V.M.G. RAJASEKARAN – RAMANI SRI SARADA SAKTHI MAT. HR. SEC. SCHOOL

STANDARD – 12

PART – III – COMPUTER SCIENCE

PART – I

1. a. Arguments

2. b. Derived datatype

3. c. =

4. c. Memorization

5. a. =

6. b. continue

7. d. -37

8. b. Multiline strings

9. a. The extend () function is used in tuple to add elements in a list.

10. d. Instantiation

11. b. EF Codd

12. c. ORDER BY

13. a. Wrapping

14. d. Select

15. b. PIP

PART – II

16. **What are subroutine?**

- ✓ Subroutines are **small sections** of code.
- ✓ Used to **perform a particular task** that can be used repeatedly.
- ✓ In Programming languages **subroutines** are called as Functions.

17. What is searching? Write its types.

- ✓ A Search algorithm is the step-by-step procedure used to locate specific data among a collection of data.

Example: Linear Search, Binary Search

18. **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.

19. Write a Python program to print

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

```
i=1
while (i<=6):
for j in range (1,i):
print (j,end='\t')
print (end='\n')
i +=1
```

20. **What are the main advantages of function?**

Main advantages of functions are ,

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

21. What will be the output of the following python code?

```
    St"Chennai"
Print(str*4)
```

Output:

Chennai Chennai Chennai Chennai

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

- ✓ Python enables reverse or negative indexing for the list elements.
- ✓ A negative index can be used to access an element in reverse order.
- ✓ 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.**

23. 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

24. **Define: Data Visualization.**

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

PART - III

25. **Why access control is required?**

- ✓ Access control is a security technique that regulates who or what can view or use resources in a computing environment.
- ✓ It is a fundamental concept in security that minimizes risk to the object.
- ✓ In other words access control is a selective restriction of access to data.
- ✓ In OOPS Access control is implemented through access modifiers.

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

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

27. What is the use of format( )? Give an example.

✓ The **format( )** function used with strings is very powerful function used for formatting strings.

✓ The curly braces { } are used as placeholders or replacement fields which get replaced along with format( ) function.

EXAMPLE:

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

OUTPUT:

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

28. What are the difference between List and Dictionary?

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

29. Write a Python program to check and print if the given number is odd or even using class.

```
class Odd_Even:
even = 0
def check(self, num):
if num%2==0:
print(num," is Even number")
else:
print(num," is Odd number")
n=Odd_Even()
```

```
        x = int(input("Enter a value: "))
        n.check(x)
```

Output 1

```
Enter a value: 4
4 is Even number
```

Output 2

```
Enter a value: 5
5 is Odd number
```

30. What is a constraint? Write short note on Primary key constraint.
Constraint is a condition applicable on a field or set of fields.

- ✓ Primary constraint declares a field as a Primary key which helps to uniquely identify a record.
- ✓ It is similar to unique constraint except that only one field of a table can be set as primary key.
- ✓ The primary key does not allow **NULL** values and therefore a primary key field must have the **NOT NULL** constraint.

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

- ✓ Python has a built-in function **open()** to open a file.
- ✓ This function returns a file object, also called a handle, as it is used to read or modify the file accordingly.
- ✓ The **default is reading** in text mode.
- ✓ In this mode, while reading from the file the data would be in the format of **strings.**
- ✓ On the other hand, binary mode returns bytes and this is the mode to be used when dealing with non-text files like image or exe files.

32. 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:

- ✓ SQLite is fast, rigorously tested, and flexible, making it easier to work.
- ✓ Python has a native library for SQLite.

33. 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.

PART – IV

34. a. Write any Five Characteristics of Modules.

The following are the desirable characteristics of a module.

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

b. **Explain the characteristics of an algorithm.**

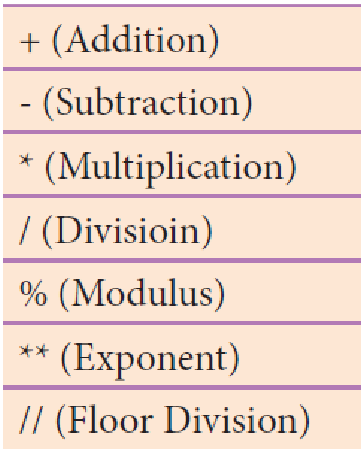
Characteristics	Meaning
Input	Zero or more quantities to be supplied.
Output	At least one quantity is produced.
Finiteness	Algorithms must terminate after finite number of steps.
Definiteness	All operations should be well defined.
Effectiveness	Every instruction must be carried out effectively.
Correctness	The algorithms should be error free.
Simplicity	Easy to implement.
Unambiguous	Algorithm should be clear and unambiguous. Each of its steps 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 and able to handle all range of inputs.
Independent	An algorithm should have step-by-step directions, which should be independent of any programming code.

