# HIGHER SECONDARY SECOND YEAR ANSWER KEY
# COMPUTER SCIENCE

**T.THIRUMALAI, M.SC(CS).,B.ED.,**
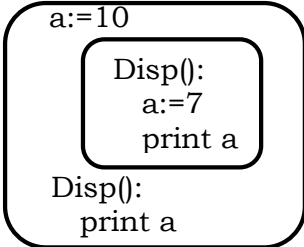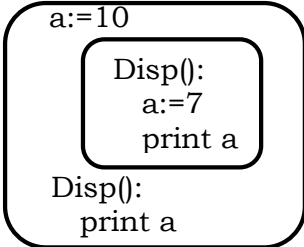**Cell: 9750827717, 7010154722**
**thirumalaibca.46@gmail.com**

## 1 MARKS

| Q. NO | ANSWER | MARKS |
|:---:|---|:---:|
| 1 | c) Definition | 1 |
| 2 | a) constructor | 1 |
| 3 | b) LEGB | 1 |
| 4 | c) Half-interval search | 1 |
| 5 | b) Ternary | 1 |
| 6 | a) for | 1 |
| 7 | d) return | 1 |
| 8 | a) Positive or negative numbers | 1 |
| 9 | a) eceicS retupmoC | 1 |
| 10 | d) . (dot) | 1 |
| 11 | b) Chen | 1 |
| 12 | b) SELECT | 1 |
| 13 | a) Modification | 1 |
| 14 | b) OS module | 1 |
| 15 | b) Distinct | 1 |

## 2 MARKS

| Q. NO | ANSWER | MARKS |
|:---:|---|:---:|
| 16 | **Pair:** <br> ➢ Pair is a compound structure which is made up of list or Tuple. <br> ➢ Any way of bundling two values together into one can be considered as a pair. <br> **Example:** lst := [10, 20] | 1 <br><br> 1 |
| 17 | The process of arranging the list items in ascending or descending order is called sorting. <br> **Example:** Bubble Sort, Selection Sort, Insertion Sort. | 1 <br><br> 1 |
| 18 | **Output:** <br> 5 10 15 | 2 |
| 19 | ➢ Slice is a substring of a main string. <br> ➢ A substring can be taken from the original string by using [ ] operator and index or subscript values. <br> ➢ Thus, [ ] is also known as slicing operator. <br> ➢ Using slice operator, you have to slice one or more substrings from a main string. <br> **General format of slice operation: str [start: end]** | 2 |
| 20 | Destructor is also a special method gets executed automatically when an object exit from the scope. <br> ♦ In python, _del_ () method is used as destructor. <br> ♦ It is just opposite to constructor. <br> **General format : def __del__(self):** <br>           **<statements>** | 2 |
| 21 | | |

| Command | Description | Component | |
|---|---|---|:---:|
| Insert | Insert data into table | DML | 1 |

| | | Create | To create a tables in the Database | DDL | 1 |
|---|---|---|---|---|---|
| **22** | **Syntax: cd: < absolute path>** <br> ✓ "cd" command used to change directory and absolute path refers to the complete path where python is installed. <br> **Example : "cd c:\program files\openoffice4\Program"** | | | | 1 <br><br> 1 |
| **23** | 🖥 Matplotlib can be installed using pip software. <br> 🖥 Pip is a management software for installing python packages. <br> **Importing Matplotlib using the command:** <br> import matplotlib.pyplot as plt <br> 🖥 Matplotlib can be imported in the workspace. | | | | 1 <br><br> 1 |
| **24** | **Output:** <br> **False** <br> **welcome** | | | | 1 <br> 1 |

