



COMPUTER SCIENCE

12th Standard

FREE
Practice Workbook
with
Lab Manual

Based on the Updated New Textbook

Salient Features

- ✦ Exhaustive Additional MCQs, VSA and SA question with answers are given in each chapter.
- ✦ All the objective type (1 Mark) questions are given with 4 options.
 - (i) Choosing the correct option
 - (ii) Matching
 - (iii) Filling the blanks
 - (iv) Choosing the Correct/Incorrect Statement.
 - (v) Picking the Odd one Out.
 - (vi) Assertion and Reason.
- ✦ Model Question Papers 1 to 6 (PTA) : Questions are incorporated in the appropriate sections.
- ✦ Govt. Model Question Paper - 2019 (Govt. MQP-2019), Quarterly Exam - 2019 (QY-2019), Half Yearly Exam - 2019 (HY-2019), Public Exam March - 2020 (Mar-2020) and Govt. Supplementary Exam - Sep - 2020 (Sep-2020) are incorporated in the appropriate sections.
- ✦ Govt. Suppl. Exam 2020 question paper is given with answers.



SURA PUBLICATIONS

Chennai

2021-22 Edition

All rights reserved © SURA Publications.

No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, digitally, electronically, mechanically, photocopying, recorded or otherwise, without the written permission of the publishers. Strict action will be taken.

ISBN : 978-81-8449-799-1

Code No : SG91

Author :

Mr. Shanmugasundaram

(Post Graduate Teacher, Chennai)

Edited by :

Mrs. Malathy Krishnamoorthy M.Sc.,

Erode

Reviewed by :

Mr. Balaji M.Sc., M.Phil.

Chennai

Head Office:

1620, 'J' Block, 16th Main Road,

Anna Nagar, **Chennai - 600 040.**

Phone: 044-4862 9977, 044-486 27755.

Mob : 81242 01000/ 81243 01000

Fax : (91) 44-26162173

e-mail : orders@surabooks.com

website : www.surabooks.com

For More Information - Contact

Queries	:	enquiry@surabooks.com
For Order	:	orders@surabooks.com
Contact	:	96001 75757 / 8124301000
Whatsapp	:	8124201000 / 9840926027
Online Site	:	www.surabooks.com
For Free Study Materials Visit http://tnkalvi.in		

PREFACE

“ The woods are lovely, dark and deep. “

*But I have promises to keep, and
miles to go before I sleep*

- Robert Frost

Respected Principals, Correspondents, Head Masters / Head Mistresses, Teachers,

From the bottom of our heart, we at SURA Publications sincerely thank you for the support and patronage that you have extended to us for more than a decade.

It is in our sincerest effort we take the pride of releasing **SURA's Computer Science Guide** for +2 Standard – Edition 2021-22. This guide has been authored and edited by qualified teachers having teaching experience for over a decade in their respective subject fields. This Guide has been reviewed by reputed Professors who are currently serving as Head of the Department in esteemed Universities and Colleges.

With due respect to Teachers, I would like to mention that this guide will serve as a teaching companion to qualified teachers. Also, this guide will be an excellent learning companion to students with exhaustive exercises and in-text questions in addition to precise answers for textual questions.

In complete cognizance of the dedicated role of Teachers, I completely believe that our students will learn the subject effectively with this guide and prove their excellence in Board Examinations.

I once again sincerely thank the Teachers, Parents and Students for supporting and valuing our efforts.

God Bless all.

Subash Raj, B.E., M.S.

- Publisher

Sura Publications

All the Best

CONTENTS

Unit	Chapter No	Title	Page No
UNIT- I Problem Solving Techniques	1.	Function	1-8
	2.	Data Abstraction	9-17
	3.	Scoping	18-27
	4.	Algorithmic Strategies	28-43
UNIT- II Core Python	5.	Python -Variables and Operators	44-60
	6.	Control Structures	61-77
	7.	Python functions	78-98
	8.	Strings and String manipulations	99-114
UNIT-III Modularity and OOPS	9.	Lists, Tuples, Sets and Dictionary	115-138
	10.	Python Classes and objects	139-151
UNIT-IV Database concepts and MySql	11.	Database Concepts	152-169
	12.	Structured Query Language (SQL)	170-191
	13.	Python and CSV files	192-209
UNIT-V Integrating Python with MySql and C++	14.	Importing C++ programs in Python.	210-223
	15.	Data manipulation through SQL	224-236
	16.	Data visualization using pyplot: line chart, pie chart and bar chart	237-249

TO ORDER WITH US

SCHOOLS and TEACHERS

We are grateful for your support and patronage to '**SURA PUBLICATIONS**'

Kindly prepare your order in your School letterhead and send it to us.

For Orders contact: 81242 01000 / 81243 01000

DIRECT DEPOSIT

A/c Name : **Sura Publications**
Our A/c No. : **36550290536**
Bank Name : **STATE BANK OF INDIA**
Bank Branch : PADI
IFSC : SBIN0005083

A/c Name : **Sura Publications**
Our A/c No. : **21000210001240**
Bank Name : **UCO BANK**
Bank Branch : Anna Nagar West
IFSC : UCBA0002100

A/c Name : **Sura Publications**
Our A/c No. : **6502699356**
Bank Name : **INDIAN BANK**
Bank Branch : ASIAD COLONY
IFSC : IDIB000A098

A/c Name : **Sura Publications**
Our A/c No. : **1154135000017684**
Bank Name : **KVB BANK**
Bank Branch : Anna Nagar
IFSC : KVBL0001154

After Deposit, please send challan and order to our address.

email : orders@surabooks.com / Whatsapp : 81242 01000.

DEMAND DRAFT / CHEQUE

Please send Demand Draft / cheque in favour of '**SURA PUBLICATIONS**' payable at **Chennai**.

The Demand Draft / cheque should be sent with your order in School letterhead.

STUDENTS

Order via Money Order (M/O) to

SURA PUBLICATIONS

1620, 'J' Block, 16th Main Road, Anna Nagar,
Chennai - 600 040.

Phone : 044 - 4862 9977, 044 - 486 27755.

Mobile : 96001 75757/ 81242 01000/81243 01000.

email : orders@surabooks.com Website : www.surabooks.com



SURA'S

2021-22
EDITION

For
Class

12th
Standard
100 Marks Pattern

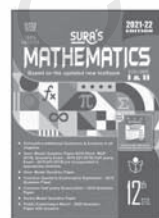
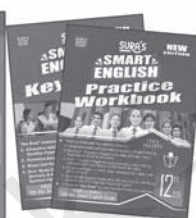
SCHOOL GUIDES



SG 142 - ₹ 360.00

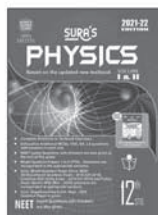


SG 101 - ₹ 399.00



SG 322 - ₹ 399.00

English
&
Tamil
Medium



SG 323 - ₹ 399.00



SG 324 - ₹ 399.00



SG 97 - ₹ 299.00



SG 281 - ₹ 299.00



SG 93 - ₹ 299.00



SG 95 - ₹ 253.00



SG 94 - ₹ 333.00



SG 325 - ₹ 399.00



SG 91 - ₹ 299.00



SG 283 - ₹ 299.00

SURA PUBLICATIONS

1620, 'J' Block, 16th Main Road, Anna Nagar,
Chennai - 600 040. INDIA. Phones: 044-48629977, 48627755
Mobile: 81242 01000 / 81243 01000
email : enquiry@surabooks.com
orders@surabooks.com

Buy online @


surabooks.com



UNIT-I PROBLEM SOLVING TECHNIQUES

CHAPTER

1

FUNCTION

CHAPTER SNAPSHOT

- 1.1 Introduction
- 1.2 Function with respect to Programming language
 - 1.2.1 Function Specification
 - 1.2.2 Parameters (and arguments)
- 1.3 Interface Vs Implementation
 - 1.3.1 Characteristics of interface
- 1.4 Pure functions
 - 1.4.1 Impure functions
 - 1.4.2 Side-effects (Impure functions)
 - 1.4.3 Chameleons of Chromeland problem using function

EVALUATION

PART - I

CHOOSE THE BEST ANSWER (1 MARK)

1. The small sections of code that are used to perform a particular task is called
 - (a) Subroutines
 - (b) Files
 - (c) Pseudo code
 - (d) Modules

[Ans. (a) Subroutines]
2. Which of the following is a unit of code that is often defined within a greater code structure?
 - (a) Subroutines
 - (b) Function
 - (c) Files
 - (d) Modules

[Ans. (b) Function]
3. Which of the following is a distinct syntactic block?
[PTA-6]
 - (a) Subroutines
 - (b) Function
 - (c) Definition
 - (d) Modules

[Ans. (c) Definition]
4. The variables in a function definition are called as
[PTA-2; QY-2019]
 - (a) Subroutines
 - (b) Function
 - (c) Definition
 - (d) Parameters

[Ans. (d) Parameters]
5. The values which are passed to a function definition are called
[HY-2019]
 - (a) Arguments
 - (b) Subroutines
 - (c) Function
 - (d) Definition

[Ans. (a) Arguments]
6. Which of the following are mandatory to write the type annotations in the function definition?
[PTA-4]
 - (a) Curly braces
 - (b) Parentheses
 - (c) Square brackets
 - (d) indentations

[Ans. (b) Parentheses]
7. Which of the following defines what an object can do?
 - (a) Operating System
 - (b) Compiler
 - (c) Interface
 - (d) Interpreter

[Ans. (c) Interface]



Sura's XII Std - Computer Science

Unit I - Chapter 1

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

- (a) Operating System (b) Compiler
(c) Implementation (d) Interpreter

[Ans. (c) Implementation]

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

[PTA-3; Mar.-2020]

- (a) Impure functions (b) Partial Functions
(c) Dynamic Functions (d) Pure functions

[Ans. (d) Pure functions]

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

- (a) Impure function (b) Partial Functions
(c) Dynamic Functions (d) Pure functions

[Ans. (a) Impure function]

PART - II

ANSWER THE FOLLOWING QUESTIONS

(2 MARKS)

1. What is a subroutine? **[PTA-1; HY-2019]**

Ans. (i) Subroutines are the basic building blocks of computer programs. Subroutines are small sections of code that are used to perform a particular task that can be used repeatedly.

