**Python training:**

**Course Name:** **Python**

**Start Date: 20 Feb 2023**

**End Date: 19 April 2023**

**------------------------------------------------------------------------------------------------------------**

**Expectation:**

**Attend in class training and then do self-learning/Hands on practice**

**To-Do list:**

**Self-study:**

1. **Go Through the E-BOX:**

Complete the relevant modules.

**Software:**

1. Try below installation-

On Windows Platform - Python 3.x on windows Install Python 3.x on windows using below link: <https://www.python.org/downloads/>

**OR**

Anaconda installation on Windows

Use the link: <https://www.anaconda.com/products/individual>

**OR**

1. Online Python interpreter:

<https://www.onlinegdb.com/online_python_interpreter>

or

<https://www.online-python.com/>

**Documentation:** Refer Python Manual doc as shown in below screen shot-

Graphical user interface, application, Teams

Description automatically generated

**Database:**

Please have access to MySQL 5.5

<https://dev.mysql.com/downloads/windows/installer/8.0.html>

**Apache Server:**

**Please install Apache server from :** [**http://archive.apache.org/dist/httpd/binaries/win32/httpd-2.2.16-win32-x86-no\_ssl.msi**](http://archive.apache.org/dist/httpd/binaries/win32/httpd-2.2.16-win32-x86-no_ssl.msi)

**\*\*\*\*Anaconda software:**

Software installation from:

<https://www.anaconda.com/download/>

Please execute it.

Start menu-Anaconda3->prompt>python Hello.py

**\*\*\*\*Eclipse Settings**

1. **To add python development plugin**

**windows🡪preferences**

**or help->install new software**

**click on ADD**

**window : Name : enter as Python 2**

**search pydev download in google**

**put this location** [**http://pydev.org/updates**](http://pydev.org/updates)

[http://update-production-pydev.s3.amazonaws.com/pydev/updates](http://update-production-pydev.s3.amazonaws.com/pydev/updates/site.xml)

**More Info at:** **http://www.pydev.org/download.html**

**2)**

**then preferences , windows option-preferences**

**interpreter**

**selete c:\python\python.exe**

\*\*\*PyCharm Execution

\*\*\*Spyder Execution

**Topics Covered :**

**Day1 :20th Feb 2023**

|  |
| --- |
| Python Introduction, Features |
| Setup the Python Programming Environment, Python Installation |
| Variables |
| Comments |
| Dynamic Vs Static Typing |
| Built-in types : int, float, str |
| Numbers and Math operators |
| Conversions |
| Comparison and other operators |
| Built-in functions: print, type, input, int, float, ord, hex, oct… etc. |

**Day2 :21st Feb 2023 (Tue)**

Strings in Python  
 User Inputs  
 String Indexing and Operations  
 String Slicing  
 Formatting Strings  
 String Methods

for loop

range()

list conversion : list()

print () function default parameters :sep =” ” , end =”\n”, file =sys.stdout, Flush = False

**Day3 :22nd Feb 2023 (Wed)**

Lists in Python

Intro to Lists

List Operations

List methods

Tuples in Python

Intro to Tuples

Tuple Operations

Tuple methods

Tuples Vs. Lists(Immutable, Mutable)

for loop on List, Tuple, for loop with range() diff

**Day4 :1st Mar 2023 (Wed)**

List sort() key, reverse parameters

pass keyword

List Comprehension

String split () method

sorted(), reversed() function

**Random/Unordered :** Set, Dictionary – Mutable

**NO N**umerical Indexing concept.

For loop on set, dict

Methods of set –

Methods of Dictionary

TyepErroe, IndexError, KeyError, AttributeError etc….

**Day5 :2nd Mar 2023 (Thu)**

Set, Dict, methods…contd…

Dictionary Comprehension

Complex Data structures: Dict with inner list, Dict with inner Dict

**Day6,7 :Mar 7 and 8 2023 (Tue, Wed)**

**Functions**

Built in functions: print() sum() len() id() type() int() float() hex() oct() sorted() reversed() list() tuple() set() dict() ……

**def display( ):**

**pass**

**display()**

parameters

Return statement.