35. a. **Explain the various operators in Python.**

- ✓ **Operators are special symbols** which represent computations, conditional matching in programming.
- ✓ Operators are categorized as Arithmetic, Relational, Logical, Assignment and Conditional.

Arithmetic operators

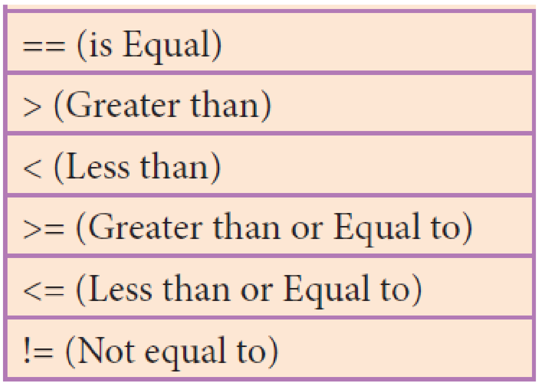
An arithmetic operator is a mathematical operator that takes two operands and performs a calculation on them. They are used for simple arithmetic.



+ (Addition)
- (Subtraction)
* (Multiplication)
/ (Divisioin)
% (Modulus)
** (Exponent)
// (Floor Division)

Relational or Comparative operators

A Relational operator is also called as **Comparative** operator which checks the relationship between two operands. If the relation is true, it returns **True**; otherwise it returns **False**.



== (is Equal)
> (Greater than)
< (Less than)
>= (Greater than or Equal to)
<= (Less than or Equal to)
!= (Not equal to)

Logical operators

In python, Logical operators are used to perform logical operations on the given relational expressions. There are three logical operators they are **and, or** and **not**.

Assignment operators

In Python, = is a simple assignment operator to assign values to variable. There are various compound operators in Python like +=, -=, *=, /=, %=, **= and //= are also available.

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

Nested if..elif...else statement:

- ✓ When we need to construct a chain of **if** statement(s) then **„elif‟** clause can be used instead of **„else‟**.
- ✓ **„elif** clause combines **if..else-if..else** statements to one **if..elif…else.**
- ✓ **„elif‟** can be considered to be abbreviation of **„else if‟.**
- ✓ In an **„if‟** statement there is no limit of **„elif‟** clause that can be used, but an **„else‟** clause if used should be placed at the end.

Syntax:

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

- ✓ In the syntax of **if..elif..else** mentioned above, condition-1 is tested if it is true then statements-block1 is executed.
- ✓ Otherwise the control checks condition-2, if it is true statements-block2 is executed and even if it fails statements-block n mentioned in **else** part is executed.

Example:

```
m1=int (input("Enter mark in first subject : "))
m2=int (input("Enter mark in second subject : "))
avg= (m1+m2)/2
if avg>=80:
print ("Grade : A")
elif avg>=70 and avg<80:
print ("Grade : B")
elif avg>=60 and avg<70:
print ("Grade : C") elif avg>=50 and avg<60:
print ("Grade : D")
else:
print ("Grade : E")
```

Output :

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

36. a. **Explain the scope of variables with an example.**

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

We can say that scope holds the current set of variables and their values.

There are two types of scopes - **local scope** and **global scope**.

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

Rules of local variable:

A variable with local scope can be accessed only within the function/block that it is created in.

When a variable is created inside the function/block, the variable becomes local to it. A local variable only exists while the function is executing.

The formal arguments are also local to function.

Example:

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

Output: 0

Global Scope

A variable, with global scope can be used anywhere in the program.

It can be created by defining a variable outside the scope of any function/block.

Rules of global Keyword

The basic rules for ***global*** keyword in Python are:

When we define a variable outside a function, it‟‟s global by default.

You don‟‟t have to use global keyword.

We use global keyword to read and write a global variable inside a function.