(ii) In Programming languages these subroutines are called as Functions.

2. Define Function with respect to Programming language.

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

3. Write the inference you get from X:=(78).

Ans. X:= (78) has an expression in it but (78) is not itself an expression. Rather, it is a function definition. Definitions bind values to names, in this case the value 78 being bound to the name 'X'.

4. Differentiate interface and implementation.

Ans. The difference between interface and implementation is

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

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

- i) `let rec sum x y :
return x + y`
ii) `let disp :
print 'welcome'`
iii) `let rec sum num :
if (num!=0) then return num + sum
(num-1)

else
return num`

Ans. (i) Recursive function

(ii) Normal function

(iii) Recursive function

PART - III

ANSWER THE FOLLOWING QUESTIONS

(3 MARKS)

1. Mention the characteristics of Interface.

[Sep-2020]

Ans. (i) The class template specifies the interfaces to enable an object to be created and operated properly.

(ii) An object's attributes and behaviour is controlled by sending functions to the object.

2. Why strlen is called pure function?

[Govt. MQP-2019]

Ans. (i) strlen is a pure function because the function takes one variable as a parameter, and accesses it to find its length.

(ii) This function reads external memory but does not change it, and the value returned derives from the external memory accessed.

3. What is the side effect of impure function. Give example.

[PTA-5]

Ans. Impure Function has the following side effects

(i) Function impure (has side effect) is that it doesn't take any arguments and it doesn't return any value.

(ii) Function depends on variables or functions outside of its definition block.

(iii) It never assure you that the function will behave the same every time it's called.

For example :

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


(iv) Here, the result of inc() will change every time if the value of 'y' get changed inside the function definition.

(v) Hence, the side effect of inc () function is changing the data of the external variable 'y'.

4. Differentiate pure and impure function.

Ans. [PTA-3, 6; Mar.-2020]

S. No.	Pure	Impure
(i)	The return value of the pure functions solely depends on its arguments passed.	The return value of the impure functions does not solely depend on its arguments passed.
(ii)	If you call the pure functions with the same set of arguments, you will always get the same return values.	If you call the impure functions with the same set of arguments, you might get the different return values.
(iii)	They do not have any side effects.	They have side effects. For example, random(), Date().
(iv)	They do not modify the arguments which are passed to them	They may modify the arguments which are passed to them

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

Ans. One of the most popular groups of side effects is modifying the variable outside of function.

For example :

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

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

PART - IV

ANSWER THE FOLLOWING QUESTIONS

(5 MARKS)

1. What are called Parameters and write a note on [PTA-2]

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

Ans. **Parameters (and arguments) :** Parameters are the variables in a function definition and arguments are the values which are passed to a function definition.

(i) **Parameter without Type :** Let us see an example of a function, definition :

(requires: $b \geq 0$)
(returns: a to the power of b)

```
let rec pow a b:=
    if b=0 then 1
    else a * pow a (b - 1)
```

- In the above function definition variable 'b' is the parameter and the value which is passed to the variable 'b' is the argument. The precondition (**requires**) and postcondition (**returns**) of the function is given.
- Note we have not mentioned any types: (**data types**). Some language compiler solves this type (**data type**) inference problem algorithmically, but some require the type to be mentioned.
- In the above function definition if expression can return 1 in the then branch, by the **typing** rule the entire if expression has type **int**.
- Since the if expression has type '**int**', the function's return type also be '**int**'. '**b**' is compared to 0 with the equality operator, so '**b**' is also a type of '**int**'. Since 'a' is multiplied with another expression using the * operator, '**a**' must be an int.

(ii) **Parameter with Type :** Now let us write the same function definition with types for some reason:

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



Sura's → XII Std - Computer Science

Unit I - Chapter 1

- When we write the type annotations for 'a' and 'b' the parentheses are mandatory. Generally we can leave out these annotations, because it's simpler to let the compiler infer them.
- There are times we may want to explicitly write down types. This is useful on times when you get a type error from the compiler that doesn't make sense. Explicitly annotating the types can help with debugging such an error message.

2. Identify in the following program [PTA-5]

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

- Name of the function
- Identify the statement which tells it is a recursive function
- Name of the argument variable
- Statement which invoke the function recursively
- Statement which terminates the recursion

- Ans.** (i) gcd
(ii) let rec gcd
(iii) a, b
(iv) gcd b (a mod b)
(v) return a

3. Explain with example Pure and impure functions.

Ans. Pure functions :

- Pure functions are functions which will give exact result when the same arguments are passed.
- For example the mathematical function sin (0) always results 0. This means that every time you call the function with the same arguments, you will always get the same result.
- A function can be a pure function provided it should not have any external variable which will alter the behaviour of that variable.

Let us see an example
let square x
return: x * x

- The above function square is a pure function because it will not give different results for same input.

- There are various theoretical advantages of having pure functions. One advantage is that if a function is pure, then if it is called several times with the same arguments, the compiler only needs to actually call the function once. Let's see an example

```
let i = 0;  
if i < strlen (s) then  
-- Do something which doesn't affect s  
++i
```

- If it is compiled, 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 a pure function and that 's' is not updated in the loop, then it can remove the redundant extra calls to strlen and make the loop to execute only one time.

- From these what we can understand, strlen is a pure function because the function takes one variable as a parameter, and accesses it to find its length. This function reads external memory but does not change it, and the value returned derives from the external memory accessed.

Impure functions :

- 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.

- When a function depends on variables or functions outside of its definition block, you can never be sure that the function will behave the same every time it's called. For example the mathematical function random() will give different outputs for the same function call.

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

- Here the function Random is impure as it is not sure what will be the result when we call the function.

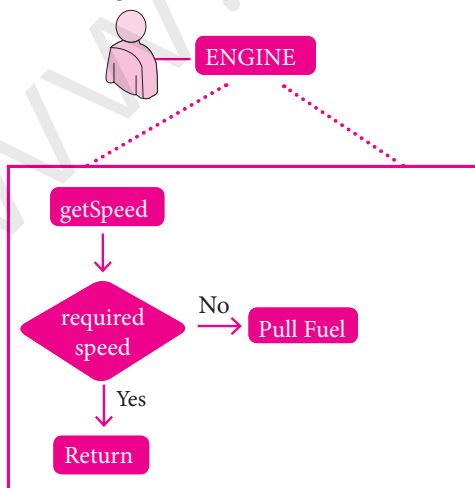
4. Explain with an example interface and implementation.

Ans. Interface :

- (i) An interface is a set of action that an object can do. For example when you press a light switch, the light goes on, you may not have cared how it splashed the light. In Object Oriented Programming language, an Interface is a description of all functions that a class must have in order to be a new interface.
- (ii) In our example, anything that **"ACTS LIKE"** a light, should have function definitions like turn_on () and a turn_off (). The purpose of interfaces is to allow the computer to enforce the properties of the class of **TYPE T** (whatever the interface is) must have functions called X, Y, Z, etc.
- (iii) A class declaration combines the external interface (its local state) with an implementation of that interface (the code that carries out the behaviour). An object is an instance created from the class. The interface defines an object's visibility to the outside world.

Implementation :

- (i) Implementation carries out the instructions defined in the interface.
- (ii) How the object is processed and executed is the implementation.
- (iii) A class declaration combines the external interface (its local state) with an implementation of that interface (the code that carries out the behaviour).
For example, let's take the example of increasing a car's speed.



- (iv) The person who drives the car doesn't care about the internal working. To increase the speed of the car he just presses the accelerator to get the desired behaviour. Here the accelerator is the interface between the driver (the calling / invoking object) and the engine (the called object).
- (v) In this case, the function call would be Speed (70): This is the interface. Internally, the engine of the car is doing all the things. It's where fuel, air, pressure, and electricity come together to create the power to move the vehicle.
- (vi) All of these actions are separated from the driver, who just wants to go faster. Thus we separate interface from implementation.