Functions Arguments: Positional, Default, Keyword, \*args and \*\*kwargs

variable number of arguments list - \*args and \*\*kwargs

lambda – anonymous function

scope of the var : global

**File Handling**

Opening and Reading Files

Reading Files: Tell, Seek and Cursors

with Keyword

Reading Files into a List

Writing to Text Files

Appending to Text Files

File Processing

**Day8:Mar 9 2023 (Thu)**

Modules

Exception Handling

**Day9, 10:Mar 14,15 2023 (Tue, Wed)**

OOP

**Day11:Mar 20 2023 (Mon)**

OOP .. *continued….*

Regex

**Day12:Mar 21 2023 (Tue)**

Regex.. *continued….*

*Functional programming, map() filter(), list.sort(), sorted() with complex sorting logic*

**Day13:Mar 28 2023 (Tue)**

Various standard libraries- os, sys, copy, math, glob, shutil, pickle , Random, datetime, timeit, doctest, unittest, cProfile, stats, XML parsing

**Day14:Mar 30 2023 (Thu)**

Various standard libraries- doctest, unittest, cProfile, stats, XML parsing

**Day15:Apr3 2023 (Mon)**

Working with CSV files

Intro to CSV

Reading CSV Files

Writing CSV Files

Working with Excel Files in Python

Setup the Environment. Installing OpenPyXL

Reading Excel Files

Writing Excel Files

Creating New Excel Files

Web technology- CGI introduction, Executing Server side Python script/web page

Anaconda Soft

**Day16:Apr4 2023 (Tue)**

Databases connectivity program

**Day16:Apr4 2023 (Tue)**

**Numpy lib**

**Anaconda software : Jupyter Notebook\**

These are the notebook file with an extension.ipynb

To execute these , use->

Anaconda3-🡪 jupyter notebook

**Execution->**

1)start menu->hit this command

OR

2)base prompt of Anacond>Demos>jupyter Notebook--🡪

It hits the web page :

It open the home folder of the current folder, Demos>

IDE execution

IDE – PyCharm

Eclipse

Spyder

**Day17:Apr6 2023 (Thu)**

Anaconda execution, Numpy

**Day18:Apr10 2023 (Mon)**

Anaconda execution, Numpy, Pandas

**Day19:Apr13 2023 (Thu)**

Matplotlib, seaborn libraries

Revision

**\*\*\*\*\*Assignments**

**Day1: 20 Feb (Monday)**

**Execute all the shared sample codes in “Day1\_20Feb” folder first.**

1. Accept 2 numbers from user and print addition, subtraction.
2. Accept a number, print the square of it.
3. Accept a password, if it matches with “PYthonic@#20” then print the message as “Correct Password, Welcome!!”

Else “Incorrect Password, Sorry! Access Denied!”

**Day2: 21 Feb (Tuesday)**

**Execute all the shared sample codes in “Day2\_21Feb” folder first.**

1. Accept a string, check whether it is palindrome

Eg. “madam”

1. Display three string “Name”, “Is”, “James” as “Name\*\*Is\*\*James”

*(Hint: Just 1 line code)*

1. Convert Decimal number to octal using print() output formatting

*(Hint: Just 1 line code)*

4. Write a program to print the following number pattern using a loop.

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

### 5.Calculate the sum of all numbers from 1 to a given number.

### 6. Write a program to display below Star (asterisk) patterns in Python.

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

**Day3: 22 Feb (Wednesday)**

**Execute all the shared sample codes in “Day3\_22Feb” folder first.**

1. Let’s say I give you a list saved in a variable: a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100].

Write Python code that takes this **list a** and makes a new **list b** that has only the even elements of this list in it.

1. List sorting

sort the list of following names by –

1. an ascending order of words
2. descending order of words
3. an ascending order of number of letters in each name

unsortedList = ['Aaaa', 'bb', 'cccccccc', 'zzzzzzzzzzzz']

3.You have given a Python list. Write a program to find value 20 in the list, and if it is present, replace it with 200. Only update the first occurrence of an item.