# 3 MARKS

| Q. NO | ANSWER | MARKS |
|---|---|---|
| **25** | **Impure Function:** <br> 🖥 The variables used inside the function may cause side effects though the functions which are not passed with any arguments. <br> 🖥 In such cases the function is called impure function. <br> **Ex:** <br> **let randomnumber :=** <br>   **a := random()** <br>   **if a>10 then** <br>     **return: a** <br>   **else** <br>     **return: 10** | 2 <br><br><br><br><br> 1 |
| **26** | **Global Scope:** <br> ✳ A variable which is declared outside of all the functions in a program is known as global variable. <br> ✳ Global variable can be accessed inside or outside of all the functions in a program. <br> **Example:** <br><br> | 2 <br><br><br> 1 |

For Q26 example:

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

✳ 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.

| **27** | **Assignment Operator:** <br> ❀ In Python, = is a simple assignment operator to assign values to variable. | 3 |

| | | | |
|---|---|---|---|
| | ❀ There are various compound operators in Python like +=, -=, *=, /=, %=, **= and //= are also available. | | |

| 28 | ceil() | floor() | 3 |
|---|---|---|---|
| | Returns the smallest integer greater than or equal to x. | Returns the largest integer less than or equal to x. | |
| | **Ex:**print(math.ceil(23.6)) Output : 24 | **Ex:**print(math.floor(23.6)) Output:23 | |

| 29 | **Syntax:** remove() → List.remove(element) pop() → List.pop(index of an element) clear() → List.clear() | 1 1 1 |
|---|---|---|

| 30 | **CARTESIAN PRODUCT (Symbol: X)** ✓ Cross product is a way of combining two relations. The resulting relation contains, both relations being combined. ✓ A X B means A times B, where the relation A and B have different attributes. **Ex:** **A=3, B=2 means AXB=3X2=6.** | 2  1 |
|---|---|---|

| 31 | reader( ) method | DictReader class | 3 |
|---|---|---|---|
| | The reader method is designed to take each line of the file and make a list of all columns | DictReader works by reading the first line of the CSV and using each comma separated value in this line as a dictionary key. | |
| | Using this method one can read data from csv files of different format like quotes (' '),(pip ( │ ), and comma(,). | DictReader is a class of csv module is used to read a CSV file into a dictionary. | |
| | csv.reader works with list\tuple. | It creates an object which maps data to a dictionary. | |
| | **Syntax:** csv.reader(fileobject,delimiter ,fmtparams) | **Syntax:** csv.DictReader work with dictionary | |

| 32 | fetchone() | fetchmany() | 3 |
|---|---|---|---|
| | The fetch one () method returns the next row of a query result set or none in case there is no row left. | The fetchmany() method returns the next number of rows (n) of the result set. | |
| | Using while loop and fetchone() method we can display all the records from a table. | Displaying specified number of records is done by using fetchmany(). | |

| 33 | **Output:** [1,4,9,16,25] | T.THIRUMALAI, M.SC(CS).,B.ED., Cell: 9750827717, 7010154722 thirumalaibca.46@gmail.com | 3 |
|---|---|---|---|

# 5 MARKS

| Q. NO | ANSWER | MARKS |
|---|---|---|
| 34 A) | **Pure Function:** ➢ Pure functions are functions which will give exact result when the same arguments are passed. | |

- Pure function does not cause any side effects to its output.
- The return value of the pure functions solely depends on its arguments passed.
- They do not modify the arguments which are passed to them.
- If we call pure functions with same set of arguments, we will always get the same result.

**Example:**

let square x:=
    return: x*x

**Impure Function:**

- Impure functions never assure you that the function will behave the same every time it's called.
- Impure function causes side effects to its output.
- The return value of the impure functions does not solely depend on its arguments passed.
- They may modify the arguments which are passed.
- If we call impure functions with same set of arguments, we might get the different return values.