HANDS ON PRACTICE

1. Write algorithmic function definition to find the minimum among 3 numbers.

Ans. 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

2. Write algorithmic recursive function definition to find the sum of n natural numbers.

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

PTA QUESTIONS AND ANSWERS

1 MARK

1. A function definition which call itself : [PTA-1]
 (a) Pure function (b) Impure function
 (c) Normal function
 (d) Recursive function

[Ans. (d) Recursive function]

3 MARKS

1. Write a function that finds the minimum of its three arguments. [PTA-4; QY-2019]

Ans. 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

GOVERNMENT EXAM QUESTIONS AND ANSWERS

1 MARK

1. Which is the basic building block of computer programs? [Sep-2020]
(a) Argument (b) Parameter
(c) Subroutine (d) Interface

[Ans. (c) Subroutine]

2 MARKS

1. Define pure function. Give one example.

[QY-2019]

Ans. 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

ADDITIONAL QUESTIONS AND ANSWERS

CHOOSE THE CORRECT ANSWER 1 MARK

1. Which of the following are expressed using statements of a programming language?
(a) Functions (b) Algorithm
(c) Interface (d) Implementation
2. What must be used when a bulk of statements to be repeated for many number of times?
(a) Algorithm (b) Program
(c) Subroutines (d) Parameters
3. Which of the following contains a set a code that works on many kinds of input and produces a concrete output?
(a) Function (b) Algorithm
(c) Arguments (d) Language

[Ans. (a) Function]

4. Which of the following are the values which are passed to a function definition?
(a) Parameters (b) Algorithm
(c) Data types (d) Arguments

[Ans. (d) Arguments]

5. The function definition is introduced by the keyword
(a) def (b) rec
(c) let (d) infer

[Ans. (c) let]

6. The recursive function is defined using the keyword

(a) let (b) let rec
(c) name (d) infer

[Ans. (b) let rec]

7. Which of the following is a description of all functions in object oriented programming language?

(a) Implementation (b) parameter
(c) Interface (d) Argument

[Ans. (c) Interface]

8. Which of the following is an instance created from the class?

(a) parameter (b) function
(c) subroutines (d) object

[Ans. (d) object]

9. Which of the following is an example of impure function?

(a) Strlen() (b) random()
(c) sqrf() (d) pure()

[Ans. (b) random()]

10. In which type of function the return type is solely depends on its argument passed?

(a) pure (b) impure
(c) parameterized (d) monochromatize

[Ans. (a) pure]

11. In which type of function the return type does not solely depends on its argument passed?

(a) Pure (b) Parameterized
(c) Impure (d) Monochromatize

[Ans. (c) Impure]

MATCH THE FOLLOWING

1. Match the following function definitions with their terms.

let rec odd xy :=

	List I		List II
i)	Keyword	1)	Xy
ii)	Recursion	2)	Odd
iii)	Function name	3)	Rec
iv)	Parameters	4)	let

(i) (ii) (iii) (iv)
(a) 4 3 2 1
(b) 1 2 3 4
(c) 4 1 2 3
(d) 1 4 2 3

[Ans. (a) (i)-4; (ii)-3; (iii)-2; (iv)-1]



CHOOSE AND FILL IN THE BLANKS

- Subroutines are called as _____.
(a) Algorithm (b) Interface
(c) Parameters (d) Functions
[Ans. (d) Functions]
- _____ are the variables in a function definition.
(a) Arguments (b) Parameters
(c) Identifiers (d) Operators
[Ans. (b) Parameters]
- Explicitly _____ the types can help with debugging.
(a) defining (b) annotating
(c) informing (d) computing
[Ans. (b) annotating]
- All functions are _____ definitions.
(a) static (b) dynamic
(c) algorithmic (d) static
[Ans. (a) static]
- A _____ combines the external interface with an implementation of the interface.
(a) parameter without type
(b) class declaration
(c) function definition
(d) parameter with type
[Ans. (b) class declaration]
- In object oriented programs _____ are the interface
(a) Implementation (b) parameters
(c) Interface (d) Arguments
[Ans. (c) Interface]
- In object oriented programs, how the object is processed and executed is _____.
(a) Implementation (b) Interface
(c) recursion (d) function
[Ans. (a) Implementation]
- Stolen is an example _____ function.
(a) user defined (b) impure
(c) pure (d) recursive
[Ans. (c) pure]
- Evaluation of _____ functions does not cause any side effects to its output?
(a) Impure (b) pure
(c) Recursive (d) built-in
[Ans. (b) pure]

CHOOSE THE CORRECT STATEMENT

- (i) Algorithms are not expressed using statements of a programming language.
(ii) An interface is a set of action that an object can do
(iii) Implementation does not carries out the instructions defined in the interface.
(iv) Pure functions will give exact result.
(a) i and iii (b) ii and iv
(c) iii and ii (d) i, ii and iv
[Ans. (a) i and iii]

VERY SHORT ANSWERS

2 MARKS

- Differentiate parameters and arguments.

Ans. Parameters are the variables in a function definition and arguments are the values which are passed to a function definition.

- Give an example of function definition parameter without type.

Ans. (requires: $b \geq 0$)
(returns: a to the power of b)
let rec pow a b:=
 if b=0 then 1
 else a * pow a (b-1)

- Give an example of function definition parameter with type.

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

- What is recursive function?

Ans. A function definition which call itself is called recursive function.

- Give an example of pure function.

Ans. let square x
 return: $x * x$
let i = 0;
 if i < strlen (s) then
 -- Do something which doesn't affect s
 ++i

- Give an example of impure function.

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



7. Construct an algorithm that arranges meetings between these two types so that they change their color to the third type. In the end, all should display the same color.

Ans. let rec monochromatize a b c :=
if a > 0 then
a, b, c := a-1, b-1, c+2
else
a:=0, b:=0, c:= a + b + c
return c

SHORT ANSWERS

3 MARKS

1. Explain the syntax of function definitions.

Ans. (i) The syntax to define functions is close to the mathematical usage: the definition is introduced by the keyword **let**, followed by the **name** of the function and its arguments; then the formula that computes the image of the argument is written after an = sign. If you want to define a recursive function: use **“let rec”** instead of **“let”**.

- (ii) **Syntax :** The syntax for function definitions:
let rec fn a1 a2 ... an := k

(iii) Here the **“fn”** is a variable indicating an identifier being used as a function name. The names **“a1”** to **“an”** are variables indicating the identifiers used as parameters. The keyword **“rec”** is required if **“fn”** is to be a recursive function; otherwise it may be omitted.

2. Write an algorithm to check whether the entered number is even or odd.

Ans. (requires: x >= 0)
let rec even x :=
x=0 || odd (x-1)
return 'even'
(requires: x >= 0)
let odd x :=
x < 0 && even (x-1)
return 'odd'

3. Write a short note on syntax for function types.

Ans. The syntax for function types :

$x \rightarrow y$
 $x1 \rightarrow x2 \rightarrow y$
 $x1 \rightarrow \dots \rightarrow xn \rightarrow y$

The **“x”** and **“y”** are variables indicating types. The type $x \rightarrow y$ is the type of a function that gets an input of type **“x”** and returns an output of type **“y”**. Whereas $x1 \rightarrow x2 \rightarrow y$ is a type of a function that takes two inputs, the first input is of type **“x1”** and the second input of type **“x2”**, and returns an output of type **“y”**. Likewise $x1 \rightarrow \dots \rightarrow xn \rightarrow y$ has type **“x”** as input of n arguments and **“y”** type as output.



CHAPTER 2

DATA ABSTRACTION

CHAPTER SNAPSHOT

- 2.1 Data Abstraction – Introduction
- 2.2 Abstract Data Types
- 2.3 Constructors and Selectors
- 2.4 Representation of Abstract datatype using Rational numbers
- 2.5 Lists, Tuples
 - 2.5.1 List
 - 2.5.2 Tuple
- 2.6 Data Abstraction in Structure

EVALUATION

PART - I

CHOOSE THE BEST ANSWER (1 MARK)

1. Which of the following functions that build the abstract data type ? *[Sep-2020]*
 - (a) Constructors
 - (b) Destructors
 - (c) Recursive
 - (d) Nested

[Ans. (a) Constructors]
2. Which of the following functions that retrieve information from the data type?
 - (a) Constructors
 - (b) Selectors
 - (c) Recursive
 - (d) Nested

[Ans. (b) Selectors]
3. The data structure which is a mutable ordered sequence of elements is called
 - (a) Built in
 - (b) List
 - (c) Tuple
 - (d) Derived data

[Ans. (b) List]

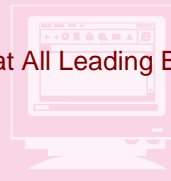
4. A sequence of immutable objects is called *[Mar.-2020]*
 - (a) Built in
 - (b) List
 - (c) Tuple
 - (d) Derived data

[Ans. (c) Tuple]
5. The data type whose representation is known are called *[PTA-2; QY-2019]*
 - (a) Built in datatype
 - (b) Derived datatype
 - (c) Concrete datatype
 - (d) Abstract datatype

[Ans. (c) Concrete datatype]
6. The data type whose representation is unknown are called
 - (a) Built in datatype
 - (b) Derived datatype
 - (c) Concrete datatype
 - (d) Abstract datatype

[Ans. (d) Abstract datatype]
7. Which of the following is a compound structure?
 - (a) Pair
 - (b) Triplet
 - (c) Single
 - (d) Quadrant

[Ans. (a) Pair]



8. Bundling two values together into one can be considered as [Govt. MQP - 2019; PTA-4]

- (a) Pair (b) Triplet
(c) Single (d) Quadrant

[Ans. (a) Pair]

9. Which of the following allow to name the various parts of a multi-item object? [PTA-6]

- (a) Tuples (b) Lists
(c) Classes (d) Quadrants

[Ans. (c) Classes]

10. Which of the following is constructed by placing expressions within square brackets?

- (a) Tuples (b) Lists
(c) Classes (d) Quadrants

[Ans. (b) Lists]

PART - II

ANSWER THE FOLLOWING QUESTIONS (2 MARKS)

1. What is abstract data type?

- Ans. (i)** Abstract Data type (ADT) is a type (or class) for objects whose behavior is defined by a set of value and a set of operations.
(ii) The definition of ADT only mentions what operations are to be performed but not how these operations will be implemented.