Given: list1 = [5, 10, 15, 20, 25, 50, 20]

Expected output:

[5, 10, 15, 200, 25, 50, 20]

4.Accept scores of a student for 5 topics from keyboard and then print the total score and % score.

**Day4: 1 mar (Wednesday)**

**Execute all the shared sample codes in “Day4\_1 Mar” folder first.**

1. Let’s say I give you a list saved in a variable: a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100].

Write one line of Python that takes this **list a** and makes a new **list b** that has only the even elements of this list in it.

1. Write a program to find the largest number among the three input numbers.
2. Accept the content from keyboard for empdata. Store it in a dictionary -“empdata”

Print all emp details on separate lines.

Print the total sal.

*Sample keyboard Input-*

*1a:25000*

*2a:30000*

*3a:60000*

**Day5: 2 mar (Thursday)**

**Execute all the shared sample codes in “Day5\_2 Mar” folder first.**

1. Accept the content from keyboard for empdata. Store it in a dictionary -“empdata”

Print all emp details on separate lines.

Print the total sal.

*1a:ABC:25:25000*

*2a:XYZ:30:30000*

*3a:LMN:45:60000*

Hint: store the data in dictionary and then process, 1a, 2a, 3a are the keys and rest all is the value section data respectively.

Sample O/P

Emp ID : 1a Details:

Name: ABC

Age : 25

Sal : 25000

Emp ID : 2a Details:

Name: XYZ

Age : 30

Sal : 30000

Emp ID : 1a Details:

Name: LMN

Age : 45

Sal : 60000

Total Salary = 115000

**Day6,7: Mar 7, 8 (Tue and Wed)**

**Execute all the shared sample codes in “Day6,7\_Mar7,8” folder first.**

1. Please enter a number a number and determine whether the number is prime or not. Write appropriate functions for the same.
2. Write a program to display all the prime numbers within an interval. Write a function for the same.
3. Write a program to display the powers of2 using anonymous function.
4. Write a Python Program to add two matrices using nested loop.

Consider below 2 sample matrices.

X = [[12,7,3],

[4 ,5,6],

[7 ,8,9]]

Y = [[5,8,1],

[6,7,3],

[4,5,9]]

1. Write a Python program (function) to count the number of strings where the string length is 2 or more and the first and last character are same from a given list of strings.

Consider the sample strings in a list as- (['abc', 'xyz', 'aba', '1221']

1. Write a Python program to get the maximum and minimum value in a dictionary.

Consider sample Input dictionary as my\_dict={'x':500,'y':5874,'z':560}

Sample output expected - Sample Output:

Maximum Value: 5874

Minimum Value: 500

Hint: use max(), lambda

7.Please enter a number a number and determine whether the number is prime or not. Write appropriate functions for the same.

14.Write a function to print the addition of 2 numbers passed to it as parameters.

8.Read file content of given file “empdata”, and print total sal.

*File data:*

*1a:ABC:25:25000*

*2a:XYZ:30:30000*

*3a:LMN:45:60000*

Hint: store the data in dictionary and then process

**Day8: Mar 9(Thu)**

**Execute all the shared sample codes in “Day8\_Mar9” folder first.**

1. Count of languages

Store the Country data from country file only for Language and its count in a dictionary.

Display the o/p in below 2 formats –

***>>>***

***Portuguese 1***

***Franch 4***

***Chinese 1***

***Vietnamese 1***

***German 1***

***English 5***

***Japanese 1***

***Greek 1***

***Indian 1***

***Spanish 3***

***Arabic 2***

***Hungerian 1***

***Italian 1***

***--------------------------------------------------***

***{'Portuguese': 1, 'Franch': 4, 'Chinese': 1, 'Vietnamese': 1, 'German': 1, 'English': 5, 'Japanese': 1, 'Greek': 1, 'Indian': 1, 'Spanish': 3, 'Arabic': 2, 'Hungerian': 1, 'Italian': 1}***

1. Language and Country

Store the Country data only for Language and its list of countries in a dictionary.