**Example:**

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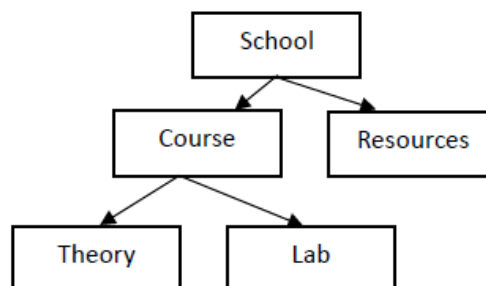      return: 10

**5**

**34 B)**

**Hierarchical Model:**

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



**2. Relational Model:**

- The Relational Database model was first proposed by E.F.Codd in 1970.
- Nowadays, it is the most widespread data model used for database applications around the world.
- The basic structure of data in relational model is tables (relations).

**5**

- All the information's related to a particular type is stored in rows of that table.
- A relation key is an attribute which uniquely identifies a particular tuple.

| Stu_id | Name | Age |
|---|---|---|
| 1 | Aruna | 17 |
| 2 | Prabhakar | 16 |
| 3 | Aakash | 15 |

| Sub_id | Subject | Teacher |
|---|---|---|
| 1 | Tamil | Jegan |
| 2 | Maths | Veni |
| 3 | Science | Jeya |

| Stu_id | Sub_id | Marks |
|---|---|---|
| 1 | 1 | 98 |
| 2 | 2 | 87 |
| 3 | 2 | 75 |

### 3. Network Model:

- Network database model is an extended form of hierarchical data model.
- The difference between hierarchical and Network data model is: In hierarchical model, a child record has only one parent node.
- In a Network model, a child may have many parent nodes. It represents the data in many-to-many relationships.
- This model is easier and faster to access the data.
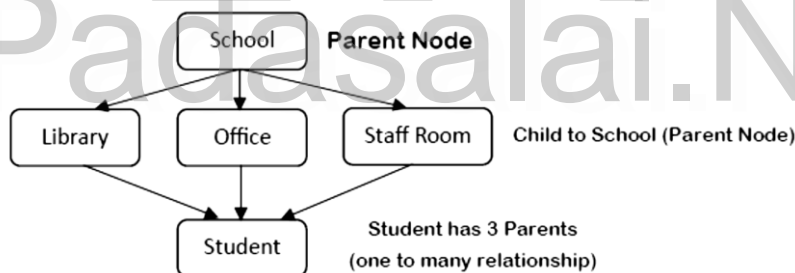


### 4. Entity Relationship Model (ER model):

- In this database model, relationships are created by dividing the object into entity and its characteristics into attributes.
- It was developed by Chen in 1976.
- This model is useful in developing a conceptual design for the database.
- It is very simple and easy to design logical view of data.
- The different shapes used in ER diagram is
- Rectangle represents the entities.
  Example: Doctor and Patient
- Ellipse represents the attributes.
  Example: D-id, D-Name, P-id and P-Name
- Diamond represents the relationships.
  Example: Doctor diagnosis the Patient.
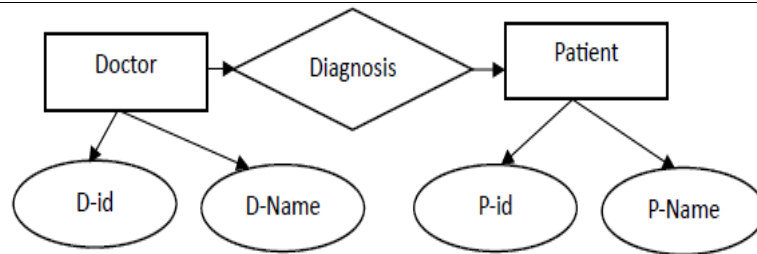
**5. Object Model:**
  ➤ Object model stores the data in the form of objects, attributes and methods, classes and Inheritance.
  ➤ This model handles more complex applications, such as Geographic information System (GIS), scientific experiments, engineering design and manufacturing.
  ➤ It is used in file Management System.
  ➤ It represents real world objects, attributes and behaviours.