2. Differentiate constructors and selectors.

Ans. [PTA-2, 3; QY-2019]

S. No.	Constructors	Selectors
(i)	Constructors are functions that build the abstract data type.	Selectors are functions that retrieve information from the data type.
(ii)	Constructors create an object, bundling together different pieces of information.	Selectors extract individual pieces of information from the object

3. What is a Pair? Give an example. [Mar.-2020]

- Ans. (i)** Any way of bundling two values together into one can be considered as a Pair. Lists are a common method to do so. Therefore List can be called as Pairs.

(ii) Example : List = [(10,10), (1,20)]

4. What is a List? Give an example. [QY - 2019]

Ans. (i) List is constructed by placing expressions within square brackets separated by commas.

(ii) 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.

Example : lst := [10, 20]

x, y := lst

5. What is a Tuple? Give an example.

Ans. (i) A tuple is a comma-separated sequence of values surrounded with parentheses. Tuple is similar to a list.

(ii) The difference between the two is that you cannot change the elements of a tuple once it is assigned whereas in a list, elements can be changed.

(iii) Example : colour= ('red', 'blue', 'Green')

PART - III

ANSWER THE FOLLOWING QUESTIONS

(3 MARKS)

1. Differentiate Concrete datatype and Abstract datatype.

Ans.

S. No.	Concrete datatype	Abstract datatype
(i)	Concrete datatypes or structures (CDT's) are direct implementations of a relatively simple concept.	Abstract Datatypes (ADT's) offer a high level view (and use) of a concept independent of its implementation.
(ii)	A concrete data type is a data type whose representation is known.	Abstract data type the representation of a data type is unknown.

2. Which strategy is used for program designing? Define that Strategy. [Govt. MQP-2019]

Ans. A powerful strategy for designing programs: '**wishful thinking**'. Wishful Thinking is the formation of beliefs and making decisions according to what might be pleasing to imagine instead of by appealing to reality.



3. Identify Which of the following are constructors and selectors? [PTA-5]

- (a) N1=number()
- (b) accetnum(n1)
- (c) displaynum(n1)
- (d) eval(a/b)
- (e) x,y= makeslope (m), makeslope(n)
- (f) display()

Ans. (a) Constructors
(b) Selectors
(c) Selectors
(d) Selectors
(e) Constructors
(f) Selectors

4. What are the different ways to access the elements of a list. Give example.

Ans. (i) The elements of a list can be accessed in two ways. The first way is via our familiar method of multiple assignment, which unpacks a list into its elements and binds each element to a different name.

lst := [10, 20]

x, y := lst

(ii) In the above example x will become 10 and y will become 20.

(iii) A second method for accessing the elements in a list is by the element selection operator, also expressed using square brackets. Unlike a list literal, a square-brackets expression directly following another expression does not evaluate to a list value, but instead selects an element from the value of the preceding expression.

lst[0]

10

lst[1]

20

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

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

Ans. (a) List
(b) Tuple
(c) Class
(d) Tuple
(e) List
(f) Class

PART - IV

ANSWER THE FOLLOWING QUESTIONS

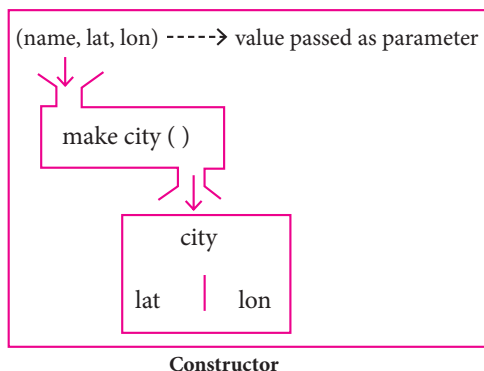
(5 MARKS)

1. How will you facilitate data abstraction. Explain it with suitable example. [PTA-2, 4]

Ans. Data abstraction is supported by defining an abstract data type (ADT), which is a collection of constructors and selectors. To facilitate data abstraction, you will need to create two types of functions: **Constructors, Selectors**

Constructors :

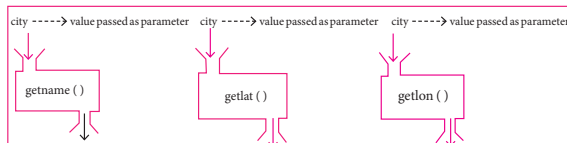
- (i) Constructors are functions that build the abstract data type.
- (ii) Constructors create an object, bundling together different pieces of information.
- (iii) For example, say you have an abstract data type called city.
- (iv) This city object will hold the city's name, and its latitude and longitude.
- (v) To create a city object, you'd use a function like **city = makecity (name, lat, lon)**.
- (vi) Here makecity (name, lat, lon) is the constructor which creates the object city.



Selectors :

- (i) Selectors are functions that retrieve information from the data type.
- (ii) Selectors extract individual pieces of information from the object.

- (iii) To extract the information of a city object, you would use functions like
`getname(city)`
`getlat(city)`
`getlon(city)`
 These are the selectors because these functions extract the information of the city object.



2. What is a List? Why List can be called as Pairs. Explain with suitable example. [PTA-6]

Ans. List :

- (i) 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.
 Example for List is [10, 20].

- (ii) The elements of a list can be accessed in two ways. The first way is via our familiar method of multiple assignment, which unpacks a list into its elements and binds each element to a different name.

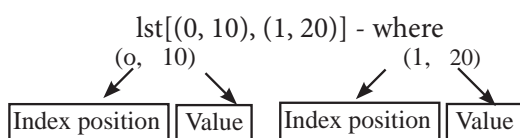
```
lst := [10, 20]
x, y := lst
```

- (iii) In the above example x will become 10 and y will become 20. A second method for accessing the elements in a list is by the element selection operator, also expressed using square brackets.

- (iv) Unlike a list literal, a square-brackets expression directly following another expression does not evaluate to a list value, but instead selects an element from the value of the preceding expression.

```
lst[0]
10
lst[1]
20
```

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



Pair :

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

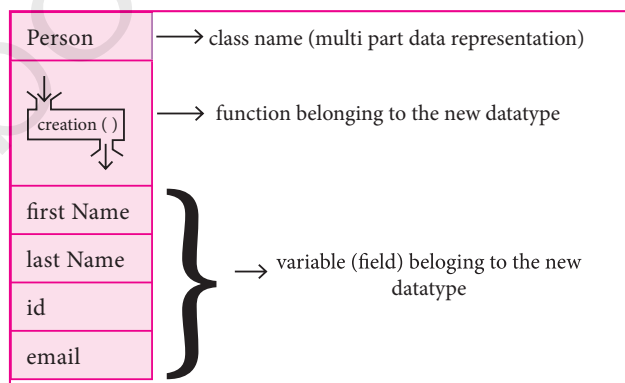
3. How will you access the multi-item? Explain with example.

Ans. (i) The structure construct (In OOP languages it's called class construct) is used to represent multi-part objects where each part is named (given a name). Consider the following pseudo code:

class Person:

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

The new data type Person is pictorially represented as



Let main() contains

<code>p1:=Person()</code>	statement creates the object
<code>firstName := "Padmashri"</code>	setting a field called first Name with value Padmashri
<code>lastName := "Baskar"</code>	setting a field called lastName with value Baskar
<code>id := "994-222-1234"</code>	setting a field called id value 994-222-1234
<code>email="compisci@gamil.com"</code>	setting a field called email with value compisci@gmail.com

- - output of firstName : Padmashri



- (ii) The class (structure) construct defines the form for multi-part objects that represent a person.
- (iii) Person is referred to as a class or a type, while p1 is referred to as an object or an instance.
- (iv) Here class Person as a cookie cutter, and p1 as a particular cookie. Using the cookie cutter you can make many cookies. Same way using class created many objects of that type.
- (v) A class defines a data abstraction by grouping related data items. A class is not just data, it has functions defined within it. We say such functions are subordinate to the class because their job is to do things with the data of the class.

PTA QUESTIONS AND ANSWERS

1 MARK

1. **Expansion of ADT :** [PTA-1]
 (a) Abstract Data Tuple
 (b) All Data Template
 (c) Abstract Data Type
 (d) Application Data Type

[Ans. (c) Abstract Data Type]

2. **ADT can be implemented using ____.** [PTA-5]
 (a) singly linked list (b) doubly linked list
 (c) either A or B (d) neither A nor B

[Ans. (a) singly linked list]

GOVERNMENT EXAM QUESTIONS AND ANSWERS

1 MARK

1. **The datatype whose representation is unknown is called** [HY-2019]
 (a) Built-in datatype (b) Derived datatype
 (c) Concrete datatype (d) Abstract datatype

[Ans. (d) Abstract datatype]

3 MARKS

1. (a) **What is selector?** [Sep-2020]
 (b) **What are the parts of a program?**
Ans. a) Selectors are nothing but the functions that retrieve information from the data type. Therefore in the above code

- (i) getname(city)
- (ii) getlat(city)

- (iii) getlon(city)
are the selectors because these functions extract the information of the city object
- b) The two parts of a program are, the part that operates on abstract data and the part that defines a concrete representation, is connected by a small set of functions that implement abstract data in terms of the concrete representation.

