SELECT * FROM EMPLOYEE WHERE DESIG = 'Operator' ;

3. What are the components of SQL? Write the commands in each.

SQL commands are divided into five categories:

i. DML - Data Manipulation Language
ii. DDL - Data Definition Language

iii. DCL - Data Control Language
iv. TCL - Transaction Control Language
v. DQL - Data Query Language

(i) DATA DEFINITION LANGUAGE

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

(ii) 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. i.e., the data can be manipulated using a set of procedures which are expressed by DML.
- SQL commands which comes under Data Manipulation Language are :
    - INSERT: Inserts data into a table
    - UPDATE: Updates the existing data within a table.
    - DELETE: Deletes all records from a table, but not the space occupied by them.

(iii) 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).
- 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.

(iv) 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.

(v) DATA QUERY LANGUAGE

- The Data Query Language consist 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.

(ii) SELECT statement using ORDER BY clause.

(i) SELECT statement using GROUP BY clause.

- The GROUP BY clause is used with the SELECT statement to group the students on rows or columns having identical values or divide the table in to groups.
- For example to know the number of male students or female students of a class, the GROUP BY clause may be used.
- Syntax for the GROUP BY clause is:

  SELECT <column-names> FROM <table-name> GROUP BY <column-name> HAVING [condition];
- To apply the above command on the student table :

  SELECT Gender FROM Student GROUP BY Gender;

(ii) SELECT statement using ORDER BY clause.

- The ORDER BY clause in SQL is used to sort the data in either ascending or descending based on one or more columns.
- By default ORDER BY sorts the data in ascending order.

- We can use the keyword DESC to sort the data in descending order and the keyword ASC to sort in ascending order.
- Syntax for the ORDER BY clause is:
  SELECT <column-name>[<column-name>…] FROM <table-name> GROUP BY <column-name>,<column-name> ASC|DESC;
- To display the students in alphabetical order of their names, use
  SELECT * FROM Student ORDER BY Name;

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
(
        Empcode char(20),
        Name char(20),
        Desig varchar(20),
        Pay integer,
        Allowance integer,
        PRIMARY KEY (Empcode)  →  Table Constraint
) ;
```

Chapter - 13 : Python And CSV Files

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.

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

- reader( ) function
- DictReader class.

3. Mention the default modes of the File.

- The following are the default modes of the file.
  - 'r' Open a file for reading. (default)
  - 't' Open in text mode. (default)

4. What is use of next() function?

- The next() function returns the next item from the iterator.
- It is used to skip a row of the csv file.

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 multiple indices.
- Syntax: operator.itemgetter(col_no)
- Example: sortedlist = sorted (data, key=operator.itemgetter(1))

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 two different methods are
  (i) open("sample.txt") as f:
  (ii) with open()
- The **open( )** method is not entirely safe. If an exception occurs when you are performing some operation with the file, the code exits without closing the file.
- But **"with open"** statement ensures that the file is closed when the block inside with is exited.

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

```
import csv
row = ['3', 'Meena','Bangalore']
with open('student.csv', 'r') as readFile:
    reader = csv.reader(readFile)
    lines = list(reader)
    lines[3] = row
with open('student.csv', 'w') as writeFile:
```

```
        writer = csv.writer(writeFile)
         writer.writerows(lines)
    readFile.close()
    writeFile.close()
```

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

```
import csv
with open('student.csv','r') as readFile:
    reader=csv.reader(readFile)
    for row in reader:
        print(row)
readFile.close( )
```

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

'w' - write mode:

- Open a file for writing. Creates a new file if it does not exist or truncates the file if it exists.

'a' – append mode:

- Open for appending at the end of the file without truncating it. Creates a new file if it does not exist.

5. What is the difference between reader()method and DictReader() class?

reader()	DictReader()
The reader function is designed to take each line of the file and make a list of all columns.	DictReader() creates an object which maps data to a dictionary.
csv.reader() works with list/tuple.	csv.DictReader() works with dictionary

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.

- There are two ways to read a CSV file.
    1. Use the csv module's reader function
    2. Use the DictReader class.

CSV Module's Reader Function:

- The csv.reader() method 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 (,).

- Syntax: csv.reader(fileobject, delimiter, fmtparams)
  where
  - file object: passes the path and the mode of the file
  - delimiter: an optional parameter containing the standard dialects like , | etc can be omitted.
  - fmtparams: optional parameter which help to override the default values of the dialects like skip initial space, quoting etc. can be omitted.
- Example:

