Padasalai's Telegram Groups!

(தலைப்பிற்கு கீழே உள்ள லிங்கை கிளிக் செய்து குழுவில் இணையவும்!)

- Padasalai's NEWS - Group
https://t.me/joinchat/NIfCqVRBNj9hhV4wu6_NqA

- Padasalai's Channel - Group
https://t.me/padasalaichannel

- Lesson Plan - Group
https://t.me/joinchat/NIfCqVWwo5iL-21gpzrXLw

- 12th Standard - Group
https://t.me/Padasalai_12th

- 11th Standard - Group
https://t.me/Padasalai_11th

- 10th Standard - Group
https://t.me/Padasalai_10th

- 9th Standard - Group
https://t.me/Padasalai_9th

- 6th to 8th Standard - Group
https://t.me/Padasalai_6to8

- 1st to 5th Standard - Group
https://t.me/Padasalai_1to5

- TET - Group
https://t.me/Padasalai_TET

- PGTRB - Group
https://t.me/Padasalai_PGTRB

- TNPSC - Group
https://t.me/Padasalai_TNPSC

Unit – I. 2 Marks:

1. FUNCTION

1. What is a Subroutine?

Subroutines are small sections of code that are used to perform a particular task that can be used repeatedly.

2. Define function with respect to programming language.

- A function is a unit of code that is defined within a greater code structure.
- A function contains a set of code that works on many kinds of inputs, like variants, expressions and produces a concrete output.

3. Differentiate interface and implementation.

Interface	Implementation
Interface defines what an object can do, but it will not do it.	Implementation carries out the instruction defined in the interface.

2. DATA ABSTRACTION

4. What is abstract data type?

Abstract Data Type is a type for objects whose behavior is defined by a set of value and a set of operations.

5. 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.

6. What is a Pair, List and Tuple with an example.

Pairs: Any way of bundling two values together into one can be considered as a Pair. List can be called as Pairs.

Example: rational(n,d):
return[n,d]

Lists: It is constructed by placing within square brackets separated by commas. List can be store multiple values. Each value can be any type and even be another list.

Example: List[10,20]

Tuples: It is a comma separated sequence of values surrounded with parentheses.

Example: colour=('red','blue','green')

3. SCOPING

7. What is a scoping?

Scope refers to the visibility of variables, parameters and functions in one part of a program to another part of the program. i.e. any part of program can use or see it.

8. What is mapping and namespaces?

mapping: The process of binding a variable name with an object.
= (equal to) is used in programming language to map the variable and object.

namespaces: containers for mapping the names of variables to objects.
Names are mapped with objects. This allows to access the objects by names choose to assign to them.

4. ALGORITHMIC STRATEGIES

9. What is an algorithm?

- An algorithm is a finite set of instructions to achieve a particular task.
- It is step by step procedures or formula for solving a given problem.

10. Define Pseudo code & Algorist.

Pseudo code:

- It is an implementation of an algorithm in the form of annotations and informative text written in plain English .
- It has no syntax and cannot be compiled or interpreted by the computer.

Algorist:

- A person skilled in the technique of performing basic algorithm.
- A person who practices algorism is skilled in the design of algorithms.
- An algorithmic artist.

11. What is sorting and searching?

Sorting: Arranging the data in ascending or descending order.

Searching: It is the process of finding a particular data in a collection of data.

Types: (i)Linear or Sequential Search (ii) Binary Search

3 Marks: UNIT-I FUNCTION

1. Mention the Characteristics of Interface.

- The class template specifies the interfaces to enable an object to be created and operated .
- An object's attributes and behaviour is controlled by functions to the object.

2. Why Strlen is called Pure function?

- ***Pure functions are functions which will give exact result when the same arguments are passed.***
- ***Strlen(s)*** is called each time and strlen needs to iterate over the whole of 's'.
- If the compiler is smart enough to work out that strlen is pure function and that 's' is not updated in the loop.

3. Differentiate Pure and Impure Function.

Pure Function	Impure Function
1. The value of the pure functions **depends** on its arguments passed.	1. The value of the impure functions **does not depend** on its arguments passed.
2. If call the pure function with the same set of arguments, we **will get same return values.**	2. If call the impure function with the same set of arguments, we will **get different return values.**
3. They do not have any side effects.	3. For example, random( ), date( )
4, They **do not modify the arguments** which are passed to them.	4, They **may modify the arguments** which are passed to them.

4. What is side effect of impure function? Give an example.

The variables used inside the function may cause side effects though the functions which are not passed with any arguments. In such cases the function is called impure function.

For example:

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

2. DATA ABSTRACTION

5. Differentiate Concrete Data Type and Abstract Data Type.

Abstract Data Type(ADT)	Concrete Data Type(CDT)
ADT mentions what operations are to be performed but not how these operations will be implemented.	In CDT representation, a definition for each function is known.
ADT offer a high level view of a concept independent of its implementation.	CDT is direct implementations of a relatively simple concept.
ADT does not specify memory and algorithm for operation.	CDT specify memory and algorithm for operation.

6. Which strategy is used for program designing? Define the 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. SCOPING

7. Why access control is required?

- Access control is security technique that regulates who or what can view or use resources.
- It is a fundamental concept in security that minimizes risk to the object.
- It is a selective restriction of access to data.
- In Object Oriented Programming languages implemented through access modifiers.
- In C++ and Java, control the class members by public, private and protected.
- In Python, control the class members by prefixing single or double underscore in private specifiers.

4. ALGORITHMIC STRATEGIES

8. Write a note on Asymptotic notations.

Asymptotic notations are languages that uses meaningful statements about time and space complexity. There are THREE notations:

(i) Big O

Big O is often used to describes the worst-case of an algorithm.

(ii) Big Ω

Big Omega is the reverse Big O.

If Big O is used to describe the upper bound, Big Omega(**Ω**) is used to describe the lower bound.

(iii) Big Θ

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

9. What do you understand by Dynamic programming?

- Dynamic programming approach is similar to divide and conquer.
- It is used whenever problems can be divided into similar sub-problems. So that, results can be re-used to complete the process.
- It approaches are used to find the solution in optimized way.
- The solutions of overlapped sub-problems are combined in order to get the better solution.

5 Marks:

UNIT-I
FUNCTION

1. What are called parameters and write a note on

(i) Parameter without Type ***(ii) Parameter with Type***

Parameters: They are the variables in a function definition.

Arguments : They are the values which are passed to a function definition.

(i) Parameter without Type:

From the example of a function definition:

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

- In the above function definition variable 'b' is the parameter and value which is passed to the variable 'b' is the argument.
- The precondition and postcondition of the function given. Here, we have not mentioned any data types.
- Some language compiler solves this data type problems. But some requires the data type to be mentioned.

(ii) Parameter with Type:

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

- When we write the type annotations for **'a'** and **'b'** the parentheses are compulsory. We cannot leave the annotations, because it is compiler to infer them.
- There are times to want write down the data types. This is useful on times when avoid to type error at the time of compiling.
- Annotating the data types can help with debugging that error message.

2. Explain with an example Interface and Implementation.

✓ **Interface:** It is a set of action that an object can do.
For example, when you press a light switch, the light switch on. For that, don't worry about it.

✓ An Interface is a description of all functions of class must have in order to be a new interface.
For example, anything that **"ACTS LIKE"** a light, above function definitions like turn_on( ) and a turn_off( ).

✓ The purpose of interface is to allow the computer to enforce the properties of the class of TYPE T must have functions called X,Y,Z, etc.

❖ ***Implementation:***

- A class declaration combines the external interface 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.
- For example: Find the minimum of three numbers.

```
let min 3 x y z :=
if x<y then
if x<z then x else z
else
if y<z then y else z
```

The difference between Interface and Implementation:

Interface	Implementation
Interface defines what an object can do, but it will not do it.	Implementation carries out the instruction defined in the interface.

2. DATA ABSTRACTION

3. What is a List? Why list can be called as Pairs? Explain with suitable example.

❖ **List is constructed by placing expressions within square brackets separated by commas. Such an expression is called a list literal.**

❖ List can store multiple values. Each value can be of any type and can even be another list.

❖ For example, List is[10,20].

❖ Any way of bundling two values together into one can be considered as a pair.

❖ List are a common method to do so. Therefore list can be called as Pairs.

❖ The elements of a list can be accessed in two ways:

- The first way is by the method of multiple assignments.
- Example:

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

In the above example, x is 10 and y is 20.

- A second method for accessing the elements in a list is by element selection operator, also expressed using square brackets.

- Example:
  lst[0]
  10
  lst[1]
  20

❖ **Representing Rational Numbers Using List:**

- We can use the structure construct to represent multi-part objects where each part is named.
- For example,

```
class Person:
creation( )
  firstName := " "
  lastName := " "
  id := " "
  email := " "
p1 := Person( )
```

The above mentioned data type example is represented as

Person - class name
creation( ) - function belonging to the new data type
firstName, lastName, id, email - variable belonging to the new data type.
p1 - object or instance.

❖ The above class structure defines the form for multi-objects that represent a person. Same way using class we can create many objects of that type.

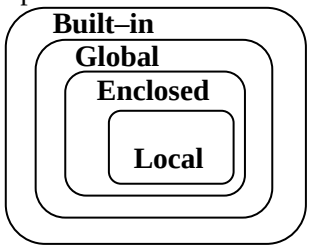
3. SCOPING

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

The LEGB rule is used to decide the order in which the scopes are to be searched for scope resolution.

There are FOUR types of Variable Scope.

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



1. Local Scope:

Local scope refers to variables defined in current function.
A function will first look up for a variable name in its local scope.