Display the o/p as shown below -

>>>

{'Portuguese': ['Brazil'], 'Franch': ['Cameroon', 'Djibouti', 'Equatorial Guinea', 'France'], 'Chinese': ['China'], 'Vietnamese': ['Vietnam'], 'German': ['Germany'], 'English': ['United Kingdom', 'United States', 'Fiji', 'Canada', 'Ireland'], 'Japanese': ['Japan'], 'Greek': ['Greece'], 'Indian': ['India'], 'Spanish': ['Venezuela', 'Argentina', 'Honduras'], 'Arabic': ['Yemen', 'Bahrain'], 'Hungerian': ['Hungary'], 'Italian': ['Italy']}

1. Place implementation of above assignment 1 and 2 in a function inside a module file and then write final application with import to that module file and call the relevant functionalities from it.

**Day9,10: Mar 14,15(Wed, Thu)**

**Execute all the shared sample codes in “Day9,10\_Mar14,15” folder first.**

1. Create Employee class.

Maintain class level variable “empCount” and write function “displayCount()” to

display the total empCount.

Define instance variables Name and salaray.

Also write instance method “displayEmployee() to display all employee details (Name and Salary)

Override \_\_str\_\_() and \_\_repr\_\_() methods to display Employee details (Name and

Salary), instead of default string representation.

1. Create BankAccount class

Specify initial amount in Bank account as 50000

Write functionality for showBalance()

Deposit(amount)

WithDrawl(amount)

Error to be handled if Withdraw amount is greater that available balance.

Appropriate Exception handling expected.

**Day11: Mar 20(Mon)**

**Execute all the shared sample codes in “Day11\_Mar20” folder first.**

1. Create a text file “emails.txt” and store a big list of valid and invalid email addresses on separate lines. Write a program to match the set of all valid e-mail addresses.
2. Create a text file “urls.txt” and store a big list of URL’s. Write a program to match simple Web domain names that begin with "www." and end with a ".com" suffix, e.g., http://www.yahoo.com.

**Day12: Mar 21(Tue)**

**Execute all the shared sample codes in “Day12\_Mar21” folder first.**

1. Let’s say I give you a list saved in a variable: a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100].

Write one line of Python that takes this list a and makes a new list b

that has only the even elements of this list in it. (Use filter and lambda)

1. Given a string

sentence = 'It is raining cats and dogs'

get 1 target list with length of each word in this sentence

Hint : Use map, lambda, split appropriately

**Day13: Mar 28(Tue)**

**Execute all the shared sample codes in “Day13\_Mar28” folder first.**

1. Read given XML file movies.xml. Print the total count of movie details stored in it. Also display all Movie details

No Additional assignment for Day 14(30 Mar) and Day 15(3 Apr)

**Day15: April 3(Mon)**

**Execute all the shared sample codes in “Day15\_April3” folder first.**

**Day16: April 4(Tue)**

**Execute all the shared sample codes in “Day16\_Apr4” folder first.**

1. Write Python script for database connectivity using sqlite3 module.

Create a table from the script, please drop it if it already exists.

CREATE TABLE users(id INTEGER PRIMARY KEY, name TEXT,

phone TEXT, email TEXT unique, password TEXT)

Accept some values for varaibles-

name1 ,phone1 ,email1, password1

e.g.

name1 = 'Ravi Verma'

phone1 = '9923849335'

email1 = 'ravi\_verma@gmail.com'

password1 = 'tough@password!!'

Insert these values in table users.

Read the data back and display

Accept name from user, display the details for that user if it exists in table users.

Write different methods for all these functionalities.

1. Write Python script for database connectivity using mysql.connector module.

Create a table from the script, please drop it if it already exists.

CREATE TABLE users(id INTEGER PRIMARY KEY, name TEXT,

phone TEXT, email TEXT unique, password TEXT)

Accept some values for varaibles-

name1 ,phone1 ,email1, password1

e.g.

name1 = 'Ravi Verma'

phone1 = '9923849335'

email1 = 'ravi\_verma@gmail.com'

password1 = 'tough@password!!'