**Example:**
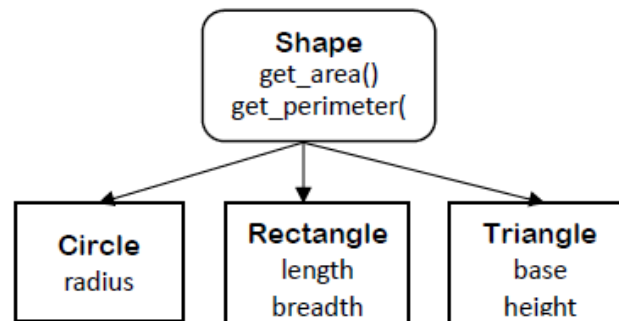


where,
  ╬ Shape is the **class**.
  ╬ Circle, Rectangle and Triangle are **objects** of class Shape.
  ╬ radius, length, breadth, base and height are **attributes**.
  ╬ get_area() and get_perimeter() are the **methods**.

| | | |
|---|---|---|
| **35 A)** | **Operators in Python:**<br>In computer programming languages operators are special symbols which represent computations, conditional matching etc.<br>  i.    The value of an operator used is called **operands**.<br>  ii.   Operators are categorized as Arithmetic, Relational, Logical, Assignment, Conditional etc.<br>  iii.  Value and variables when used with operator are known as **operands**.<br><br>**Arithmetic operators:**<br>  1. An arithmetic operator is a mathematical operator that takes two operands and performs a calculation on them.<br>  2. They are used for simple arithmetic.<br>**Relational or Comparative operators:**<br>  1. A Relational operator is also called as Comparative operator which checks the relationship between two operands.<br>  2. If the relation is true, it returns True; otherwise it returns False.<br>**Logical operators :** | **5** |

a. In python, Logical operators are used to perform logical operations on the given relational expressions.

b. There are three logical operators they are and, or and not.

**Assignment operators:**

a. In Python, = is a simple assignment operator to assign values to variable.

b. There are various compound operators in Python like **+=, -=, *=, /=,** %=, **\*\*=** and **//=** are also available.

**Conditional operator or Ternary operator :**

❀ Ternary operator is also known as conditional operator that evaluate something based on a condition being true or false.

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

---

| 35 B) | **Constraint:** |  |
|---|---|---|

**Constraint:**

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

**Types of Constraint:**

1. Unique Constraint 2.Primary Key Constraint 3.Default Constraint 4. Check Constraint 5. Table constraint

**1.Unique Constraint:**

▪ This constraint ensures that no two rows have the same value in the specified columns.

▪ For example **UNIQUE** constraint applied on Admno of student table ensures that no two students have the same admission number and the constraint can be used as:

CREATE TABLE Student (

Admno integer NOT NULL UNIQUE, → Unique constraint

Name char (20) NOT NULL, Gender char (1),Age integer,

Place char (10));

▪ The **UNIQUE** constraint can be applied only to fields that have also been declared as **NOT NULL**.

▪ When two constraints are applied on a single field, it is known as multiple constraints.

▪ In the above Multiple constraints **NOT NULL** and **UNIQUE** are applied on a single field Admin no.

**2.Primary Key Constraint:**

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

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

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

**Example:**

CREATE TABLE Student (

Admno integer NOT NULL PRIMARY KEY, → **Primary Key constraint** Name char(20)NOT NULL, Gender char(1), Age integer, Place char(10));

**3.DEFAULT Constraint :**

➢ The **DEFAULT** constraint is used to assign a default value for the field.

**5**

> When no value is given for the specified field having **DEFAULT** constraint, automatically the default value will be assigned to the field.

**Example:**

CREATE TABLE Student (

Admno integer NOT NULL PRIMARY KEY, Name char(20)NOT NULL, Gender char(1), Age integer DEFAULT = "17", → Default Constraint Place char(10) );

> In the above example the "Age" field is assigned a default value of 17, therefore when no value is entered in age by the user, it automatically assigns 17 to Age.