Example:

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

2. Global Scope:

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

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

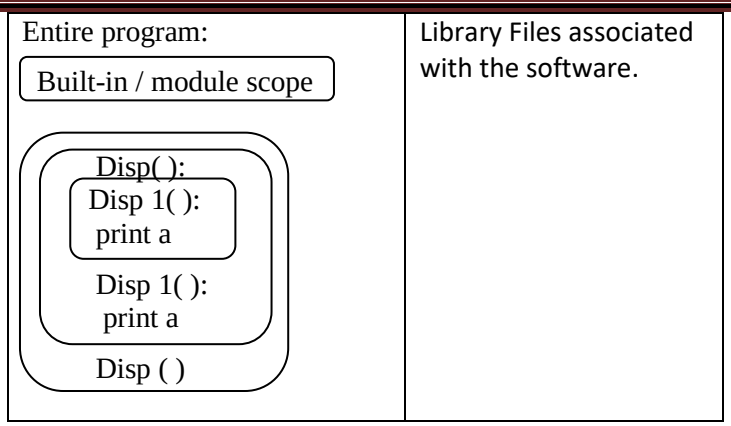
3. Enclosed Scope:

A variable which is declared inside the function which contains another function definition with in it, the inner function can access the variable of the outer function.

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

4, Built-in Scope:

The built in scope has all the names that are pre-loaded into the program when we start the compiler or interpreter.
Any variable or module which is defined in the library function has Built-in or module scope.



5. Write characteristics of the modules.

The following are the characteristics of the Modules:

- Modules contain instructions, processing logic and data.
- Modules can be separately compiled and stored in a library.
- Modules can be included in a program.
- Module segments can be used by invoking a name and some parameters and also other modules.

6. Write the Benefits in using Modular Programming.

- Less code to be written.
- A single procedure can designed more easily for reuse.
- Programs can be designed more easily to deals with small part of entire code.
- This allows many programmers to work together on the same application.
- Code is stored multiple files as short, simple and easy to understand.
- Errors can easily be identified.
- The same code can be used many applications.
- The scoping variables can be controlled easily.

4. ALGORITHMIC STRATEGIES

7. Explain the characteristics of an algorithm.

1. **Input** : Zero or more quantities to be supplied.
2. **Output** : At least one quantity is produced.
3. **Finiteness** : Algorithms terminated after finite number of steps.
4. **Definiteness** : All operations should be well defined. For example, division by zero or square for negative number are unacceptable.
5. **Effectiveness** : Every instruction must be carried out effectively.
6. **Correctness** : The algorithms should be error free.
7. **Simplicity** : Easy to implement.
8. **Unambiguous (Unmistakable)** : Algorithm should be clear and unmistakable. Each steps should be clear and it gives one meaning.
9. **Feasibility** : Algorithm should be possible with available resources.
10. **Portable and independent** : Algorithm should be generic, independent and also able to handle all range of inputs.

8. Explain in detail about Linear Search algorithm.

Linear Search (Sequential Search):

- It sequential method for finding particular value in a list.
- This method checks the search element in sequence until the desired element is found or the list is executed.
- **Pseudo code:**
  - Go through the array using for loop in the sequential order.
  - In every iteration, compare the target search key value with the current value.
  - 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.
  - If no match is found, display the search element not found.
- **For example,**
  - To search the number 25, in the array.
  - Linear search will go in sequential order start from first element.
  - If the search element is found, index is returned.
  - If not found the element, 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

Example 1:

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

Output: 4

Example 2:

Input: values[ ] = {101, 392, 1, 54, 32, 90}
Target = 200

Output: -1(not found).

9. Explain the Binary Search with example.

Binary Search (Half-Interval Search) :

- It finds the position of a search element within a sorted array.
- This method can be done as divide and get the better value search algorithm and executes in logarithmic time.
- **Pseudo Code:**
  - Start with the middle element in the array.
  - If the search element is equal to the middle element, returned to index of the middle element. (middle value = no. of elements / 2)
  - If not, then compare middle element with the search value.
  - If the search value is greater than the middle index value, then select the elements to the right side of the middle index and start with step 1.
  - If search value is less than the middle index value, then select the elements to the left side of the middle index and start with step 1.
  - When match is found, then display the success message.
  - If no match is found for all comparisons, then display the unsuccessful message.
- **For example,**
  - List of elements in an array must be sorted first for Binary search.
  - The search element is 60 then search the location or index of element.

10	20	30	40	50	60	70	80	90	99
0	1	2	3	4	5	6	7	8	9

  - First, we find index of middle element of the array by,
    mid value = low + (high – low)/2 = 0 + (9 – 0)/2 = 9/2 = 4.5 = 4
  - Now compare with search element(60) with mid value(50), which is not match with search element.
  - Now we change the low value as mid value + 1 and find the new mid value again using the formula.
  - low value = mid value + 1 = 4 + 1 = 5
  - mid value = low + (high – low)/2 = 5 + (9 – 5)/2 =5 + 4/2 = 5 + 2= 7
  - Now mid value is 7. We compare the value with search value.

10	20	30	40	50	60	70	80	90	99
0	1	2	3	4	5	6	7	8	9

  - The search element is still not found. We calculate the mid value again by using formula. Search value is less than the mid value. For this, process can be done in the reverse direction.
  - high value = mid value – 1 = 7 – 1 = 6
  - mid value = low + (high – low)/2 = 5 + (6 – 5)/2 = 5 + 0.5 = 5.5 = 5

10	20	30	40	50	60	70	80	90	99
0	1	2	3	4	5	6	7	8	9

  - Now, we compare the value stored with location 5, we found that it is a match.
  - We can conclude that the search element 60 is found at location or index 5. If we search value as 95, it returns as unsuccessful.

10. Explain the Bubble Sort in detail with example.

Bubble Sort Algorithm:

- Bubble Sort is a simple sort 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 .

Pseudo Code:

- Start with first element, i.e., compare the current element with the next element of the array.
- If the current element is greater than the next element of the array, swap them.
- If the current element is less than the next or right side of the element, move the next element.
- Go to step 1 and repeat until end of the index is reached.
- **For example:**
- Consider an array with the values {15, 11, 16, 12, 14, 13}, Following representation gives the bubble sort of the given array.

15 > 11
So Interchange

15	11	16	12	14	13

→ 1

16 > 15
So No Swapping

11	15	16	12	14	13

→ 2

16 > 12
So Interchange

11	15	12	16	14	13

→ 3

16 > 14
So Interchange

11	15	12	16	14	13

→ 4

16 > 13
So Interchange

11	15	12	14	16	13

→ 5

Do the Same Process From Step 1

11	15	12	14	13	16

→ 6

- At the end of all iterations it will give the sorted values in an array as given below.

11	12	13	14	15	16

Unit – II
5. VARIABLES AND OPERATIONS

2 MARKS:

1. What are the different modes that can be used to text Python Programming?

- In Python, Programs can be written in the two ways:
  (i) Interactive Mode (ii) Script Mode
- **Interactive Mode:**
- It allows to write codes in command prompt(>>>)
- Python code directly typed and interpreter displays the results immediately. It act as simple calculator.
- **Script Mode:**
- It can be written and stored as separate file with the extension as **.py** and executed.
- It is used to create and edit python source file.

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

- Literal is a raw data given in a variable or constant.
- In Python, there are various types of literals.
- **Numeric Literals:** It consists of digits and are immutable(unchangeable)
- **String Literals :** It is a sequence of characters surrounded by quotes.
- **Boolean Literals :** It can have any of the two values as results: True or False.

6. CONTROL STRUCTURES

3. List the control structures in Python.

The control structures in Python are:

- Sequential
- Alternative or Branching
- Iterative or Looping

4. Define Control Structures.

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 a note on range( ) in loop.

- ***range( )*** generates a list of value starting from start till stop -1.
- The syntax is : **range(start, stop, [step])** where
  - **start** – refers as initial value.
  - **stop** – refers as final value.
  - **step** – refers as increment value, i.e., optional.

For example:

range(1,30,2) ➔ start the range values from 1 and end at 28 as increment of 2.

7. PYTHON FUNCTIONS

6. What is Function and its types?

Functions are named blocks of code that are designed to do specific job.
Types of Python Functions are:

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

7. What are the main advantages of function?

Main advantages of functions are:

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

8. 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.

8. STRINGS AND STRING MANIPULATIONS

9. What is String?

- String is 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.
- **Example:**
  - 'Welcome to learning Python'
  - "Welcome to learning Python"
  - "' "Welcome to learning Python" '"

10. How can you modify and delete a string a Python?

- **Modification:**
- Strings in Python are immutable.
- Once we define a string modifications or deletion is not allowed. For modify the string define as a new string.
- **For example:**
- str1="How are you",
- In this above string can be modified as
- str1="How about you"
- **Deletion:**
- We can remove entire string variable using **del** command.

- **For example:**
- str1 = "How are you"
- del str1
- Delete the string str1 now.