ADDITIONAL QUESTIONS AND ANSWERS

CHOOSE THE CORRECT ANSWER 1 MARK

1. **Which of the following is a powerful concept that allows programmers to treat codes as objects?**

- (a) Encapsulation (b) Data Abstraction
- (c) Inheritance (d) Polymorphism

[Ans. (b) Data Abstraction]

2. **Which of the following provides modularity?**

- (a) Datatypes (b) Subroutines
- (c) Classes (d) Abstraction

[Ans. (d) Abstraction]

3. **Which of the following is a type for objects whose behavior is defined by a set of value and a set of operations?**

- (a) User-defined datatype
- (b) Derived datatype
- (c) Built-in datatype (d) Abstract datatype

[Ans. (d) Abstract datatype]

4. **ADT behavior is defined by**

- (i) Set of Variables (ii) Set of Value
- (iii) Set of Functions (iv) Set of Operations
- (a) i, ii (b) ii, iii
- (c) ii, iv (d) i, iii

[Ans. (c) ii, iv]

5. **The process of providing only the essentials and hiding the details is known as**

- (a) Functions (b) Abstraction
- (c) Encapsulation (d) Pairs

[Ans. (b) Abstraction]

6. **Which of the following gives an implementation independent view?**

- (a) Abstract (b) Concrete
- (c) Datatype (d) Behavior of an object

[Ans. (a) Abstract]



- 7. How many ways to implement an ADT?**
(a) Only one (b) Two
(c) Three (d) Many
[Ans. (d) Many]
- 8. Which of the following are implemented using & lists?**
(a) Singly linked list ADT
(b) Doubly Linked list ADT
(c) Stack ADT (d) Queue ADT
(e) All of these **[Ans. (e) All of these]**
- 9. Which of the following replicate how we think about the world?**
(a) Queue ADT (b) Data Hiding
(c) Data Abstraction (d) Stack ADT
[Ans. (c) Data Abstraction]
- 10. To facilitate data abstraction, How many types of functions are created?**
(a) 2 (b) 3
(c) 4 (d) Only one
[Ans. (a) 2]
- 11. Which of the following function that facilitate the data abstraction?**
(a) Constructors (b) Destructors
(c) Selectors (d) a and c
[Ans. (d) a and c]
- 12. Which of the following are functions that build the abstract datatype?**
(a) Constructors (b) Destructors
(c) Selectors (d) All of these
[Ans. (a) Constructors]
- 13. Which of the following extract the information of the object?**
(a) Constructors (b) Functions
(c) Selectors (d) Destructors
[Ans. (c) Selectors]
- 14. In which data representation, a definition for each function is known.**
(a) User defined (b) Buil-in
(c) Abstract (d) Concrete
[Ans. (d) Concrete]
- 15. How many parts are there in the program?**
(a) 2 (b) 3
(c) 4 (d) Many
[Ans. (a) 2]
- 16. To implement the concrete level of data abstraction the language python provides a compound structure called**
(a) ADT (b) Concrete data
(c) Pair
(d) User defined function **[Ans. (c) Pair]**
- 17. Which of the following is contracted by placing expressions within square brackets separated by commas?**
(a) List (b) Tuple
(c) Set (d) Dictionary
[Ans. (a) List]
- 18. How many values can be stored in the list?**
(a) 4 (b) 10
(c) 100 (d) Multiple
[Ans. (d) Multiple]
- 19. l = [10, 20] is an example**
(a) Tuple (b) Set
(c) List (d) Dictionary
[Ans. (c) List]
- 20. List can also be called as**
(a) Functions (b) Class
(c) Structure (d) Pairs
[Ans. (d) Pairs]
- 21. How many ways are there to represent pair datatype?**
(a) 2 (b) 4 (c) 3 (d) 5
[Ans. (a) 2]
- 22. Color = ('red', 'green', 'blue') is an example of**
(a) Dictionary (b) List
(c) Set (d) Tuple
[Ans. (d) Tuple]
- 23. Which of the following does not allow us to name the various parts of a multi-item object?**
(a) List (b) Tuple
(c) Pair (d) All of these
[Ans. (d) All of these]
- 24. Which of the following defines a data abstraction by grouping related data items?**
(a) List (b) Pair
(c) Class (d) Tuple
[Ans. (c) Class]
- 25. Which of the following as bundled data and the functions that work on that data?**
(a) Object (b) Pair
(c) List (d) Class
[Ans. (d) Class]



26. CDT expansion is

- (a) Collective Data Type (b) Class Data Type
(c) Concrete Data Type
(d) Central Data Type

[Ans. (b) Class Data Type]

MATCH THE FOLLOWING

1.	List I	List II
i)	List	1) arr (1,2,3,4)
ii)	Tuples	2) getname (city)
iii)	Class	3) Student [rno, name, mark]
iv)	Selectors	4) arr [1,2,3,4]

- | | | | | |
|-----|-----|------|-------|------|
| | (i) | (ii) | (iii) | (iv) |
| (a) | 1 | 2 | 3 | 4 |
| (b) | 4 | 3 | 2 | 1 |
| (c) | 4 | 3 | 2 | 1 |
| (d) | 3 | 2 | 4 | 1 |

[Ans. (c) (i)-4; (ii)-3; (iii)-2; (iv)-1]

CHOOSE THE ODD MAN OUT

1. (a) List
(b) Multiple assignment
(c) Classes
(d) Element selection operator

[Ans. (c) Classes]

CHOOSE AND FILL IN THE BLANKS

1. Data Abstraction allows programmers to treat code as ____.

- (a) Objects (b) Classes
(c) Members (d) Parameters

[Ans. (a) Objects]

2. _____ are the representation for Abstract Data types.

- (a) Objects (b) Classes
(c) Functions (d) Lists

[Ans. (b) Classes]

3. Classes are the representation for ____

- (a) Abstract datatype
(b) Built-in datatype
(c) Concrete datatype
(d) Essential datatype

[Ans. (a) Abstract datatype]

4. The _____ can be implemented using singly linked list or doubly linked list.

- (a) Tuple ADT (b) List ADT
(c) Function ADT (d) List ADT

[Ans. (b) List ADT]

5. The basic idea of _____ is to structure programs so that they operate on abstract data (a)

- (a) Encapsulation (b) Polymorphism
(c) Data type (d) Data Abstraction

[Ans. (d) Data Abstraction]

6. A _____ data representation is defined as an independent part of the program.

- (a) Abstract (b) Concrete
(c) List (d) Tuple

[Ans. (b) Concrete]

7. _____ are functions that retrieve information from the data type.

- (a) Constructors (b) Selectors
(c) List (d) Tuples

[Ans. (b) Selectors]

8. _____ is made up of list or Tuples.

- (a) Set (b) Pair
(c) Dictionary
(d) Control Structures

[Ans. (b) Pair]

9. List is constructed by using _____ and _____.

- (a) (), , (b) <>, ;
(c) [], , (d) [], :

[Ans. (c) [], ,]

10. A _____ is a comma separated values surround with parentheses.

- (a) List (b) Tuple
(c) Set (d) Dictionary

[Ans. (b) Tuple]

11. Tuple is constructed by using _____ and _____

- (a) (), (b) [], (c) []:, (d) (),:

[Ans. (a) (),]

12. A _____ is not just data, it has functions defined within it.

- (a) Class (b) List
(c) Pair (d) Object

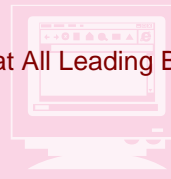
[Ans. (a) Class]

CHOOSE THE INCORRECT STATEMENT

1. (i) ADT is defined by set of values and set of operations
(ii) ADT does specify how data will be organized in the memory.
(iii) Constructors are not used to built abstract data type.
(iv) Selectors are functions that retrieve information from the data type.

- (a) i, ii (b) ii, iv
(c) ii, iii (d) i, iii, iv

[Ans. (c) ii, iii]



CHOOSE THE INCORRECT PAIR

1. (a) Abstraction – hiding the details
(b) Abstract data type–constructor & destructor
(c) Abstraction – providing only the essentials
(d) Abstract data type – constructor & selectors

[Ans. (b) Abstract data type – Constructor & destructor]

VERY SHORT ANSWERS

2 MARKS

1. Give an example of implementing an ADT.

Ans. (i) There can be different ways to implement an ADT, for example, the List ADT can be implemented using singly linked list or doubly linked list.

- (i) Similarly, stack ADT and Queue ADT can be implemented using lists.

2. Identify which is the constructor and selector from the following statement.

- (i) The Functions that retrieve information from the datatype
- (ii) The function which creates an object.