### 4. Check Constraint :

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

**Example:**

CREATE TABLE Student (

Admno integer NOT NULL PRIMARY KEY Name char(20)NOT NULL, Gender char(1), Age integer (CHECK<=19), → Check Constraint Place char(10));

> In the above example the check constraint is set to Age field where the value of age must be less than or equal to 19.

### 5.TABLE CONSTRAINT :

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

CREATE TABLE Student 1 (

Admno integer NOT NULL, Firstname char(20), Lastname char(20), Gender char(1), Age integer, Place char(10),

PRIMARY KEY (Firstname, Lastname) → Table constraint );

- In the above example, the two fields, First name and Last name are defined as Primary key which is a Table constraint.

---

**36 A)**

**Syntax:**

while <condition>:

    statements block 1

[else:

    statements block2]

✓ In the **while** loop, the condition is any valid Boolean expression returning True or False.
✓ The **else** part of while is optional part of **while**.
✓ The **statements block1** is kept executed till the condition is true.
✓ If the **else** part is written, it is executed when the condition is tested False.

**5**

| | | |
|---|---|---|
| | ✓ Recall **while** loop belongs to entry check loop type that is it is not executed even once if the condition is tested false in the beginning.<br><br>**Example:**<br>**Program to print numbers from 10 to 15 using while loop.**<br>i=10<br>while (i<=15):<br>    print (i,end='\t'<br>     i=i+1<br>**Output:** 10 11 12 13 14 15 | |
| 36 B) | **Python File Modes:**<br>1) **'r'** → Open a file for reading (default).<br>2) **'w'** → Open a file for writing. Creates new file if it does not exist or truncates the file if it exists.<br>3) **'x'** → Open a file for exclusive creation. If the file already exists, the operation fails.<br>4) **'t'** → Open in text mode (default).<br>5) **'a'** → Open for appending at the end of the file without truncating it. Creates a new file if it does not exist.<br>6) **'b'** → Open in binary mode.<br>7) **'+'** → Open a file for updating (reading and writing). | 5 |
| 37 A) | **Recursion Function:**<br>• When a function calls itself is known as recursion.<br>• Recursion works like loop but sometimes it makes more sense to use recursion than loop.<br>• You can convert any loop to recursion. A recursive function calls itself.<br>• Imagine a process would iterate indefinitely if not stopped by some condition! Such a process is known as infinite iteration.<br>• The condition that is applied in any recursive function is known as base condition.<br>• A base condition is must in every recursive function otherwise it will continue to execute like an infinite loop.<br><br>**Overview of how recursive function works:**<br>❖ Recursive function is called by some external code.<br>❖ If the base condition is met then the program gives meaningful output and exits.<br>❖ Otherwise, function does some required processing and then calls itself to continue recursion.<br>❖ Here is an example of recursive function used to calculate factorial.<br>**Example:**            **Output:**<br>def fact(n):                    1<br>    if n == 0:                  120<br>        return 1<br>    else:<br>        return n * fact (n-1)<br>    print (fact (0)) | 5 |
| 37 B) | **Python's sys module:** | |

❈ This module provides access to some variables used by the interpreter and to functions that interact strongly with the interpreter.

**sys. argv:**
- sys.argv is the list of command-line arguments passed to the Python program.
- **argv contains** all the items that come via the command-line input, it's basically a list holding the command line arguments of the program.
- To use **sys.argv**, **import sys** should be used. The first argument, sys. argv [0] contains the name of the python program
- (example pali.py) and sys. argv [1] is the next argument passed to the program (here it is the C++ file).

**Python's OS Module :**
- The OS module in Python provides a way of using operating system dependent functionality.
- The functions that the OS module allows you to interface with the Windows operating system where Python is running on.
- **os.system():** Execute the C++ compiling command (a string contains Unix, C command which also supports C++ command) in the shell (Here it is Command Window).
- For Example to compile C++ program **g++ compiler** should be invoked.