Use of global keyword outside a function has no effect

Use of global Keyword

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

Example:

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

Output:

```
Inside add() function x value is : 5
In main x value is : 5             #value of x changed outside the function
```

b. **Explain about string operators in python with suitable example.**

STRING OPERATORS

Python provides the following string operators to manipulate string.

(i) Concatenation (+)

- ✓ Joining of two or more strings using plus **(+) operator** is called as **Concatenation**. **Example** >>> "welcome" + "Python"

Output: 'welcomePython'

(ii) Append (+ =)

- ✓ Adding more strings at the end of an existing string using **operator +=** is known as **append.**

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

Output:

Welcome to Learn Python count(str, beg, end)

(iii) Repeating (*)

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

Example:

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

Output:

Welcome Welcome Welcome Welcome

(iv) String slicing

- ✓ Slice is a substring of a main string.
- ✓ A substring can be taken from the original string by using **[ ] slicing operator** and index values.
- ✓ Using slice operator, you have to slice one or more substrings from a main string. **General format of slice operation:** str[start:end]
- ✓ Where ***start*** is the beginning index and ***end*** is the last index value of a character in the string.
- ✓ Python takes the end value less than one from the actual index specified.

Example:

slice a single character from a string

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

Output:

```
        T
```

(v) Stride when slicing string

- ✓ When the slicing operation, you can specify a third argument as the stride, which refers to the number of characters to move forward after the first character is retrieved from the string.
- ✓ The default value of stride is 1.
- ✓ Python takes the last value as n-1

- ✓ You can also use negative value as stride, to prints data in reverse order.

Example:

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

Output:

```
Learn
nhy re teolW
```

37. a. **What is the purpose of range( )? Explain with an example.**

range():

- ✓ The range( ) is a function used to generate a series of values in Python.
- ✓ Using range( ) function, you can create list with series of values.
- ✓ The range( ) function has three arguments.

Syntax of range ( ) function:

range (start value, end value, step value) where,

- ✓ **start value** – beginning value of series. Zero is the default beginning value.
- ✓ **end value** – upper limit of series. Python takes the ending value as upper limit – 1.
- ✓ **step value** – It is an optional argument, which is used to generate different interval of values.

Example :

Generating whole numbers upto 10

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

Output:

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

Creating a list with series of values

- ✓ Using the range( ) function, you can create a list with series of values.
- ✓ To convert the result of range( ) function into list, we need one more function called list( ).

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

Syntax:

List_Varibale = list (range ( ))

Example :

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

Output:

```
[2, 4, 6, 8, 10]
```

- ✓ In the above code, list( ) function takes the result of range( ) as Even_List elements.
- ✓ Thus, Even_List list has the elements of first five even numbers.

b. **Explain the different types of data model.**

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

Types of Data Model

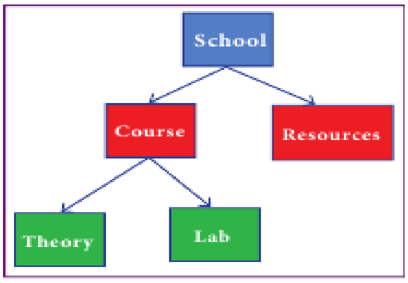
The different types of a Data Model are,

- ✓ Hierarchical Model
- ✓ Relational Model
- ✓ Network Database Model
- ✓ Entity Relationship Model
- ✓ Object Model

i). Hierarchical Model:

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

Example:

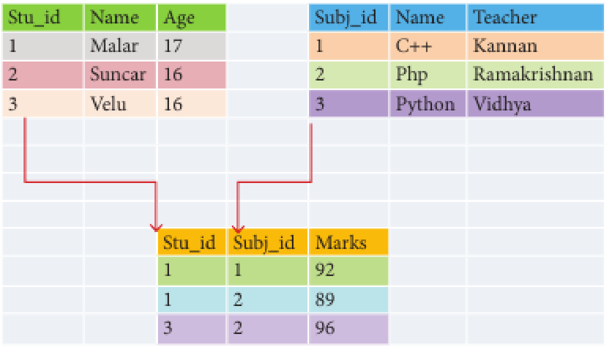


ii). Relational Model

- ✓ The Relational Database model was first proposed by E.F. Codd in 1970 .
- ✓ The basic structure of data in relational model is tables (relations).