```
import csv
with open('Student.csv','r') as readFile:
    reader=csv.reader(readFile)
    for row in reader:
        print(row)
readFile.close( )
```

Reading CSV File Into A Dictionary:

- To read a CSV file into a dictionary can be done by using DictReader which creates an object which maps data to a dictionary.
- DictReader works by reading the first line of the CSV and using each comma separated value in this line as a dictionary key.
- The columns in each subsequent row then behave like dictionary values and can be accessed with the appropriate key.
- Example:

```
import csv
filename="Student.csv"
myfile=csv.DictReader(open('Student.csv','r'))
for row in myfile:
    print(dict(row))
```

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

```
import csv
csv.register_dialect('myDialect', delimiter = '|', quoting=csv.QUOTE_ALL)
with open('grade.csv', 'w') as csvfile:
    fieldnames = ['Name', 'Grade']
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames, dialect="myDialect")
    writer.writeheader()
    writer.writerows ( [ {'Grade': 'B', 'Name': 'Anu'},
                         {'Grade': 'A', 'Name': 'Beena'},
                          {'Grade': 'C', 'Name':  'Tarun'} ] )
print("writing completed")
```

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. 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
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”,
, , White

Chapter - 14 : Importing C++ Programs In Python

II. Answer the following questions: (2 Marks)

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

- The theoretical difference between the two is that scripting languages do not require the compilation step and are rather interpreted.
- A scripting language requires an interpreter while a programming language requires a compiler.

2. Differentiate compiler and interpreter.

Compiler	Interpreter
Scans the entire program and translates it as a whole into machine code.	Translates program one statement at a time.
It generates the error message only after scanning the whole program.	It continues translating the program until the first error is met, in which case it stops.
Debugging is comparatively hard.	Debugging is easy.

3. Write the expansion of (i) SWIG (ii) MinGW

(i) SWIG - Simplified Wrapper Interface Generator
(ii) MinGW - Minimalist GNU for Windows

4. What is the use of modules?

- We use modules 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.

- cd command is used to change directory.
- In this Example, the prompt shows the "C:\>”. To goto the directory “pyprg”, type the command 'cd pyprg' in the command prompt.
- It changes to “c:\pyprg>”

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?

- To automate certain tasks in a program
- Extracting information from a data set
- Less code intensive as compared to traditional programming language
- 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, 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 via the command-line input, it's basically a list holding the command-line arguments of the program.
- The first argument, sys.argv[0] contains the name of the python program.
- sys.argv[1] is the next argument passed to the program.

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.

For example, >>>python pycpp.py –i pali

3. What is the purpose of sys, os, getopt module in Python.Explain

Python's sys module:

- This module provides access to built-in variables used by the interpreter. One among the variable in sys module is argv.
- sys.argv is the list of command-line arguments passed to the Python program.
- argv contains all the items that come via the command-line input, it's basically a list holding the command-line arguments of the program.
- The first argument,
  sys.argv[0] contains the name of the python program.
  sys.argv[1] is the next argument passed to the program.

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.
- Execute the C++ compiling command in the shell. For Example to compile C++ program g++ compiler should be invoked through the following command
  os.system ('g++ ' + <variable_name1> ' -<mode> ' + <variable_name2>

Python getopt module:

- This method parses command-line options and parameter list.
- Following is the syntax for this method
  <opts>,<args>=getopt.getopt(argv, options, [long_options])

4. Write the syntax for getopt() and explain its arguments and return values

- The getopt module of Python helps you to parse (split) command-line options and arguments.
- Syntax: <opts>,<args>=getopt.getopt(argv, options, [long_options])
- Here,
  - argv − This is the argument list of values to be parsed (splited)
  - 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 contains a list of strings. Argument of Long options should be followed by an equal sign ('=').
  - In our program the C++ file name along with its path will be passed as string and 'i' i will be also passed 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 and path.
  - args contains error string, if at all the comment is given with wrong path or mode.
  - args will be an empty list if there is no error.

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:")
    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)
    os.system(exe_file)
    if __name__=='__main__':
        main(sys.argv[1:])
```