11. What is slicing?

- ★ Slice is a substring of a main string.
- ★ A substring can be taken from the original string by using [ ] operator.
- ★ [ ] is called as Slicing Operator.
  - ♦ **Syntax:** str[start : end]
  - ♦ **Example:**
  - ♦ str1 = "SCHOOL"
  - ♦ print(str1[0:3])
  - ♦ **output:** SCH

Unit – II

5. VARIABLES AND OPERATIONS

3 MARKS:

1. What are the Assignment Operators can be used in Python?

- ★ In Python, **=** is a simple assignment operator to assign the values to variables.
- ★ **Example:**
- ★ a = 5 and b = 10
- ★ Value 5 to a, Value 10 to b
- ★ It can assigned as a,b = 5,10
- ★ Various assignment operators are like as **+=, –=, *=, /=, %=**

2. Define Ternary Operator with example.

- ♦ Ternary operator is also called as conditional operator.
- ♦ It evaluates based on a condition is true or false.
- ♦ It allows testing a condition in a single line by replacing multiline if-else code.
- ♦ **Syntax:** Variable Name = [on_true] if [test expression] else [on_false]
- ♦ **Example:**
- ♦ (i) min = 50 if 49 < 50 else 70 output value is min = 50
- ♦ (ii) min = 50 if 49 > 50 else 70 output value is min = 70

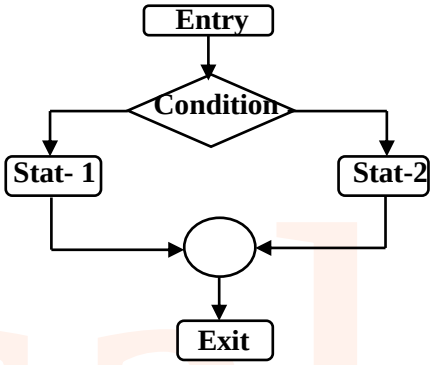
3. Write the escape sequences with examples.

- ★ In Python, the backslash **" \"** is a special character, also called as **"escape"** character.
- ★ It is used like as **" \n"** is a new line, **" \t"** is a tab.
- ★ Example: >>>print("It **\'**s raining") Output: It's raining.

6. CONTROL STRUCTURES

4. Write a note on if…else structure.

- The **if…else** statement provides control to check the true block and also the false block.
- **Syntax:**
  - if <condition>:
    statements – block-1
    else
    statements – block-2
- **Flow chart – if…else statement:**



. Example:

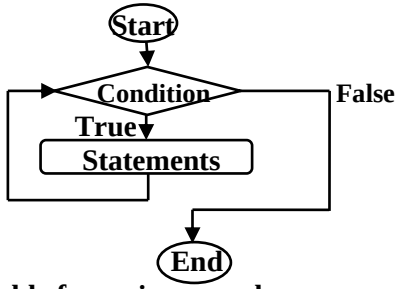
```
a = int (input("Enter value a:"))
if a%2 = = 0:
    print(a, "is even")
else:
    print(a, "is odd")
```

5. Write the difference between break and continue statements.

Break	Continue
1. Break statement terminates the current loop.	1. Continue statement is used to skip the remaining part of the loop.
2. Control of the program flows to the statement immediately comes out of the body of the loop.	2. But it starts with next iteration.

6. Write the syntax of while loop.

```
while <condition>:
    statement – 1
[else:
    statement – 2]
```



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

```
num = int (input("Enter multiplication table Number:")
for i in range (1,16):
print(i, 'x', num, '=' , num * i)
```

7. PYTHON FUNCTIONS

8. Write the rules of local variable.

- ★ A variable with local scope can be accessed only within the function / block.
- ★ A variable is created inside the function / block is becomes local to it.
- ★ A local variable only exists while the function is executing.
- ★ The formate arguments are also local to function.

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

- ★ When we define a variable outside the function, it's global by default. We don't use as global keyword.
- ★ We use global keyword to read and write a global variable inside a function.
- ★ Use of global keyword outside the function has no effect.

10. Write the difference between ceil( ) and floor( ).

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

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

When defining a function following things to be noted:

- ★ Function blocks begin with the keyword **"def"** followed by function name and parenthesis( ).
- ★ Any input parameters or arguments should be placed within these parentheses when define a function.
- ★ The code block always comes after a colon(:).
- ★ The statement **"return"** exits a function. A return statement without argument is same value can be return.

8. STRINGS AND STRING MANIPULATION

12. Write a short note on the following: (a) capitalize( ) (b) swapcase( )

(a) capitalize( ) :

It is used to capitalize the first character of the string.

Example:

```
>>>city="chennai"
>>>print(city.capitalize( ))      output: Chennai
```

(b)swapcase( ):

It is used change the case of every character to its opposite case and vice-versa.

Example:

```
>>>str = "tAmilNaDu"
>>>print(str.swapcase( ))          output: TaMILnAdU
```

13. What is the use of format( ) ? Given an example.

- ❖ The format( ) function used with strings is very flexible and powerful function used for formatting strings.
- ❖ The curly braces { } are used as placeholders which get replaced along with format( ) function.
- ❖ **Example:**

```
n1=int(input("Number 1: "))
n2=int(input("Number 2: "))
print("The sum of { } and { } is { }".format ( n1, n2,(n1 + n2)))
```

- ❖ **output:**

```
Number 1: 34
Number 2: 54
The sum of 34 and 54 is 88\
```

14. Write a note on count( ) function with example in Python.

- ★ Returns the number of substrings occurs within the given range.
- ★ Substring may be a single character.
- ★ Range arguments optional.
- ★ Search is case sensitive.

Example

```
>>>str1 = "Raja Rajan"
>>>print(str1.count('Raj'))
    2
>>>print(str1.count('r'))
    0
>>>print(str1.count('R'))
    2
```

5. VARIABLES AND OPERATORS

5 MARKS:

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

- **Input( ):**

  The **input( )** function helps to enter the data and access the data at run time.

- **Syntax:**
  - **Variable = input ("prompt string")**
  - **prompt string** is a statement or message to the user.
  - **promt string** is used to display the data on the monitor.

- The **input( )** displays whatever is typed by the user.
- If prompt string is not given, the input( ) no message displayed on the screen.
- **Example 1:**
  - >>>city =input("Enter the city:")
  - Enter the city: Madurai
- **Example 2:**
  - >>>city = input( )
  - Rajarajan
- The **input( )** accepts all data as string characters but not as members.
- The **int( )** function is used to convert string data as integer data explicitly.

print( ):

- In Python, the **print( )** is used to display result on the screen.
- **Syntax:**
- print ("string to be displayed as output")
- print(variable)
- print("string 1", variable, "string 2", variable, ……)
- **Example 1:**
  - >>>print("Welcome to Python Programming…")
    output: Welcome to Python Programming…
- **Example 2:**
  - >>>x = 5, y = 10
  - >>>z=x + y
  - >>>print ("The sum of ", x, "and", y, "is", z)
  - The sum of 5 and 10 is 15
- The **print( )** evaluates the expression before printing it on the monitor.
- The **print( )** displays an entire statement which is specified within print( ).
- **Comma(,)** is used as a separator in **print( )** to print more than one item.

2. Discuss in detail about Tokens in Python.

- Python breaks each logical line into a sequence of elementary lexical components known as **Tokens.** There are various types:
  - Identifiers
  - Keywords
  - Operators
  - Delimiters and
  - Literals
- **IDENTIFIERS:**
  - An identifiers is a name used to identify a variable, function, class, module or object.
  - An identifiers must start with an alphabet or underscore(_) and it contains numbers also.
  - Python identifiers are case sensitive and do not allow punctuations.
  - **Example:** sum, num_1
- **KEYWORDS:**
  - **Keywords** are special words used by Python interpreter to recognize the program structure.
  - They cannot be used for any other purpose.
- **Example:** pass, break
- **OPERATORS:**
  - **Operators** are special symbols with represent the computations, conditions etc.
  - **Operators** are categorized as Arithmetic, Relational, Logical, Assignment values and variables are used with operator are called operands.
- **DELIMETERS:**
  - Python uses the symbol combinations as delimiters in expressions, lists and strings.
- **Example: ( ), [ ], { }, :, =** etc.
- **LITERALS:**
  - **Literal** is a raw data given in a variable or constant.
  - In Python, there are various types of Literals.
  - **Numeric:** Numeric Literals consists of digits and are immutable.
  - **String:** In Python, a string literal is sequence of characters surrounded by quotes.
  - **Boolean:** A Boolean literal can have any of the two values: True or False.
  - **Examples:**
    - **Numeric:** b=100
    - **String:** strings="This is Python"
    - **Boolean:** boolean_1 = True, Boolean_2 = False.

6. CONTROL STRUCTURES

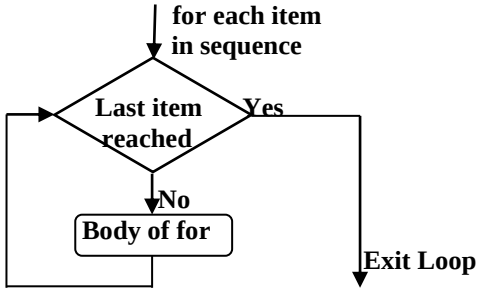
3. Write detail note on for loop.

★ **for loop** is entry check loop.
★ The condition is checked in the beginning and the body of the loop is executed if it is True otherwise the loop is not executed.
★ **Syntax:**
  - for counter_variable in sequence:
    - statement 1
  - [else:
    - statement 2]
★ The *counter_variable* is control variable of the loop.
★ *sequence* refers to the initial, final and increment values.

★ In Python, **for loop** uses the *range( )* in the sequence. *range( )* generates a list of values starting from **start** till **stop-1.**

★ **Syntax of range:**
- range(start, stop, [step])
- where, start → is initial value, stop → is final value, step→is increment value. This is optional part.

★ **Flow Chart:**



★ **Example:**

```
a = ['A', 'B', 'C', 'D', 'E']
for i in range(0,6):
for j in range(0, i):
  print(a[j], end=" ")
else:
  print( )
```

4. Write in detail about if..elif…else statement with example.

❖ When we need to construct a chain of **if** statements then **'elif '** clause can be used instead of **'else'**.

❖ **Syntax:**

```
if <condition>:
   statement 1
elif <condition>:
   statement 2
else:
   statement n
```

❖ In the above, syntax of **if..elif..else**, condition-1 is tested if it is true then statement 1 is executed, otherwise the control checks condition-2, if it is true statement 2 is executed and if it fails statement n mentioned in **else** is executed.