Ans. (i) Selector

- (ii) Constructor

3. Write the pseudo code for the representation of the rational number.

Ans. The pseudo code for the representation of the rational number

```
x,y:=8,3
rational(n,d)
numer(x)/numer(y)
- - output :
```

4. How the concrete level of data abstraction implemented?

Ans. (i) To implement the concrete level of data abstraction, languages like Python provides a compound structure called Pair which is made up of list or Tuple.

- (ii) The first way to implement pairs is with the List construct.

5. Write a note on pair datatype.

Ans. (i) A pair is a compound data type that holds two other pieces of data. The two ways of representing the pair data type.

- (ii) The first way is using List construct and the second way to implement pairs is with the tuple construct.

6. Write a pseudocode to depressant rational numbers using list.

Ans. rational(n, d):

```
return [n, d]
```

```
numer(x):
```

```
return x[0]
```

```
denom(x):
```

```
return x[1]
```

7. How a class defines a data abstraction?

Ans. (i) A class defines a data abstraction by grouping related data items. A class is not just data, it has functions defined within it.

- (ii) Functions are subordinate to the class because their job is to do things with the data of the class.

8. From the statement P1 := Preson(), What does P1 and person referred.

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

9. How the elements of a list can be accessed?

Ans. (i) The elements of a list can be accessed in two ways.

- (ii) The first way is via multiple assignment and the second method is by the element selection operator.

SHORT ANSWERS

3 MARKS

1. Identify the constructor and selector from the following.

- (i) City = Make city (name, lat, lon)
- (ii) Get name (city)
- (iii) Make point (x,y)
- (iv) x coord (point)
- (v) y coord (point)

Ans. (i) Constructor

- (ii) Selector

- (iii) Constructor

- (iv) Selector

- (v) Selector

2. Write a note on Data Abstraction.

Ans. (i) Data abstraction is supported by defining an abstract data type (ADT), which is a collection of constructors and selectors.

- (ii) Constructors create an object, bundling together different pieces of information, while selectors extract individual pieces of information from the object.



3. Give an example of an ADT for rational numbers.

Ans. An ADT for rational numbers :

- - constructor
- - constructs a rational number with numerator n, denominator d

rational(n, d)

- - selector

numerator(x) → returns the numerator of rational number x

denominator(y) → returns the denominator of rational number y

LONG ANSWERS

5 MARKS

1. Explain the representation of Abstract datatype using rational numbers.

- Ans. (i)** The basic idea of data abstraction is to structure programs so that they operate on abstract data. That is, our programs should use data in such a way, as to make as few assumptions about the data as possible.
- (ii)** At the same time, a concrete data representation is defined as an independent part of the program.
- (iii)** Any program consist of two parts. The two parts of a program are, the part that operates on abstract data and the part that defines a concrete representation, is connected by a small set of functions that implement abstract data in terms of the concrete representation.

(iv) To illustrate this technique, let us consider an example to design a set of functions for manipulating rational numbers.

(v) Example : A rational number is a ratio of integers, and rational numbers constitute an important sub-class of real numbers. A rational number such as $\frac{8}{3}$ or $\frac{19}{23}$ is typically written as :

$\frac{\langle \text{numerator} \rangle}{\langle \text{denominator} \rangle}$

(vi) where both the $\langle \text{numerator} \rangle$ and $\langle \text{denominator} \rangle$ are placeholders for integer values. Both parts are needed to exactly characterize the value of the rational number. Actually dividing integers produces a float approximation, losing the exact precision of integers.

(vii) However, you can create an exact representation for rational numbers by combining together the numerator and denominator.

(viii) As we know from using functional abstractions, we can start programming productively before you have an implementation of some parts of our program.

(ix) Let us begin by assuming that you already have a way of constructing a rational number from a numerator and a denominator. You also assume that, given a rational number, you have a way of selecting its numerator and its denominator component.



CHAPTER 3

SCOPING

CHAPTER SNAPSHOT

- | | |
|---|--|
| <ul style="list-style-type: none">3.1 Introduction3.2 Variable Scope3.3 LEGB rule3.4 Types of Variable Scope<ul style="list-style-type: none">3.4.1. Local Scope3.4.2. Global Scope3.4.3. Enclosed Scope3.4.4. Built-in-Scope | <ul style="list-style-type: none">3.5 Module<ul style="list-style-type: none">3.5.1. Characteristics of Modules3.5.2. The benefits of using modular programming include3.5.3. Access Control |
|---|--|

EVALUATION

PART - I

CHOOSE THE BEST ANSWER (1 MARK)

1. Which of the following refers to the visibility of variables in one part of a program to another part of the same program.
(a) Scope (b) Memory
(c) Address (d) Accessibility
[Ans. (a) Scope]
2. The process of binding a variable name with an object is called [Sep-2020]
(a) Scope (b) Mapping
(c) late binding (d) early binding
[Ans. (b) Mapping]
3. Which of the following is used in programming languages to map the variable and object? [PTA-2; HY-2019]
(a) :: (b) :=
(c) = (d) ==
[Ans. (c) =]
4. Containers for mapping names of variables to objects is called [QY-2019]
(a) Scope (b) Mapping
(c) Binding (d) Namespaces
[Ans. (d) Namespaces]
5. Which scope refers to variables defined in current function?
(a) Local Scope (b) Global scope
(c) Module scope (d) Function Scope
[Ans. (a) Local Scope]
6. The process of subdividing a computer program into separate sub-programs is called
(a) Procedural Programming
(b) Modular programming
(c) Event Driven Programming
(d) Object oriented Programming
[Ans. (b) Modular programming]
7. Which of the following security technique that regulates who can use resources in a computing environment?
(a) Password (b) Authentication
(c) Access control (d) Certification
[Ans. (c) Access control]
8. Which of the following members of a class can be handled only from within the class? [Mar.-2020]
(a) Public members
(b) Protected members
(c) Secured members
(d) Private members
[Ans. (d) Private members]



9. Which members are accessible from outside the class?

- (a) Public members
- (b) Protected members
- (c) Secured members
- (d) Private members

[Ans. (a) Public members]

10. The members that are accessible from within the class and are also available to its sub-classes is called [PTA-6]

- (a) Public members
- (b) Protected members
- (c) Secured members
- (d) Private members

[Ans. (b) Protected members]

PART - II

ANSWER THE FOLLOWING QUESTIONS (2 MARKS)

1. What is a scope?

Ans. Scope refers to the visibility of variables, parameters and functions in one part of a program to another part of the same program.

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

Ans. Essentially, variables are addresses (references, or pointers), to an object in memory. When you assign a variable with := to an instance (object), you're binding (or mapping) the variable to that instance. Multiple variable can be mapped to the same instance.

3. What is Mapping? [PTA-5]

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

4. What do you mean by Namespaces?
[Govt. MQP-2019; PTA-4; Mar.-2020]

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

Example : a := 5

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

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

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

PART - III

ANSWER THE FOLLOWING QUESTIONS

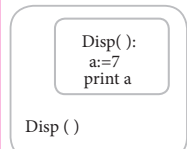
(3 MARKS)

1. Define Local scope with an example.

Ans. (i) Local scope refers to variables defined in current function. Always, a function will first look up for a variable name in its local scope.

(ii) Only if it does not find it there, the outer scopes are checked.

(iii) Look at this example :

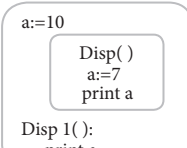
1. Disp(): 2. a:=7 3. print a 4. Disp()	Entire program 	Output of the Program 7
--	--	----------------------------

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

2. Define Global scope with an example. [PTA-6]

Ans. (i) A variable which is declared outside of all the functions in a program is known as Global variable.

(ii) This means, global variable can be accessed inside or outside of all the functions in a program. Consider the following example

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

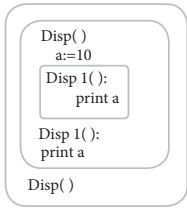
(iii) On execution of the above code the variable a which is defined inside the function Disp() displays the value 7 for the function call Disp() and then it displays 10, because a is defined in global scope.



3. Define Enclosed scope with an example.

[PTA-3]

- Ans. (i)** All programming languages permit functions to be nested. A function (method) within another function is called nested function.
- (ii)** A variable which is declared inside a function which contains another function definition with in it, the inner function can also access the variable of the outer function. This scope is called enclosed scope.
- (iii)** When a compiler or interpreter search for a variable in a program, it first search Local, and then search Enclosing scopes. Consider the following example

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

4. Why access control is required?

[PTA-1; HY-2019]

- Ans. (i)** Access control is a security technique that regulates who or what can view or use resources in a computing environment.
- (ii)** It is a fundamental concept in security that minimizes risk to the object.
- (iii)** In other words access control is a selective restriction of access to data.
- (iv)** In oops Access control is implemented through access modifiers.
- 5. Identify the scope of the variables in the following pseudo code and write its output**
- ```
color:= Red
mycolor():
b:=Blue
lue
myfavcolor():
g:=Green
printcolor, b, g
myfavcolor()
printcolor, b
mycolor()
print color
```

**Ans. Output :**

Red Blue Green  
Red Blue  
Red

**Scope of Variables :**

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

**PART - IV**

**ANSWER THE FOLLOWING QUESTIONS**

**(5 MARKS)**

**1. Explain the types of scopes for variable or LEGB rule with example. [PTA-1; Sep-2020]**

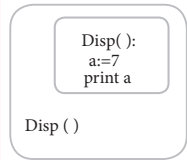
**Ans. Types of Variable Scope :**

There are 4 types of Variable Scope, let's discuss them one by one:

**Local Scope :**

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

Look at this example

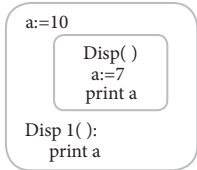
| 1. Disp():<br>2. a:=7<br>3. print a<br>4. Disp() | Entire program<br> | Output<br>of the<br>Program<br>7 |
|--------------------------------------------------|---------------------------------------------------------------------------------------------------------|----------------------------------|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

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

**Global Scope:**

- (i)** A variable which is declared outside of all the functions in a program is known as global variable.
- (ii)** This means, global variable can be accessed inside or outside of all the functions in a program. Consider the following example

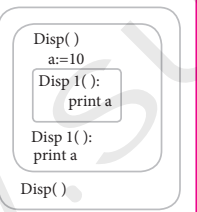


|                                                                            |                                                                                                     |                                  |
|----------------------------------------------------------------------------|----------------------------------------------------------------------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

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

#### Enclosed Scope :

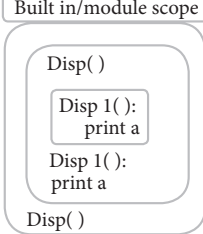
- (i) All programming languages permit functions to be nested. A function (method) within another function is called nested function.
- (ii) A variable which is declared inside a function which contains another function definition within it, the inner function can also access the variable of the outer function. This scope is called enclosed scope.
- (iii) When a compiler or interpreter searches for a variable in a program, it first searches Local, and then searches Enclosing scopes. Consider the following example

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

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

#### Built-in Scope :

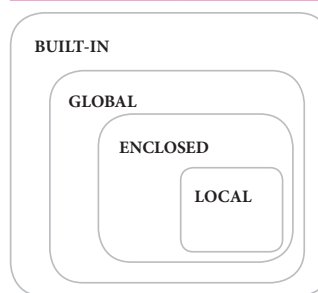
- (i) The built-in scope has all the names that are pre-loaded into the program scope when we start the compiler or interpreter.
- (ii) Any variable or module which is defined in the library functions of a programming language has Built-in or module scope. Consider the following example.

|                                                                                                      |                                            |
|------------------------------------------------------------------------------------------------------|--------------------------------------------|
| Entire program<br> | Library files associated with the software |
|------------------------------------------------------------------------------------------------------|--------------------------------------------|

#### LEGB rule :

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

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



#### 2. Write any Five Characteristics of Modules.

[PTA-4, 6; HY-2019; Sep-2020]

**Ans.** The following are the desirable characteristics of a module.

- 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.
- Module segments can be used by other modules.



## Sura's XII Std - Computer Science

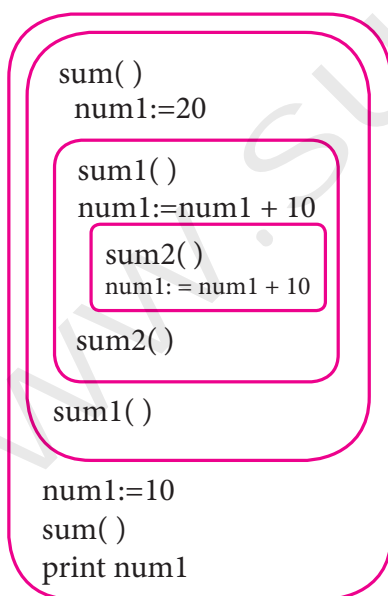
### Unit I - Chapter 3

#### 3. Write any five benefits in using modular programming. [Govt. MQP-2019]

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

### HANDS ON PRACTICE

#### 1. Observe the following diagram and Write the pseudo code for the following.



**Ans. sum():**  
 num 1:=20  
 sum1()  
 num1:= num1 + 10

```

sum2()
 num1:= num1 + 10
sum2()
sum1()
num1:= 10
sum()
Print num 1

```

### PTA QUESTIONS AND ANSWERS

#### 1 MARK

#### 1. A variable which is declared inside a function which contains another function definition : [PTA-1]

- (a) Local (b) Global  
(c) Enclosed (d) Built-in

[Ans. (c) Enclosed]

#### 2. Which are loaded as soon as the library files are imported to the program? [PTA-3]

- (a) Built-in scope variables  
(b) Enclosed scope variables  
(c) Global scope variables  
(d) Local scope variables

[Ans. (a) Built-in scope variables]

#### 3. Which of the following is not the example of modules? [PTA-5]

- (a) procedures (b) subroutines  
(c) class (d) functions

[Ans. (c) class]

#### 2 MARKS

#### 1. What are modules? [PTA-4]

**Ans.** A module is a part of a program. Programs are composed of one or more independently developed modules.

### GOVERNMENT EXAM QUESTIONS AND ANSWERS

#### 1 MARK

#### 1. The kind of scope of the variable 'a' used in the pseudo code given below. [Govt. MQP-2019]

- (a) Disp(): (b) a: = 7  
(c) print a (d) Disp()  
(a) Local (b) Global  
(c) Enclosed (d) Built-in

[Ans. (a) Local]



2. The SQL command to make a database as current active database is [Govt. MQP-2019]  
(a) CURRENT (b) USE  
(c) DATABASE (d) NEW

[Ans. (b) USE]

### 2 MARKS

1. What is LEGB rule? [QY-2019]

**Ans.** Scope also defines the order in which variables have to be mapped to the object in order to obtain the value.

### ADDITIONAL QUESTIONS AND ANSWERS

#### CHOOSE THE CORRECT ANSWER 1 MARK

1. The part of a program that can see or use the variables are called  
(a) Scope (b) Parameter  
(c) Function (d) Indentation

[Ans. (a) Scope]

2. Which of the following refers to the addresses to an object in memory?  
(a) Functions (b) Indentation  
(c) Variables (d) Operators

[Ans. (b) Indentation]

3. How many variables can be mapped to the same instance?  
(a) 2 (b) 3  
(c) 4 (d) Multiple

[Ans. (d) Multiple]

4. Which of the following keeps track of all these mappings with namespaces?  
(a) Programming languages  
(b) Application software  
(c) System software  
(d) My SQL

[Ans. (a) Programming languages]

5. How the names are mapped with objects in programming language?  
(a) name == object (b) name :: object  
(c) name := object (d) object := name

[Ans. (c) name := object]

6. The order in which variables have to be mapped to the object in order to obtain the value is called

- (a) Rule (b) Syntax  
(c) Scope (d) Hierarchy

[Ans. (c) Scope]

7. Which of the following rule is used to decide the order in which the scopes are to be searched for scope resolution?

- (a) LEGB (b) LGEB  
(c) LBEG (d) LGBE

[Ans. (a) LEGB]

8. Write the below interns of hierarchy (highest to lowest)?

- (1) Reversed names in built in functions  
(2) Defined inside function  
(3) Defined inside enclosing function  
(4) Defined at the uppermost level

- (a) 3, 2, 1, 4 (b) 1, 4, 2, 3  
(c) 2, 3, 1, 4 (d) 2, 3, 4, 1

[Ans. (d) 2, 3, 4, 1]

9. How many types of variable scope are there?

- (a) 2 (b) 4 (c) 3 (d) 6

[Ans. (b) 4]

10. Which of the following is not a variable scope?

- (a) Global (b) Enclosed  
(c) List (d) Built-in

[Ans. (c) List]

11. Choose the type of scope for a variable 'a' defined in the following program.

Disp () :

a := 7

Print a

Disp ()

- (a) Global (b) Enclosed  
(c) Local (d) Built-in

[Ans. (c) Local]

12. A variable which is declared outside all the functions in a program is known as

- (a) Local (b) Enclosed  
(c) Extern (d) Global

[Ans. (d) Global]



**13. Which of the following variable can be accessed inside or outside of all the functions in a program?**

- (a) Local (b) Global
- (c) Enclosed (d) Built-in

**[Ans. (b) Global]**

**14. What is the output of the statement in the following program?**

```
X := 10
Disp () :
a := 7
print a
Displ () :
Print a
```

- (a) 710 (b) 107 (c) 7 (d) 10

**[Ans. (d) 10]**

**15. Which of the following can ease the job of programming and debugging the program?**

- (a) Statements (b) Interaction
- (c) Modules (d) Scopes

**[Ans. (c) Modules]**

**16. Which of the following programming enables programmers to divide up the work and retry pieces of the program independently?**

- (a) Modular Programming
- (b) Procedural Programming
- (c) Object Oriented Programming
- (d) Structural Programming