Chapter - 15 : Data Manipulation Through SQL

II. Answer the following questions: (2 Marks)

1. Mention the users who uses the Database.

- Users of database can be human users, other programs or applications.

2. Which method is used to connect a database? Give an example.

- connect() is used to connect/open the existing databases.
- Example: connection = sqlite3.connect ("Academy.db")

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: cursor.execute("INSERT INTO Student (Rollno, Sname) VALUES (1561, 'Aravind');")

5. Which method is used to fetch all rows from the database table?

- cursor.fetchall() - fetchall () method is to fetch all rows from the database table.
- Example: result = cursor.fetchall()
  print(result)

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 files or even in the internal memory of the computer.

Advantages:

- It is designed to be embedded in applications, instead of using a separate database server program such as MySQL or Oracle.
- 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()
This method returns the next row of a query result set or None in case there is no row left.	This method returns the next number of rows (n) of the result set. (Displaying specified number of records)
It takes no argument.	It takes one argument.
Example: res = cursor.fetchone()	Example: res = cursor.fetchmany(3)

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.
- For example, to display the different grades scored by male students from “student table” the following code can be used.

```
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

```
import sqlite3
connection=sqlite3.connect("organization.db")
cursor=connection.cursor()
cursor.execute("create table employee(eno integer primary key, empname varchar(20),
                esal integer, dept varchar(20));")
cursor.execute("insert into employee values ('1001', 'Raja', '10500', 'Sales');")
cursor.execute("insert into employee values ('1002', 'Surya', '9000', 'Purchase');")
cursor.execute("insert into employee values ('1003', 'Peter', '8000', 'Production');")
connection.commit()
print("Employee Details Department-wise :")
cursor.execute("Select * from employee order by dept;")
result=cursor.fetchall()
```

```
print(*result, sep="\n")
connection.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.

```
import sqlite3
connection=sqlite3.connect("organization.db")
cursor=connection.cursor()
cursor.execute("create table employee(eno integer primary key, empname varchar(20),
        esal integer, dept varchar(20));")
cursor.execute("insert into employee values ('1001', 'Raja', '10500','Sales');")
cursor.execute("insert into employee values ('1002', 'Surya', '9000','Purchase');")
cursor.execute("insert into employee values ('1003', 'Peter', '8000','Production');")
connection.commit()
print("Records in descending order of employee number:")
cursor.execute("Select * from employee order by eno desc")
result=cursor.fetchall()
print(*result,sep="\n")
connection.close()
```

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.
- SQLite is fast, rigorously tested, and flexible, making it easier to work.
- Python has a native library for SQLite.
- To use SQLite,
    - Step 1 - import sqlite3
    - Step 2 - create a connection using connect() method and pass the name of the database file. 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(). It 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.
- Example:

```
import sqlite3
connection=sqlite3.connect("organization.db")
cursor=connection.cursor( )
```

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

```
import sqlite3
connection=sqlite3.connect("inventory.db")
cursor=connection.cursor()
cursor.execute("""select * from product;""")
print("Displaying all records in the table")
result=cursor.fetchmany(5)
print(*result, sep="\n")
connection.close()
```

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()
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

```
import sqlite3
connection=sqlite3.connect("ABC.db")
cursor=connection.cursor()
cursor.execute("drop table item;")
cursor.execute("create table item(icode integer primary key, itemname varchar(25), rate
integer);")
cursor.execute("insert into item values('1005','Printer','8000');")
connection.commit()
connection.close()
print("Item table is created and a record is added Successfully")
```