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

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

❖ **Example:**

```
n1= float(input("Enter first number :"))
n2=float(input("Enter second number :"))
n3=float(input("Enter third number :"))
if (n2 >= n2) and (n1 >= n3):
   biggest =n1
elif (n2 >= n1) and (n2 >= n3):
   biggest = n2
else:
   biggest = n3
print("The biggest number is : ",biggest)
```

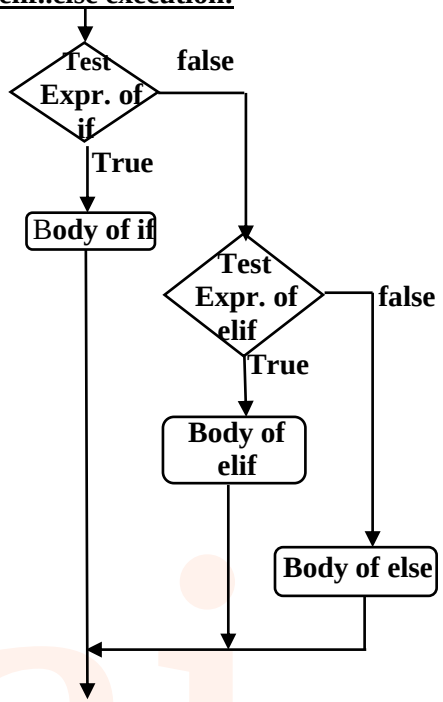
❖ **Output 1:**

Enter the first number : 20
Enter the second number : 10
Enter the third number : 30
The biggest number is : 30

❖ **Output 2:**

Enter the first number : 25
Enter the second number : 20
Enter the third number : 15
The biggest number is : 25

if..elif..else execution:



7. PYTHON FUNCTIONS

5. Different types of functions with example.

Types of Functions:

We can divide the functions into the following types:

★ **User–defined functions**
★ **Built-in functions**
★ **Lambda functions**
★ **Recursion functions**

(i) User–defined functions:

Functions defined by the users themselves.

Syntax:

```
def <function_name ([parameter 1, parameter 2, ...])>:
      <Block of Statements>
return <expression / none>
```

Example:

```
def printinfo(name, salary = 3500)
print ("Name:", name)
print("Salary:",salary)
return
printinfo("MANI")
```

output:
Name: MANI
Salary: 3500

(ii) Built-in Functions:

Functions that are inbuilt within python are called built-in functions.

Example:

(i) abs ( ) : returns an absolute of a number. The arguments may be integer or float.

```
abs(x):
  x= 20, y = -23.2
 print('x=', abs(x))
 print('y=',abs(y))
```

output:

x = 20 y= 23.2

(iii) Lamda functions:

1. Functions that are anonymous un-named function are called as Lamda functions.
2. In Python, anonymous functions is a function that is defined without a name.
3. These Lamda functions defined using the lamda keyword.

Syntax:

Lamda[argument(s)]:expression

Example:

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

Output:

The sum is: 70
The sum is: 10

(iv) Recursion functions:

Functions that calls itself is known as recursive function.

Example:

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

Output:

1
120

6. Explain the Scope of variables with an example.

Scope of the variable refers to the part of the program, where it is accessible.
i.e., area where you can use it.

There are TWO type of Scopes: **(i) Local Scope (ii) Global Scope**

(i) Local Scope:

A variable declared with inside the function's body or in the local scope is called as local variables.

Rules of local variables:

★ A variable with local scope can be accessed only within the function / block.
★ A variable is created inside the function / block is becomes local to it.
★ A local variable only exists while the function is executing.
★ The formate arguments are also local to function.

Example:

```
def loc( ):
y=2   #local scope
print(y)
loc( )
```

Output:

2

Global Scope:

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

Rules of Global variable:

★ When we define a variable outside the function, it's global by default. We don't use as global keyword.
★ We use global keyword to read and write a global variable inside a function.
★ Use of global keyword outside the function has no effect.

Example:

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

Output:

Inside the add( ) function x value is : 7
In main x value is : 5

8. STRINGS AND STRING MANIPULATION

7. Explain about string operators in Python with suitable example.

Python provides the following operators for string operations.

(i) Concatenation(+)

Joining of two or more strings is called as Concatenation. The plus(+) operator is used to concatenate string in Python.

Example:

>>>"Welcome" + "Python" **o/p:** ***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"
>>>str1+="Learn Python"
>>>print(str1)
```

o/p: 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
```

(iv) String Slicing:

Slice is a substring of a main string. A substring can be taken from the original by using [ ] operator. [ ] is a Slicing Operator. Using operator, slice one or more substrings from a main string.

General format:

str[start:end]

Example:

Slice a substring from index 0 to 4

```
>>>str1= "COMPUTER"
>>>print(str1[0:5])
    COMPU
```

(v) Stride when slicing string:

When the slice, specify a third argument as the stride. The default value of stride is 1.

Example:

```
>>>str1= "Welcome to Learn Python"
>>>print(str1[10:16:2])
```

Output: er

UNIT – III

9. LISTS, TUPLES, SETS AND DICTIONARY

2 Marks:

1. What is Lists? How will access the list elements in reverse order?

Lists: A list is called 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. It can be any type as numbers, characters, strings and nested lists.

Reverse Order: Python enables reverse or negative indexing for the list elements. 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.

2. Differentiate del and remove( ) in the List?

del	remove
del statement is used to delete known elements.	remove( ) function is used to delete elements of a list if its index is unknown.
The del statement can be used to delete entire list.	The remove( ) function used to delete an element by the given index value.
Example: >>>mysub=['Tamil', 'Hindi', 'Telgu'] >>>del mysubjects[2] >>>print(mysubjects) ['Tamil', 'Telgu']	Example: >>>mysub=['Tamil', 'Hindi', 'Telgu'] >>>mysub.remove('Hindi') >>>print(mysubjects) ['Tamil', 'Telgu']

10. PYTHON CLASSES AND OBJECTS

3. What is class and instantiation?

class:

1. class is the main building block in Python.
2. object is a collection of data and function that act on those data.
3. class is a template for the object.

Instantiation:

1. A class is created, next create an object or instance of the class.
2. The process of creating object is called as "class instantiation".

4. What is the purpose of destructor?

- Destructor is a gets executed automatically when an object exit from the scope.
- It is used removes the memory of the object.
- In Python, _ _del_ _( ) method is used as destructor.

5. How will you create constructor?

★ Constructor is executed automatically when an object is created.
★ It used to initialize the class variables.
★ In Python, _ _init_ _ ( ) method is used as constructor.

9. LISTS, TUPLES, SETS AND DICTIONARY

3 Marks:

1. What are advantages of tuples over a list?

★ The elements of a tuple are unchangeable(immutable)
The elements of a list are changeable(mutable)
★ The elements of a tuple are enclosed by paranthesis.
The elements of a list are enclosed by square brackets.
★ Tuple is faster than list.

2. Define sort( ) with an example.

sort( ): Sorts the elements in the list.

syntax: List.sort(reverse=True | False, key=myFunc)

example:

```
MyList=['Thilagar', 'Uvan', 'Ajay', 'Saran', 'Raja', 'David']
MyList.sort( )
print(MyList)
MyList.sort(reverse=True)
print(MyList)
```

output:

['Ajay', 'David', 'Raja', 'Saran', 'Thilagar', 'Uvan']
['Uvan', 'Thilagar', 'Saran', 'Raja', 'David', 'Ajay']

3. Differentiate List and Dictionary.

List	Dictionary
List is an ordered set of elements.	Dictionary is a data structure used for matching one element with another.
The index values can be used to access a particular element.	Dictionary key represents as index. The key may be a number of a string.
Lists are used to look up a value.	Dictionary is used to take one value and look up another value.

4. Difference between del and clear( ) with example.

del	clear( )
del keyword is used to delete the particular element and also used for remove entire dictionary.	clear( ) function is used to delete all elements.
syntax: del dictionary_name[key] del dictionary_name	**syntax:** dictionary_name.clear( )
Example: del Dict['Mark1'] – to delete a particular element. del Dict – to delete entire dictionary.	**Example:** Dict.clear( ) – to delete all elements in the dictionary.

10. PYTHON CLASSES AND OBJECTS

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

★ Variables defined inside a class are called as "Class Variable" and functions are called as "Methods".
★ Class variables and Methods are called class members.
★ The class members should be accessed through objects or instance of class.
★ Any class members can be accessed by using object with a dot(.) operator.
★ **Example:**

```
class student:
    m1, m2, m3 = 45, 91, 71
    def process(self):
        sum=student.m1 + student.m2 + student.m3
        print("Total Marks = ", sum)
        return
    s=student( )
    s.process( )
```

6. How do define constructor and destructor in Python?

Constructor	Destructor
Constructor automatically executed when an object of a class is created.	Destructor automatically executed when an object exit from the scope.
It is used to initialize the class variables.	It is used removes the memory of the object.
_ _ init _ _ ( ) method is used as constructor.	_ _ del _ _( ) method is used as a destructor.
Syntax: def _ _ init _ _ (self, [arguments]): <statements>	**Syntax:** def _ _ del _ _(self):

9. LISTS, TUPLES, SETS AND DICTIONARY

5 Marks:

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

In Python, adding elements in following ways.

- append( ) method
- extend( ) method
- insert ( ) method

append( ) method:

It is used to add a single element in a list. In existing List, value added as a last element.