- All the information's related to a particular type is stored in rows of that table.
- Hence tables are also known as relations in a relational model.
- A relation key is an attribute which uniquely identifies a particular tuple (row in a relation (table)).

Example:



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

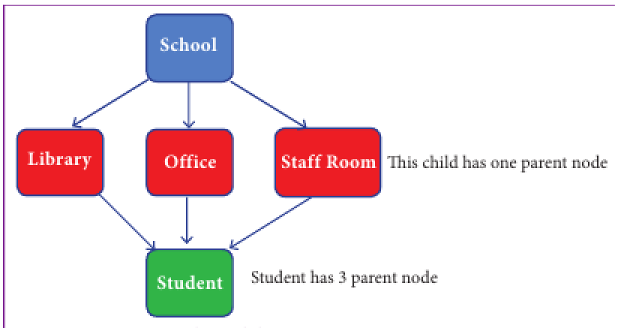
Subj_id	Name	Teacher
1	C++	Kannan
2	Php	Ramakrishnan
3	Python	Vidhya

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

Relational Model

iii.) Network Model

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

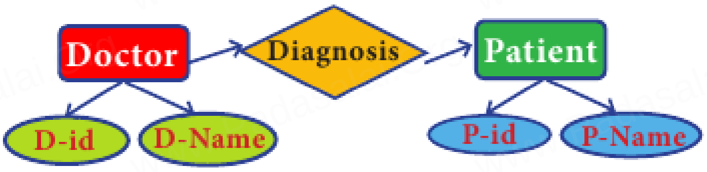


Network Model

iv.) Entity Relationship Model. (ER model)

- In this database model, relationship are created by dividing the object into entity and its characteristics into attributes.
- It was developed by Chen in 1976.
- ER model constructed by,
  - **Rectangle** represents the entities.
  - **Ellipse** represents the attributes .
  - **Attributes** describes the characteristics and each entity.
  - **Diamond** represents the relationship in ER diagrams

Example:Doctor diagnosis the Patient.

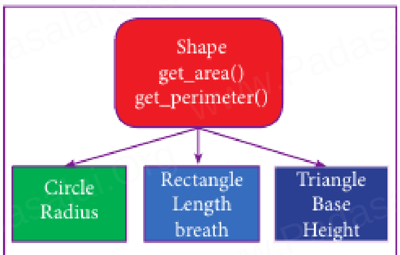


ER model

v.) 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.

Example: 2.



Object Model

38. a.Consider the following student table. Write SQL command for the questions (i) to (v)

(i) To display the details of all the students in ascending order of Name.

SELECT * FROM STUDENT ORDER BY ASC;

(ii) To display all students in A2 group.

SELECT * FROM student WHERE Group A2;

(iii) To display the details group wise

SELECT * FROM student WHERE Group BETWEEN A1 AND B1;

(iv) To add a new row.

INSERT INTO student (rollno,name,group)VALUES(1010, Kishan, A1);

(iii) To remove the student who are in B1 group.

DELETE FROM employee WHERE group="B1";

b. Mylist={10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

Write the Python commands for the following based on above list.

i) To print all elements in list
ii) Find list length
iii)Add Multiple elements [110, 120, 130]
iv) Delete from fourth element to seventh element
v) Delete entire list.

i) To print all elements in list

```
        Mylist=[10,20,30,40,50,60,70,80,90,100]
        i=0
        while i<10:
        print(Mylist[i])
        i=i+1
```

ii) Find list length

```
>>>MyList = [10,20.30,40,50,60,70,80,90,100]
>>>len(MyList)
```

iii) Add multiple elements [110, 120, 130]

```
>>>MyList = [10,20.30,40,50,60,70,80,90,100]
>>>Mylist.extend([110,120,130])
```

iv) Delete from fourth element to seventh element

```
>>>MyList [10,20.30,40,50,60,70,80,90,100,110,120,130]
>>>del MyList[3:7]
>>>print(MyList)
        [10,20.30,80,90,100,110,120,130]
```

v) Delete entire list.

```
>>>MyList[10,20.30,80,90,100,110,120,130]
>>>del MyList
>>>print(MyList)
Name Error: Name'MyList' is not defined.
```

M.GEETHA,

PG ASSISTANT,

V.M.G.RAJASEKARAN – RAMANI SRI SARADA SAKTHI MHSS,

VIRUDHUNAGAR.