**[Ans. (a) Modular Programming]**

**17. The example of modules are**

- (a) Procedures (b) Subroutines
- (c) Functions (d) All of these

**[Ans. (d) All of these]**

**18. Which of the following contain instructions, processing logic and data?**

- (a) Scopes (b) Modules
- (c) Indentation (d) Access control

**[Ans. (b) Modules]**

**19. The following are the type of variable scopes Find the odd one out**

- (a) Local (b) Enclosed
- (c) Global (d) Protected

**[Ans. (d) Protected]**

**20. Which of the following members of a class are denied access from outside the class?**

- (a) Private (b) Protected
- (c) Public (d) Enclosed

**[Ans. (a) Private]**

**21. Which of the following is not a classical object oriented language?**

- (a) C++ (b) Java
- (c) Python (d) C **[Ans. (d) C]**

**22. Which of the following keywords are not used to control the access to class members?**

- (a) Public (b) Protected
- (c) Public (d) Global

**[Ans. (d) Global]**

**23. How many access control keywords are there?**

- (a) 2 (b) 3 (c) 4 (d) 6

**[Ans. (b) 3]**

**24. Find the odd man out**

- (a) Public (b) Local
- (c) Protected (d) Private

**[Ans. (b) Local]**

**25. The arrangement of private instance variables and public methods ensures the principle of**

- (a) Inheritance (b) Polymorphism
- (c) Encapsulation (d) Abstraction

**[Ans. (c) Encapsulation]**

**26. Which of the following members of a class are accessible from within the class and available to its subclass?**

- (a) Private (b) Protected
- (c) Public (d) All of these

**[Ans. (b) Protected]**

**27. By default, the Python. class members are**

- (a) Private (b) Protected
- (c) Global (d) Public

**[Ans. (d) Public]**

**28. By default, the C++ and Java class members are**

- (a) Private (b) Protected
- (c) Public (d) Local

**[Ans. (a) Private]**



**29. Programs are composed of one or more independently developed**

- (a) Access control (b) Encapsulation  
(c) Modules  
(d) Members of a class [Ans. (c) Modules]

**MATCH THE FOLLOWING**

| 1.   | List I         | List II                    |
|------|----------------|----------------------------|
| i)   | Scope          | 1) Mapping names           |
| ii)  | Name spaces    | 2) Visibility of variables |
| iii) | Module         | 3) Security technique      |
| iv)  | Access control | 4) Sub dividing program    |

- (i) (ii) (iii) (iv)  
(a) 2 1 4 3  
(b) 3 1 4 2  
(c) 2 4 1 3  
(d) 3 2 4 1

[Ans. (a) (i)-2; (ii)-1; (iii)-4; (iv)-3]

**CHOOSE AND FILL IN THE BLANKS**

**1. Scope refers to the visibility of \_\_\_\_\_**

- (a) Variables (b) Parameters  
(c) Functions (d) All of these

[Ans. (d) All of these]

**2. The duration for which a variable is alive is called its \_\_\_\_\_.**

- (a) End time (b) Life time  
(c) Scope time (d) Visible time

[Ans. (b) Life time]

**3. The scope of a \_\_\_\_\_ is that part of the code where it is visible.**

- (a) Keyword (b) Variable  
(c) Function (d) Operator

[Ans. (b) Variable]

**4. A Function always first look up for a variable name in its \_\_\_\_\_ scope.**

- (a) Local (b) Enclosed  
(c) Global (d) Built-in

[Ans. (a) Local]

**5. The inner function can access the variable of the outer function. This is called \_\_\_\_\_ scope.**

- (a) Local (b) Function  
(c) Enclosed (d) Global

[Ans. (c) Enclosed]

**6. \_\_\_\_\_ can be separately compiled and stored in a library.**

- (a) Characteristics (b) Syntax  
(c) Modules (d) none of these

[Ans. (c) Modules]

**7. In Object Oriented Programming Language security is implanted through \_\_\_\_\_**

- (a) Access modifiers (b) Access modules  
(c) Access variables (d) Keywords

[Ans. (a) Access modifiers]

**8. \_\_\_\_\_ is a selective restriction of access to data in a program?**

- (a) Control variable  
(b) System authentication  
(c) Access control (d) Modules

[Ans. (c) Access control]

**9. \_\_\_\_\_ members of the class are accessible from outside the class.**

- (a) Private (b) Protected  
(c) Public (d) All of these

[Ans. (c) Public]

**CONSIDER THE FOLLOWING STATEMENT**

**1. Assertion :** The fundamental concept of access control is that minimizes risk to the object.

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

- (a) A & R is Fales  
(b) A is True but R is False  
(c) A is False but R is True  
(d) A & R is True [Ans. (d) A & R is True]

**CHOOSE THE CORRECT STATEMENT**

- 1.** (i) A Program cannot be divided into modules that work together to get the output.  
(ii) Modules can be separately compiled and stored in a library.  
(iii) Procedure, subroutines and functions are not examples of modules.  
(iv) Modules contain instructions, logic and data

- (a) i and ii (b) ii and iii  
(c) iii and iv (d) ii and iv

[Ans. (d) ii and iv]



### CHOOSE THE INCORRECT STATEMENT

1. (i) There a different types of variable scope  
(ii) Enclosed and extended are the type of variable scope  
(iii) A variable is declared outside of all the function is called global variable  
(iv) Built-in Scope is also called Module scope.  
(a) i, iii and iv (b) ii and iii  
(c) i and ii (d) iii only

[Ans. (c) i and ii]

### VERY SHORT ANSWERS

2 MARKS

#### 1. Define variable.

**Ans.** Variable are addresses (references, or pointers), to an object in memory.

#### 2. What is the use of LEGB rule?

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

#### 3. Name the types of variable scope.

- Ans.** (i) Local scope  
(ii) Enclosed scope  
(iii) Global scope  
(iv) Built-in scope.

#### 4. What is modular programming?

**Ans.** The process of subdividing a computer program into separate sub-programs is called modular programming.

### SHORT ANSWERS

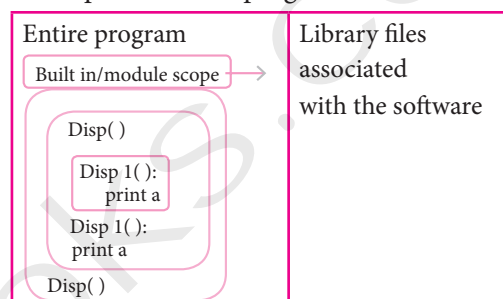
3 MARKS

#### 1. How the changes inside the function can't affect the variable on the outside of the function in unexpected ways?

- Ans.** (i) Every variable defined in a program has global scope.  
(ii) Once defined, every part of your program can access that variable. But it is a good practice to limit a variable's scope to a single definition.  
(iii) This way, changes inside the function can't affect the variable on the outside of the function in unexpected ways.

### 2. Write a note on built-in scope.

- Ans.** (i) Built-in scope is the widest scope. The built-in scope has all the names that are pre-loaded into the program scope when we start the compiler or interpreter.  
(ii) Any variable or module which is defined in the library functions of a programming language has Built-in or module scope. They are loaded as soon as the library files are imported to the program.



- (iii) Normally only Functions or modules come along with the software, as packages, therefore they will come under Built in scope.

### 3. Write a note on module.

- Ans.** (i) A module is a part of a program. Programs are composed of one or more independently developed modules. A single module can contain one or several statements closely related each other.  
(ii) Modules work perfectly on individual level and can be integrated with other modules. A software program can be divided into modules to ease the job of programming and debugging as well.  
(iii) A program can be divided into small functional modules that work together to get the output. The process of subdividing a computer program into separate sub-programs is called Modular programming.  
(iv) Modular programming enables programmers to divide up the work and debug pieces of the program independently. The examples of modules are procedures, subroutines, and functions.





**4. How will you ensure the principle of data encapsulation in object – oriented programming?**

**Ans.** Public members (generally methods declared in a class) are accessible from outside the class. The object of the same class is required to invoke a public method. This arrangement of private instance variables and public methods ensures the principle of data encapsulation.

**5. Write a note on access modifiers of a class.**

**Ans. (i)** Public members (generally methods declared in a class) are accessible from outside the class.

**(ii)** Protected members of a class are accessible from within the class and are also available to its sub-classes.

**(iii)** Private members of a class are denied access from outside the class. They can be handled only from within the class.

**6. Write a short note on types of variable scope.**

**Ans. (i)** Public members (generally methods declared in a class) are accessible from outside the class.

**(ii)** A variable which is declared outside of all the functions in a program is known as global variable.

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

**(iv)** Built-in scope the widest scope has all the names that are pre-loaded into program scope when we start the compiler or interpreter.

## LONG ANSWERS

**5 MARKS**

**1. Explain the concept access control.**

**Ans. (i)** Access control is a security technique that regulates who or what can view or use resources in a computing environment.

**(ii)** It is a fundamental concept in security that minimizes risk to the object.

**(iii)** In other words access control is a selective restriction of access to data. IN Object oriented programming languages it is implemented through access modifiers.

**(iv)** Classical object-oriented languages, such as C++ and Java, control the access to class members by public, private and protected keywords.

**(v)** Private members of a class are denied access from the outside the class. They can be handled only from within the class.

**(vi)** Public members (generally methods declared in a class) are accessible from outside the class. The object of the same class is required to invoke a public method. This arrangement of private instance variables and public methods ensures the principle of data encapsulation.

**(vii)** Protected members of a class are accessible from within the class and are also available to its sub-classes. No other process is permitted access to it. This enables specific resources of the parent class to be inherited by the child class.

**(viii)** Python doesn't have any mechanism that effectively restricts access to any instance variable or method. Python prescribes a convention of prefixing the name of the variable or method with single or double underscore to emulate the behaviour of protected and private access specifiers.

**(ix)** All members in a Python class are public by default, whereas by default in C++ and java they are private. Any member can be accessed from outside the class environment in Python which is not possible in C++ and java.