5. Consider the following table Supplier and item .Write a python script for (i) to (ii)

i) Display Name, City and Itemname of suppliers who do not reside in Delhi.
ii) Increment the SuppQty of Akila by 40

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

```
import sqlite3
connection=sqlite3.connect("item.db")
cursor=connection.cursor()
```

i) Display Name, City and Itemname of suppliers who do not reside in Delhi.

```
cursor.execute ("SELECT supplier.name, supplier.city, item.itemname FROM
                 supplier, item WHERE supplier.city <> 'Delhi' ")
ans=cursor.fetchall ( )
for i in ans:
    print()
```

ii) Increment the SuppQty of Akila by 40

```
cursor.execute ("UPDATE supplier SET supplier.SuppQty =SuppQty + 40
                 WHERE name='Akila' ")
connection.commit()
connection.close()
```

Chapter - 16 : Data Visualization Using Pyplot: Line Chart, Pie Chart And Bar Chart

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

1. What is 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
- Pie chart

4. How will you install Matplotlib?

- Install matplotlib using pip.
- Pip is a Package manager software for installing python packages.
- Syntax: python –m pip install –U matplotlib

5. Write the difference between the following functions:

plt.plot([1,2,3,4]), plt. plot([1,2,3,4], [1,4,9,16]).

Case 1: plt.plot([1,2,3,4])

- matplotlib assumes it is a sequence of y values, and automatically generates the x values for you. Python ranges start with 0, the default x vector has the same length as y but starts with 0. Hence the x data are [0, 1, 2, 3].

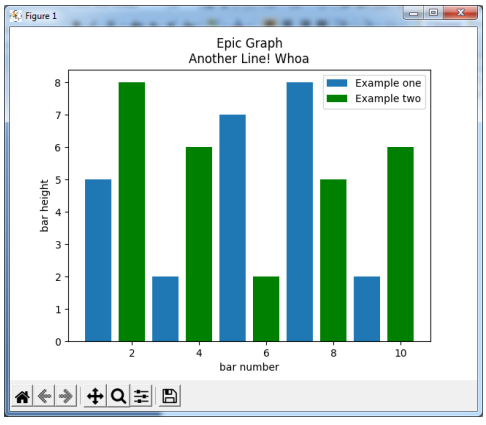
Case 2: plt.plot([1,2,3,4], [1,4,9,16])

- This plot takes many parameters, but the first two here are 'x' and 'y' coordinates. This means, you have 4 co-ordinates according to these lists: (1,1), (2,4), (3,9) and (4,16).

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

1. Draw the output for the following data visualization plot.

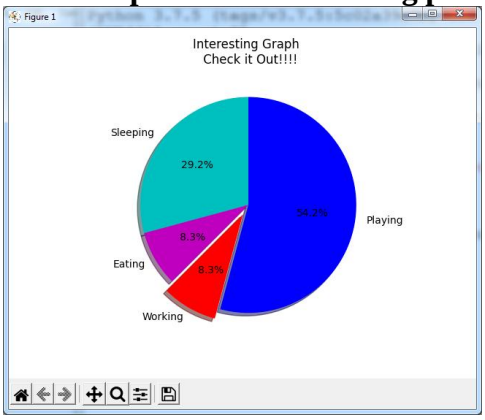
```
import matplotlib.pyplot as plt
plt.bar([1,3,5,7,9],[5,2,7,8,2], label="Example one")
plt.bar([2,4,6,8,10],[8,6,2,5,6], label="Example two",
color='g')
plt.legend()
plt.xlabel('bar number')
plt.ylabel('bar height')
plt.title('Epic Graph\nAnother Line! Whoa')
plt.show()
```



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 plot for the following pie chart output.