Syntax:

List.append(element value to be added)

Example:

```
>>>mylist=[34, 45, 47]
>>>mylist.append(90)
>>>print(mylist)
        [34, 45, 47, 90]
```

Mylist.append(90) statement add value 90 with the existing list in last.

extend( ) method:

It is used to add more elements to an existing list. In the List, values added in the last elements with existing elements.

Syntax:

List.extend([elements to be added])

Example:

```
>>>mylist=[34,45,47]
>>>mylist.extend([71, 32, 29])
>>>print(mylist)
        [34,45,47,71,32,29]
```

The print statement shows all elements of the list after the inclusion of additional elements.

insert ( ) method:

The insert ( ) function is used to insert an element at any position of a list. That is, we want include an element at the desired position.

Syntax:

List.insert(position index, element)

Example:

```
>>>mylist=[34,45, 'Raj', 'Kumar', 55]
>>>mylist.insert(3, 'Lenin')
>>>print(mylist)
        [34,45, 'Raj', 'Lenin', 'Kumar', 55]
```

2. What is Nested Tuple? Explain with an example.

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

For example:

```
Toppers=(("Raj", "XII-G", 98.7), ("Kumar", "XII-F", 93.5), ("Sai", "XII-G", 91.7))
for i in Toppers:
    print(i)
```

output:

("Raj", "XII-G", 98.7)
("Kumar", "XII-F", 93.5)
("Sai", "XII-G", 91.7)

All the functions used in List can be applicable even for tuples.

3. What is set? Explain the different set operations supported by Python with suitable example.

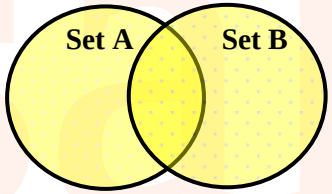
In Python, a Set is another type of collection data type.

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

The Python, supports the set operations as follows:

(i) Union (ii) Intersection (iii) Difference (iv) Symmetric difference

(i) Union:



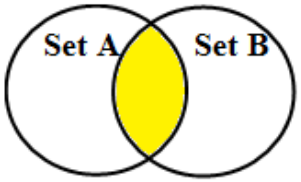
★ It includes all elements from two or more sets.
★ The function **Union ( )** is used to join two sets in Python.
★ The operator | is used to Union of two sets. Also, used the keyword 'union'.
★ **Example:**

```
setA= {2,4,6,8}      setB= {'A', 'B', 'C', 'D'}
setU= setA | setB    or setU=setA . union(setB)
print(setU)
```

output:

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

(ii) Intersection:



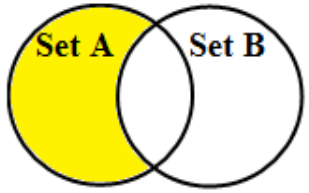
★ It includes the common elements in two sets.
★ The function **intersection( )** is used to intersect two sets in Python.
★ The operator **&** is used to Intersect two sets. Also, used the keyword 'intersection'
★ **Example:**

```
setA= {'A', 4, 6, 'D'}      setB= {'A', 'B', 'C', 'D'}
setU= setA | setB    or setU=setA . intersection(setB)
print(setU)
```

output:

{'A', 'D'}

(iii) Difference:



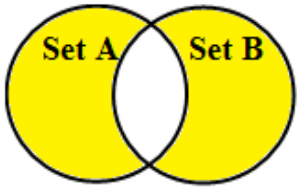
★ It includes all elements that are in first set but not in the second set.
★ The function **difference ( )** is used to difference operation.
★ The operator **–** is used to difference the sets. Also used the keyword 'difference' to difference the set.
★ **Example:**

```
setA= {'A', 4, 6, 'D'}      setB= {'A', 'B', 'C', 'D'}
setU= setA & setB    or setU=setA . difference(setB)
print(setU)
```

output:

{4, 6}

(iv) Symmetric Difference:



★ It includes all elements which are difference between two sets. But not in one set that are difference between two sets.
★ The function **symmetric_difference ( )** is used to do difference between the two sets.
★ The operator **^** is used to symmetric difference between the sets. Also used the keyword 'symmetric_difference' is used to difference the sets.
★ **Example:**

```
setA= {'A', 4, 6, 'D'}      setB= {'A', 'B', 'C', 'D'}
setU= setA ^ setB    or setU= setA . symmetric_difference (setB)
print(setU)
```

output:

{4, 6, 'B', 'C'}

Programs:

1. Write a program to display as

A
A B
A B C
A B C D
A B C D E

```
a = ['A', 'B', 'C', 'D', 'E']
for i in range (0,6):
        for j in range (0, i):
            print(a[j], end = " ")
        else:
            print( )
```

output:

A
A B
A B C
A B C D
A B C D E

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

```
num = int(input("Display Multiplication table of ? :"))
  for i in range (1, 6):
    print(i, 'x', num, '=', num * i)
    # print(num, 'x', i, '=', num * i)
```

Output:

Display Multiplication table of ?: 3
1 x 3 = 3
2 x 3 = 6
3 x 3 = 9
4 x 3 = 12
5 x 3 = 15

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

```
n1=float(input("Enter first number: "))
n2=float(input("Enter second number: "))
n3=float(input("Enter third number: "))
    if (n1> = n2) and (n1 > = n3):
            biggest = n1
    elif(n2 > = n3) and (n2 > = n1):
            biggest = n2
    else:
            biggest = n3
print ("The biggest number between three numbers is : ", biggest)
```

output:

```
Enter first number: 25
Enter second number: 55
Enter third number: 40
The biggest number between three numbers is : 55
```

4. Write a program to display all 3 digit odd number.

```
lower = int(input("Enter the lower limit for the range: "))
upper = int(input("Enter the upper limit for the range: "))
   for i in range (lower, upper + 1):
     if(i % 2 != 0):
        print(i, end = " ")
```

output:

```
Enter the lower limit for the range: 101
Enter the upper limit for the range: 110
101  103  105  107  109
```

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

```
def leap_year(y):
    if (y % 400 = = 0):
       print(y, "is the leap year")
    elif(y%4 = = 0):
       print(y, "is the leap year")
    else:
       print(y, "is not a leap year")
year = int(input("Enter a year…")
print(leap_year(year))
```

output:

```
Enter a year…  2007
2007 is the leap year
```

6. Write a Python code to find the L. C. M. 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
    n1 = int (input("Enter the first number: "))
    n2 = int(input("Enter the second number: "))
    print("The L.C.M. is", lcm(n1, n2))
```

output:

```
Enter the first number: 100
Enter the second number: 102
The L.C.M. is 5100
```

7. Write a Python program to display the given pattern:

C O M P U T E R
C O M P U T E
C O M P U T
C O M P U
C O M P
C O M
C O
C

```
str1 = "C O M P U T E R"
index = len(str1)
for i in str1:
    print(str1[:index]
    index - = 1
```

output:

C O M P U T E R
C O M P U T E
C O M P U T
C O M P U
C O M P
C O M
C O
C

11. DATABASE CONSEPTS

2 Marks:

1. What is Data Consistency?

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

2. What is normalization?

Normalization reduces data redundancy and improves data integrity in RDBMS. But not performed in DBMS.

3. Mention examples of RDBMS and Database.

Database: Dbase, FoxPro. **RDBMS:** SQL, Oracle, mysql, MariaDB, SQLite.

4. Define Relational Algebra.

Relational Algebra is a procedural query language used to query the database tables using SQL. Relational Algebra was first created by Edgar F Codd in 1970.

3 Marks:

1. Write note on different types of DBMS users.

Database Administrators(DBA): is manages the complete DBMS. DBA take care the security of DBMS, Managing the license keys, user accounts and access etc.

Application Programmers or Software Developers: is involved in developing and designing of DBMS.

End User: This user can collect user data and store the data on DBMS systems running on the server. End users are the store, retrieve, update and delete data.

Database Designers: are responsible for identifying the data to be stored in the database for choosing structures to represent and store the data.

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

Select: The Select operation is used for selecting subset with tuples according to a given condition. Select filters out all tuples that do not satisty C.

For example: σ_{course} **= "Big Data"(STUDENT)**

Project: The projection elimination 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.

For example: Π_{course}**(STUDENT)**

3. Explain Cartesian Product with a suitable example.

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

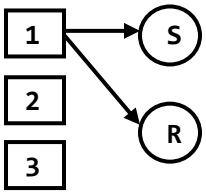


Table A = 3
Table B = 2
Table A x B = 3 x 2 = 6

4. Explain the various Relational operations with example.

Various Relational Algebra operations from the set theory.

- **UNION(∪):**It includes all tuples that are in tables A or in B. It eliminates duplicates. Set A Union Set B is expressed as **A ∪ B.**
- **INTERSECTION(∩):** Defines a relation consisting of a set of all tuple that are in both A and B. Set A Intersection Set B is expressed as **A ∩ B.**
- **SET DIFFERENCE(–):** 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.
- **PRODUCT OR CARTESIAN PRODUCT(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.

12. STRUCTURED QUERY LANGUAGE

2 MARKS:

1. Differentiate Unique and Primary Key Constraint.

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

2. What is the difference between SQL and MySQL?

SQL is a language that helps to create and operate relational databases.
MySQL is a database management system which manages the tables in DBMS.

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

COLUMN CONSTRAINT	TABLE CONSTRAINT
This constraint applied only to individual column.	This constraint applied to a group of one or more columns.
It applied only on the column. It given at the end of the column definition.	It applied to a group of fields of the table. It given at the end of table definition.

4. What is constraint?

- Constraints are used to limit the type of data into a table.
- This gives accuracy and reliability of the data in the database.
- Constraints could either column or table level.

3 MARKS:

1. Write a SQL statement using DISTINCT keyword.

- The **DISTINCT** keyword is used along with the **SELECT** command to eliminate duplicate rows in the table.
- This helps to eliminate unnecessary data.
- For example, **select DISTINCT Place FROM Student;**

2. Write the use of SAVEPOINT command with an example.

- The **SAVEPOINT** command is used to temporarily save a transaction. ROLLBACK is used whenever required.
- Syntax: **SAVEPOINT savepoint_name;**
- **UPDATE Student SET Name = 'Mini' WHERE Admno=105; SAVEPOINT A;**

13. PYTHON AND CSV FILES

2 MARKS:

1. What is CSV file?

- A **CSV(Comma Separated Values)** file is a human readable text file where each line has a number of fields, separated by commas or some other delimiter.

2. What is the use of next()function?

- When we sort the data, the row heading is also sorted.
- To avoid that the first row should be skipped.
- This can be done by next() command.

3. Mention the two ways to read a CSV file.

There are two ways to read a CSV file.

1. Use the CSV module's reader function.
2. Use the DictReader class.

4. How will you sort more than one column from a CSV file? Give an example.

To sort by more than one column can use itemgetter with multiple index as operator.itemgetter.
For example: sortedlist=sorted(data,key=operator.itemgetter(1)).

3 MARKS:

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

- **Making some changes in the data of the existing file or adding more data is called modification.**

For example:

```
import csv
row=['3', 'Meena', 'Bangalore']
with open ('student.csv', 'r') as rf:
reader=csv.reader(rf)
lines=line(reader)
lines[3]=row
with open('student.csv', 'w') as wf:
writer=csv.writer(wf)
writer.writerows(lines)
rf.close()
wf.close()
```

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

CSV Reader()	DictReader()
CSV reader and CSV Writer work with **list/tuple.**	DictReader and DictWriter work with dictionary.
It takes default delimiter as separated by **comma(,).**	It takes additional argument fieldnames used as **dictionary keys.**
For example: ItemName, Quantity 'Keyboard', '48'	For example: 'ItemName': 'Keyboard', 'Quantity': '48'

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

The following program read a file with default delimiter comma(,):

```
import csv
with open('d:\\XII\\sample1.csv', 'r') as f:
reader=csv.reader(f)
for r in reader:
   print(r)
f.close()
```

OUTPUT:

```
['SNO', 'NAME', 'CITY']
['12101', 'RAM', 'CHENNAI']
['12102', 'KUMAR', 'TRICHY']
['12103', 'LAVANYA', 'MADURAI']
```

14. IMPORTING C++ PROGRAMS IN PYTHON

2 MARKS:

1. Write the difference between Scripting language and other programming language.

SCRIPTING LANGUAGE	OTHER PROGRAMMING LANGUAGE
Scripting language do not requires an interpreter.	Programming language requires compiler.
It do not require compilation step and needs interpreted.	It requires compilation step and needs compiled.
For example: Python, JavaScript	For example: C, C++

2. What is use of modules?

- Use the modules to breakdown large program into small manageable and organized files.
- Modules provides reusability of code.
- Modules used in a function and import it instead of copying definitions into different programs.

3. Differentiate Compiler and Interpreter.

Compiler	Interpreter
Translates the whole program into machine code.	Translates one statement in a program at a time.
It takes large amount of time to analyze the source code but fast execution.	It takes less amount of time to analyze the source code but slow execution.
For example: C, C++	For example: Python, JavaScript

3 MARKS:

1. What are the applications of scripting language?

- To automate certain tasks in a program.
- Extracting information from a data set.
- Less code as compared to programming language.
- It can bring new functions to applications and glue complex system together.

2. Differentiate PYTHON and C++.

PYTHON	C++
PYTHON is an 'interpreted' language.	C++ is a 'compiled' language.
PYTHON is a dynamic typed language.	C++ is compiled statically typed language.
Data type is not required.	Data type is required.
It acts as scripting and general purpose language.	It acts as a general purpose language.

3. What is MinGW and its usage?

- MinGW is a set of running header files, used in compiling and linking the code of C,C++ to be run on Windows Operating System.
- MinGW used to compile and execute C++ program dynamically through Python program using g++.

15. DATA MANIPULATION THROUGH SQL

2 MARKS:

1. Mention the users who uses the Database.

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

2. 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, a NULL will be used as an input for this column.
- The NULL will be automatically converted into an integer which is having highest value as used in that column.

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

- The fetchall() method is used to fetch all rows from the database table.
- For example:

```
cursor.execute("select * from student")
result=cursor.fetchall()
```

3 MARKS:

1. What is SQLite and its advantage?

- SQLite is a simple Relation Data Base System saves its data in regular data files or in the internal memory of the computer.
- It is designed to set in applications, instead of using separate database server program.
- **ADVANTAGE:** SQLite is fast, carefully tested and flexible as making it easier to work.
- PYTHON has a native library for SQLite.

2. Mention the difference between fetchone() and fetchmany().

fetchone()	fetchmany()
The fetchone() method returns the next row of a query result set or in case None, there no row left.	Displaying specified number of records is done by using fetchmany().
Using while loop and fetchone() method can display all the records from a table.	This method returns the next number of rows(n) of the result set.
For example: cursor.execute("select * from student") result=cursor.fetchone()	For example: cursor.execute("select * from student") result=cursor.fetchmany(n) (n=number of records)

3. What is the use of WHERE Clause? Give a Python statement using the WHERE clause.

- The WHERE clause is used to extract only records with the specified condition.
- For example:

```
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)
```

16. DATA VISUALIZATION USING PYPLOT

2 MARKS:

1. Define – Data Visualization.

- Data Visualization is the graphical representation of information and data.
- Data Visualization is to communicate information visually to users.
- Data Visualization is using statistical graphics.

2. List the General types of Data Visualization.

- Charts
- Tables
- Graphs
- Maps
- Infographics
- Dashboards

3. List the types of Visualizations in Matplotlib.

- Line plot
- Scatter plot
- Histogram
- Box plot
- Bar Chart and
- Pie Chart

3 Marks:

1. 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.

2. Write the coding for following:

a. To Check if PIP is installed in your PC:

In command prompt, we type as

C:\Users\XII\AppData\Local\Programs\Python\Python36-32 > **Python –m pip install –U pip**

b. To Check the version of PIP installed in your PC:

In command prompt, we type as

C:\Users\XII\AppData\Local\Programs\Python\Python36-32\Scripts > **pip --version**

c. To list the packages in matplotlib:

In command prompt, we type as

C:\Users\XII\AppData\Local\Programs\Python\Python36-32\Scripts > **pip list**

d. To install Matplolib in the PC:

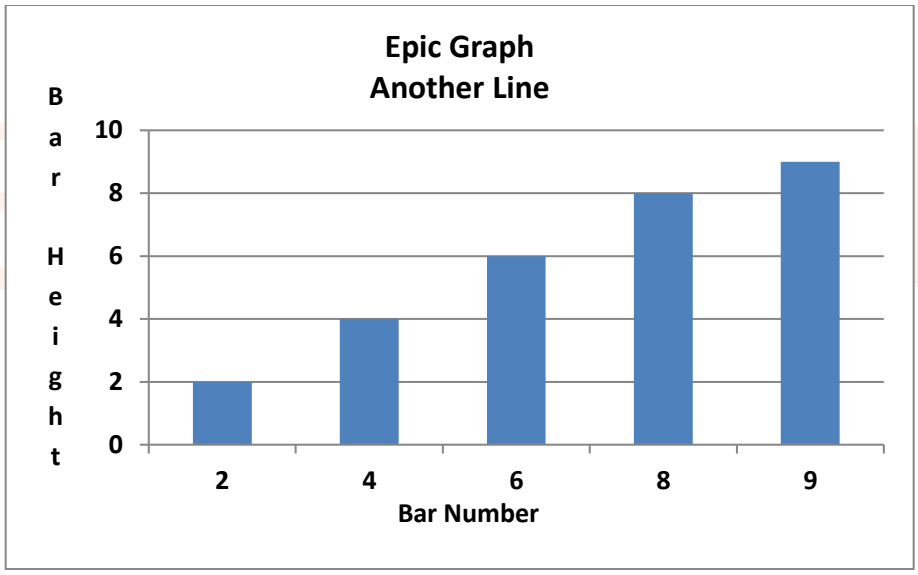
In command prompt, we type as

C:\Users\XII\AppData\Local\Programs\Python\Python36-32\Scripts >

Python –m pip install –U matplotlib

3. 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!)
plt.show()
```