**Command**:
**os.system('g++' + <variable_name1> + '-<mode>' + <variable_name2>)**

where each argument contains,

**os.system:** function system() defined in **os** module to interact with the operating system
**g++:** General compiler to compile C++ program under windows operating system.
**variable_name1:** Name of the C++ file along with its path and without extension .cpp in string format
**mode:** To specify input or output mode. Here it is o prefixed with Hyphen.
**variable_name2:** Name of the executable file without extension .exe in string format

**Python getopt module :**
- ➤ The getopt module of Python helps you to parse (split) command-line options and arguments.
- ➤ This module provides two functions to enable command-line argument parsing.
- ➤ This method parses command-line options and parameter list.

**Syntax:**

**<opts>,<args>=getopt.getopt(argv, options, [long_options])**

**5**

**Here is the details of the parameters:**
**1.argv :**
→ This is the argument list of values to be parsed (splited). In our program the complete command will be passed as a list.
**2.options :**
→ This is string of option letters that the Python program recognize as, for input or for output, with options (like 'i' or 'o') that followed by a colon (:). Here colon is used to denote the mode.
**3.long_options :**
→ This parameter is passed with a list of strings. Argument of Long options should be followed by an equal sign ('=').
**getopt() method returns value consisting of two elements.** Each of these values are stored separately in two different list (arrays) **opts** and **args**. **args** will be an empty array if there is no error in splitting strings by getopt().

For example:
**opts, args = getopt.getopt (argv, "i:",['ifile='])**
where **opts** contains      [('-i', 'c:\\pyprg\\p4')]
**-i :-** option nothing but mode should be followed by : (colon)
'c:\\pyprg\\p4' value nothing but the absolute path of C++ file.

→ In our examples since the entire command line commands are parsed and no leftover argument, the second argument args will be empty [ ].
→ If args is displayed using print () command it displays the output as [ ].

>>>print(args)
[ ]

| 38 A) | {1,2,3,4,5,6,7,8}<br>{4,5}<br>{1,2,3}<br>{6,7,8}<br>{1,2,3,6,7,8}          T.THIRUMALAI, M.SC(CS).,B.ED.,<br>Cell: 9750827717, 7010154722<br>thirumalaibca.46@gmail.com |
|---|---|
| 38 B) | **Aggregate functions of SQL.**<br>❀ These functions are used to do operations from the values of the column and a single value is returned.<br>1. COUNT()      2. AVG()      3. SUM()<br>4. MAX()      5. MIN()<br><br>**1. COUNT()**<br>❋ The SQL COUNT() function returns the number of rows in a table satisfying the criteria specified in the WHERE clause.<br>❋ COUNT() returns 0 if there were no matching rows.<br><br>**Example:**<br>import sqlite3<br>connection = sqlite3.connect("Academy.db")<br>cursor = connection.cursor() |

```
cursor.execute("SELECT COUNT(*) FROM student ")
result = cursor.fetchall()
print(result)
```
**Output:** [(7,)]

## 2. AVG()
※ The following SQL statement in the python program finds the average mark of all students.

**Example:**
```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT AVG(AVERAGE) FROM student ")
result = cursor.fetchall()
print(result)
```
**Output**
[(84.65714285714286,)]

## 3. SUM():
※ The following SQL statement in the python program finds the sum of all average in the Average field of "Student table".

**Example:**
```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT SUM(AVERAGE) FROM student ")
result = cursor.fetchall()
print(result)
```
**Output:** [(592.6,)]

## 4.MAX()
❖ The MAX() function returns the largest value of the selected column.

## 5.MIN()
❖ The MIN() function returns the smallest value of the selected column.

**Prepared By**

T.THIRUMALAI, M.SC(CS).,B.ED.,
Cell: 9750827717, 7010154722
thirumalaibca.46@gmail.com