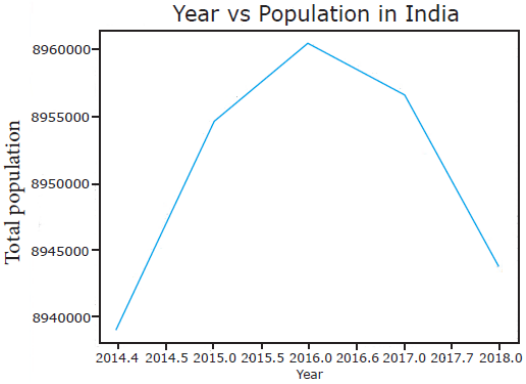

```
import matplotlib.pyplot as plt
sizes=[29.2,8.3,8.3,54.2]
labels=["Sleeping","Eating","Working","Playing"]
cols=['c','m','r','b']
plt.pie(sizes, labels=labels, colors=cols,startangle=90,
        shadow=True,explode=(0,0,0.1,0),
autopct="%1.1f%%")
plt.title("Interesting Graph \n Check it Out!!!!")
```

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

1. Explain in detail the types of pyplots using Matplotlib.

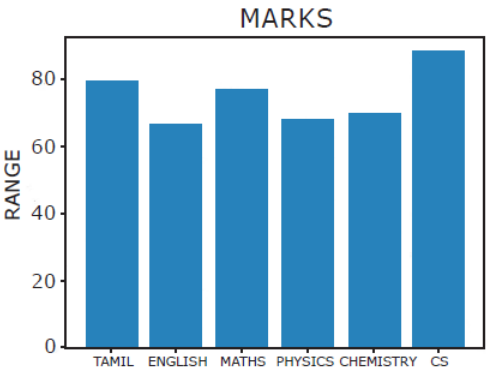
Line Chart:

- 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.
- It is often used to visualize a trend in data over intervals of time – a time series – thus the line is often drawn chronologically.



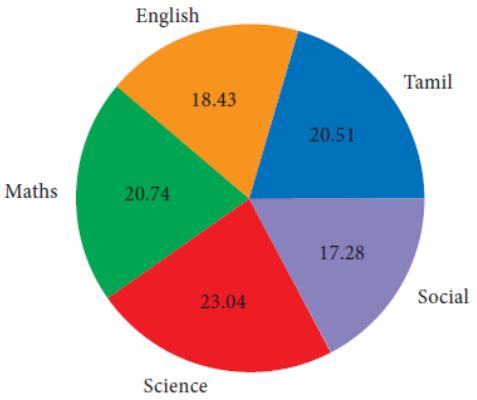
Bar Chart:

- It is one of the most common types of plot. It shows the relationship between a numerical variable and a categorical variable.
- Bar chart represents categorical data with rectangular bars.
- The bars can be plotted vertically or horizontally.
- It's useful when we want to compare a given numeric value on different categories.

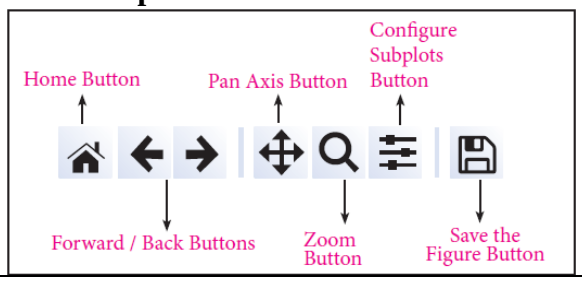


Pie Chart:

- Pie Chart is probably one of the most common types 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.



2. Explain the various buttons in a matplotlib window.



Home Button	The Home Button will help to return back to the original view.
Forward/Back buttons	These buttons can be used to move back to the previous point you were at, or forward again.
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 figure in various forms.

3. Explain the purpose of the following functions:
a. plt.xlabel b. plt.ylabel c. plt.title d. plt.legend() e. plt.show()

plt.xlabel	It is used to assign label for X-axis.
plt.ylabel	It is used to assign label for Y-axis.
plt.title	It is used to assign title for the chart.
plt.legend()	It is used to add legend for the data plotted. It is needed when more data are plotted in the chart.
plt.show()	It is used to display our plot.