4. Read the following details. Based on that write a python script to display records in descending 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((“select * from Employee ORDER BY Eno DESC”)
result=cursor.fetchall()
print(*result,sep=”\n”)
```

----

11. DATABASE CONSEPTS

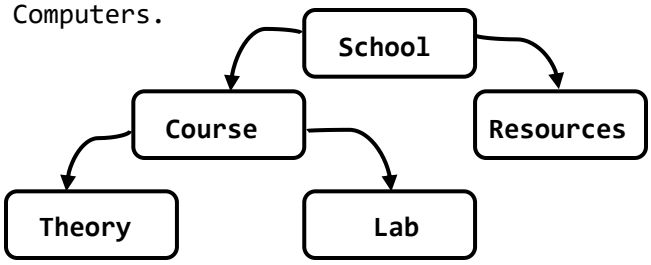
5 Marks:

1. Explain different types of data model.

Following are different types of a Data Model:

1. Hierarchical Model:

- This model was developed by **IBM** as **Information Management System.**
- In this model, data was represented as a simple tree like structure form.
- This model represents a one-to-many relationship like as Parent-Child relationship.
- This model is mainly used in IBM Main Frame Computers.



2. Relational Model:

- This Model was first proposed by E.F. Codd in 1970.
- In this model, the basic structure of data is table or relations.
- All the informations related to a particular type is stored in rows of that table.
- A Relation key is uniquely identifies a particular record or tuple.

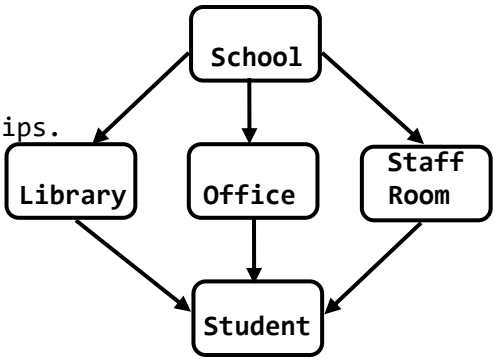
Stud_id	Name	Age
1	Malar	17
2	Sankar	16
3	Velu	16

Subj_id	Name	Teacher
1	C++	Kannan
2	PHP	Ram
3	Python	Vidhya

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

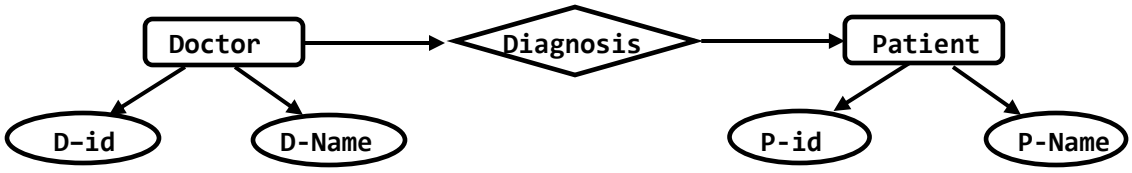
3. Network Model:

- This model is an extended form of hierarchical model.
- This model represents the data in many-to-many relationships.
- This model is easier and faster to access the data.



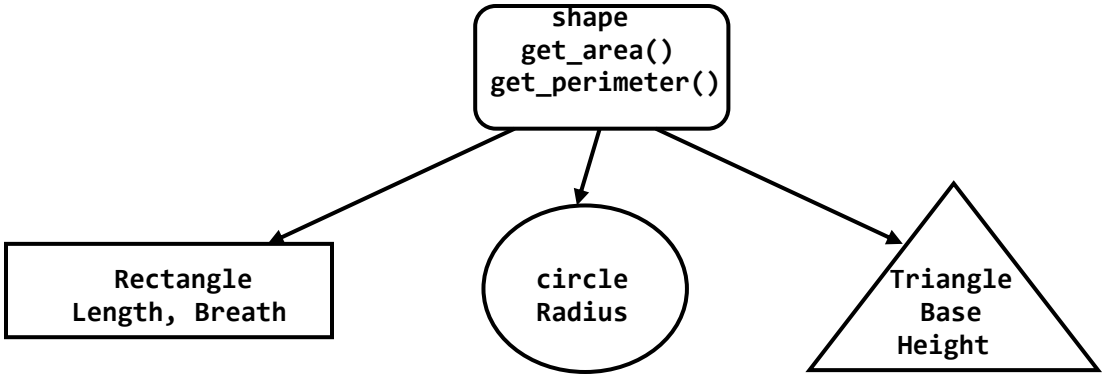
4. Entity Relationship Model(ER Model):

- This model was developed by **Chen** in **1976.**
- In this model, relationships are created by dividing the object into entity and its characteristics into attributes.
- This model is useful in developing a abstract design for the database.
- It is very simple and easy to design logical view of data and the developer can easily understand the system by looking at ER model constructed.



5. Object Model:

- This model stores the data in the form of objects, attributes and methods, classes and inheritance.
- This model handles more complex applications.
- It provides a clear modular structure and it is easy to maintain and modify the existing code.
- It is used in File Management System and also it represents real world objects, attributes and behaviors.



An example of object model is

- **Circle** has the attribute **radius.**
- **Rectangle** has the attributes **length and breadth.**
- **Triangle** has the attributes **base and height.**

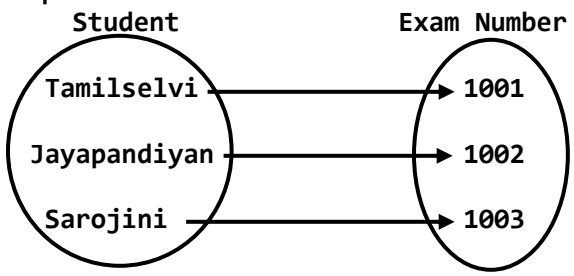
2. Explain the different types of relationship mapping.

Following are the types of Relationships used in the database.

1. **One-to-One Relationship**
2. **One-to-Many Relationship**
3. **Many-to-One Relationship**
4. **Many-to-Many Relationship**

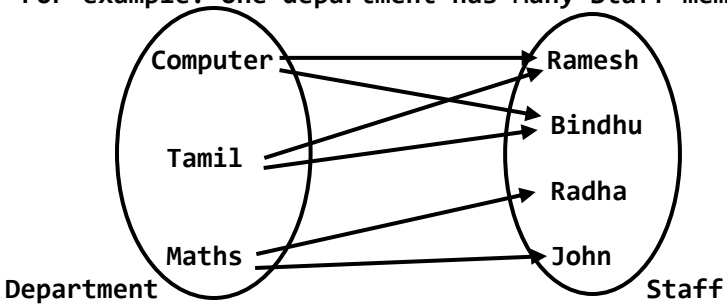
1. One-to-One Relationship:

- 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.
- **Example: A student can have one exam number.**



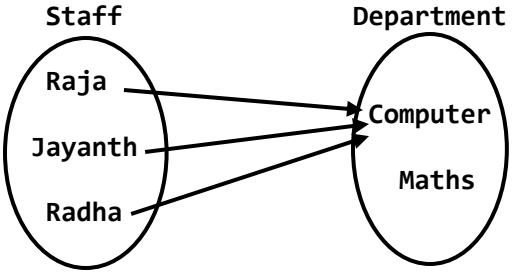
2. One-to-Many Relationship:

- In One-to-Many Relationship, **One Entity** is **related** to **many** other entities.
- One row in a table is linked with many rows in another table.
- **For example: One department has Many Staff members.**



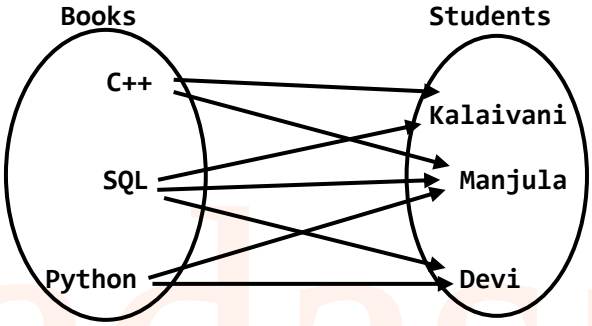
3. Many-to-One Relationship:

- In Many-to-One Relationship, **Many Entities** can be **related** with **only one** in other entity.
- Multiple rows in a table is related with one row in another table.
- **For example: Many staff members can work with one department.**



4. Many-to-Many Relationship:

- A Many-to-Many Relationship **Multiple records** can be **related** with **Multiple records** with another table.
- **For example: (i) Many Customers can relate with Many Products.**
  (ii) Many Students can register with Many Courses.
  (iii) Many Books are issued to Many Students.



3. Differentiate DBMS and RDBMS.

Sl.No.	Basis of Comparison	DBMS	RDBMS
1.	Expansion	DataBase Management System	Relational DataBase Management System
2.	Data Storage	Navigational Model	Relational Model
3.	Data Redundancy	Exhibit	Not Present
4.	Data Access	Consumes more time	Faster, compared to DBMS
5.	Distributed Databases	Not supported	Supported by DBMS
6.	Keys and indexes	Does not use	Used to create relationship
7.	Normalization	Not performed	Used to reduce redundancy
8.	Transaction Management	Inefficient	Efficient and secure
9.	Example	Dbase, ForPro	SQL server, Oracle, mysql, MariaDB, SQLite

4. Explain the characteristics of DBMS.

Sl.No.	Characteristics	Definition
1.	**Data stored into Tables**	Data is never directly stored into the database. Data is stored into tables, created inside the database.
2.	**Reduced Redundancy**	DBMS follows Normalization which divides the data in such a way that repetition is minimum.
3.	**Data Consistency**	The data is being continuously updated and added, maintaining the consistency of data.
4.	**Support Multiple user and Concurrent Access**	DBMS allows multiple users to work on it and manages to maintain the data consistency.
5.	**Query Language**	DBMS provides users with a simple query language.
6.	**Security**	The DBMS also takes care of the security of data, protecting the data from unauthorized access.
7.	**DBMS Supports Transactions**	It allows us to better handle and manage data integrity in real world applications.

12. STRUCTURED QUERY LANGUAGE

5 Marks:

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

Constraints:

i) Constraints are used to limit the type of data that can go into a table.
ii) This ensures the accuracy and reliability of the data in the database.
iii) Constraints could be either on a column level or a table level.

Types:

1. Unique constraints
2. Primary Key Constraints
3. Default Constraints
4. Check Constraints

(i) Unique Contraints:

1. This constraint ensures that no two have the same value in the specified columns.
2. This constraint can be applied only to fields that have also been declared as NOT NULL.
3. When two constraints like as NOT NULL and UNIQUE are applied on a single field.

(ii) Primary Key Constraint:

1. This constraint declares a field as a Primary key which helps to Uniquely identify a record.
2. The Primary Key does not allow NULL values and a field declared as primary key must have the NOT NULL constraint.

(iii) Default Constraint:

1. This constraint is used to assign a default value for the field.
2. When no value is given for the specified field, the default value will be assigned to the field for the default constraint.

(iv) Check Constraint:

1. This constraint helps to set a limit value placed for a field.
2. When we define a check constraint on a single column, it allows only the restricted values on that field.

(v) Table Constraint:

1. When the constraint is applied to group of fields of the table.
2. This constraint is normally given at the end of the table definition.

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

SQL(Structured Query Language) commands are divided into **FIVE** categories.

1. **DDL**(Data Definition Language)
2. **DML**(Data Manipulation Language)
3. **DCL**(Data Control Language)
4. **TCL**(Transaction Control Language)
5. **DQL**(Data Query Language)

1. DDL(Data Definition Language):

DDL consist of SQL statements used to define the database structure and is used to create and modify the structure of database objects in databases.

SQL Commands for DDL:

- **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 and release the space occupied by those records.

2. DML(Data Manipulation Language):

DML is a computer programming language used for inserting, deleting and updating data in a database.

SQL Commands for DML:

- **Insert:** Inserts data into a table.
- **Update:** Updates the existing data within a table.
- **Delete:** Deletes records from a table, but not the space occupied by them.

3. DCL(Data Control Language):

DCL is a programming language used to control the access of data stored in a database.

SQL commands for DCL:

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

4. TCL(Transaction Control Language):

TCL commands are used to manage transactions in the database.

SQL commands for TCL:

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

5. DQL(Data Query Language):

DQL consist of commands used to query or retrieve data from a database.

SQL commands for DQL:

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

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

Table Creation:

```
create table employee
   (ecode integer NOT NULL UNIQUE,
    ename char(20) NOT NULL,
    design char(10),
    pay integer,
    allowance integer);
```

Insert rows:

```
insert into employee(ecode, ename, design, pay, allowance)
  values (1001,'Harish','superviser',29000, 12000);
insert into employee(ecode, ename, design, pay, allowance)
  values (1002,'Shaji','operator',12000, 6500);
insert into employee values (1003,'Ratheesh','mechanic', 20000, 7000);
insert into employee values (1004,'Manju','clerk', 8000, 4500);
```

Display records:

Select * from employee;

ecode	ename	design	pay	allowance
1001	Harish	supervisor	29000	12000
1002	Shaji	operator	12000	6500
1003	Ratheesh	mechanic	20000	7000
1004	Manju	clerk	8000	4500

13. PYTHON AND CSV FILES

5 Marks:

1. Differentiate Excel file and CSV file.

Excel	CSV
Excel is a binary file holds information about all the worksheets in a file.	CSV format is a plain text format with a series of values separated by commas.
Excel files can only can be read the applications have been written to read their format and its same way.	CSV can be open with any text editor in Windows.
Excel is spreadsheet that saves files as extension of .XLS or .XLSX	CSV is a format that saves files as extension of .csv
Excel consumes more memory while importing data.	CSV files can be much faster and consumes less memory.

2. Tabulate the different mode with its meading.

Mode	Description
'r'	**Open a file for reading(default)**
'w'	Open a file for writing and also creating new file if it does not exist.
'x'	Open a file for exclusive creation.
'a'	Open for appending at the end of the file without truncating it
't'	**Open in text mode(default)**
'b'	Open in binary mode
'+'	Open a file for updating(reading and writing)

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

- Each record to be located on a separate line, delimited by a line break by pressing enter key. **For example: xxx.yyy↵**
- The last record in the file may or may not have an ending line break.
  For example: ppp,qqq↵
  yyy,xxx
- There may be optional header line appearing as the first line of the file with the same format as normal record lines.
  For example: aaa,bbb,ccc↵
  zzz,yyy,xxx CRLF(Carriage Return and Line Feed)
- Within the header and each record, there may be one or more fields, separated by commas. The last field in the record must not be followed by a comma.
  For example: Red, Blue

- Each field may or may not be enclosed in double quotes. The double quotes may not appear inside the fields.
  For example : "Red", "Blue", "Green"↵
  Black, White, Yellow
- Fields containing line breaks, double quotes and commas should be enclosed in double-quotes.
  For example: Red. ",",Blue
  Red, Blue, Green
- 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

14. IMPORTING C++ PROGRAMS IN PYTHON

5 Marks:

1. Write features of Python.

- Python uses Automatic Garbage Collection whereas C++ does not.
- Python runs through an interpreter, while C++ is pre-compiled.
- Python code tends to be 5 to 10 times shorter then C++.
- Python no need to declare types but in C++ should be done.
- In Python, a function may accept any type and return multiple values but 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**

15. DATA MANIPULATION THROUGH SQL

5 Marks:

1. What is the use of HAVING clause? Give an example python script.

HAVING clause:

- 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.
- **For 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)]

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

Assume, Database name : shop.db
Table name : electronics

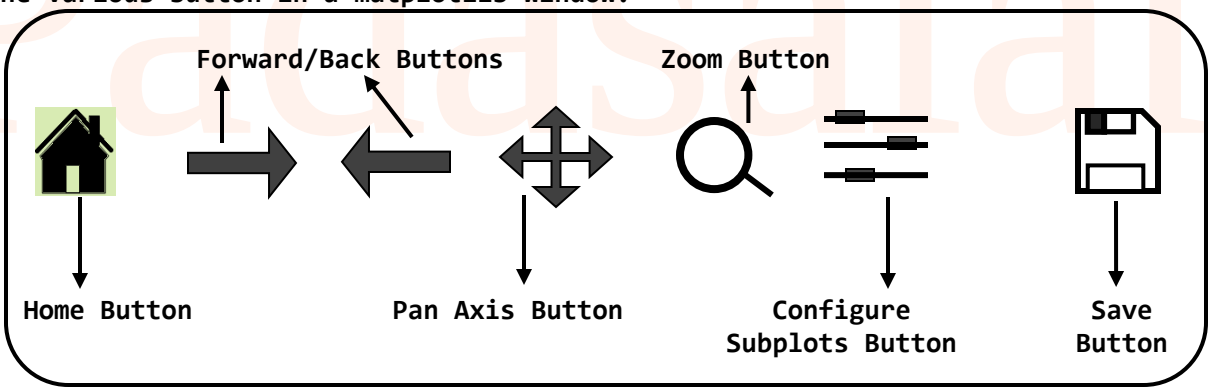
Python script:

```
import sqlite3
connection = sqlite3.connect("shop.db")
cursor = connection.cursor()
cursor.execute("select * from electronics")
print("fetching the records")
result = cursor.fetchmany(5)
print(*result, sep = "\n")
```

16. DATA VISUALIZATION USING PYPLOT: LINE CHART,PIE CHART AND BAR CHART

5 Marks:

1. Explain the various button in a matplotlib window.



Home Button: This button will help once navigating the chart. If want return back to original view, click on this.

Forward/Back Button: This buttons can be used to Forward and Back the Browser. This can be used to move previous point or forward again.

Pan Axis Button: This button allows to drag the graph around by click this.

Zoom Button: By clicking this button allows to drag a square would like zoom specifically. For Zooming requires a left click and Zoom out is right click and drag.

Configure Subplots: This button allows to configure various options with figure and plot.

Save figure Button: This button will allows to save figure in various forms.

2. Explain the purpose of the following functions:
(i) plt.xlabel (ii) Plt.ylabel (iii) plt.title
(iv) plt.legend() (v) plt.show()

(i) **Plt.xlabel:** Specifies label for x-axis.
(ii) **Plt.ylabel:** Specifies label for y-axis.
(iii) **Plt.title:** Specifies title to the graph or assign the plot title.
(iv) **Plt.legend():** Invoke the default legend with plt.
(v) **Plt.show():** Display the plot.

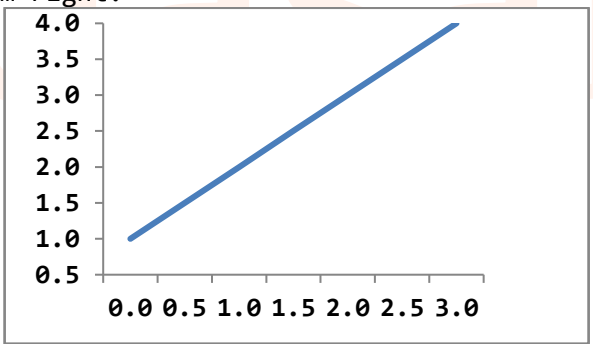
3. Write the key difference between Histogram and Bar Graph:

Histogram	Bar Graph
Histogram refers a graphical representation.	Bar Graph refers a pictorial representation.
Histogram represents the frequency distribution of continuous variables.	Bar Graph represents diagrammatic comparison of discrete variables
Histogram presents Numerical data.	Bar Graph shows categorical data.
Histogram drawn no gap between bars.	Bar Graph drawn proper spacing between bars to indicate discontinuity.
Items of Histogram are numbers represents ranges of data.	Items of Bar Graph are individual entities.
In Histogram, cannot be rearrange block of bars. Because they are in sequence.	In Bar Graph rearrange the block of bars from high to low.
Histogram blocks Width are may or may not be same.	Bar Graph blocks Width are always same.

3 Marks:

1. Write the difference between the following functions:
plt.plot([1,2,3,4]) and plt.plot([1,2,3,4], [1,4,9,16])

plt.plot([1,2,3,4]):
This matplotlib window allows to see the graph. This can over the graph and see the co-ordinates in the bottom right.



plt.plot([1,2,3,4],[1,4,9,16]):
This window takes many parameters, but the first two are 'x' and 'y' co-ordinates. These co-ordinates according to the lists: (1,1),(2,4), (3,9) and (4,16).