Insert these values in table users.

Read the data back and display

Accept name from user, display the details for that user if it exists in table users.

Write different methods for all these functionalities.

**Day17, 18: April 6, 10(Thu, Mon)**

**Execute all the shared sample codes in “Day17\_Apr6” and Day18\_Apr10 folder.**

**Execute shared Jupyter Notebook**

**Day19: April 13(Thu)**

**Execute all the shared sample codes in “Day19\_Apr13” folder.**

**Execute shared Jupyter Notebook and solve the assignment**

**Day20: April 19(Wed) and Day21 : 18 May – Revision Connect session**

**Execute all the given assignments:**

**Q1. Merge and Sort List**

The list of 'n' numbers is given to the student of 3rd class by the Maths teacher. And another 'n' numbers are given to another student.  Now they both have to merge the given numbers which were given by the Maths teacher. After merging the two set of list elements, then they have to write those merge elements in the sorted order.  
Help the students to write a program.  
So, in this program enter the two lists of length 3, and print them in the sorted order.  
  
**Input Format:**  
The first line of input consists of an integer which corresponds to a size of both the lists.  
The second line of input consists of an 'n' integer values which correspond to elements of the list1.  
  
**Output Format:**  
The first line of output consists of merge list which is having elements from both the lists.  
The second line of output consists of sorting order of merge list.  
Refer the Sample Input Output.  
  
**Note:**  
[All the bold text corresponds to input and rest corresponds to output]  
  
**Sample Input and Output:**  
Enter the length of the list:  
**3**  
Enter the elements forfirst list:  
**57 2 12**  
Enter the elements forsecond list:  
**11 14 1**  
Merging of two lists:  
[57, 2, 12, 11, 14, 1]  
Sorted list:  
[1, 2, 11, 12, 14, 57]

2.

PROBLEM

**INHERITANCE - V - ALLDETAILS**

**Inheritance - V - AllDetails**

Write a python program to get the user , product , and order details and display the user details and their product details by using the customer name.

Create a class ' **User**' with argumented constructor with following attributes

|  |  |
| --- | --- |
| **DataType** | **Attribute** |
| String | FirstName |
| String | LastName |
| String | DateOfBirth |
| String | Age |

This class '**RentalOrderDetails**' which inhero contains the following methods

|  |  |
| --- | --- |
| **method name** | **Description** |
| def displayUserDetails(self) | This method is used to print the User details. |

Create a class '**Address**' with argumented constructor , contains the following attributes

|  |  |
| --- | --- |
| **DataType** | **Attribute** |
| String | Street |
| String | City |
| String | PinCode |

This class '**RentalOrderDetails**' which inhero contains the following methods

|  |  |
| --- | --- |
| **method name** | **Description** |
| def displayAddressDetails(self) | This method is used to print the Address of the user. |

Create a class '**Product**' with following attributes attributes

|  |  |
| --- | --- |
| **DataType** | **Attribute** |
| String | ProductName |
| String | ProductBrand |
| String | ProductType |
| String | ProductColor |

This class '**RentalOrderDetails**' which inhero contains the following methods

|  |  |
| --- | --- |
| **method name** | **Description** |
| def displayProductDetails(self) | This method is used to print the Product details. |

Create a class '**RentalOrderDetails**' with argumented constructor , this class inherites the class **'User**' , '**Address**' ,  and '**Product**'  and contains the following attributes

|  |  |
| --- | --- |
| **DataType** | **Attribute** |
| Object | User Object |
| Object | Address Object |
| Object | Product Object |
| String | CustomerName |
| String | OrderedDate |
| String | DeliveryDate |

This class '**RentalOrderDetails**' which inhero contains the following methods

|  |  |
| --- | --- |
| **method name** | **Description** |
| def displayOrderDetails(self , list , customer\_name) | This method takes list of RentalOrderDetails objects and customer name and displays the Order details of the appropriate customer and Prints '**No Details found for the given Customer name**' if the customer name is not in the list. |

**[All Texts in bold corresponds to the input and rest are output]**

**Sample Input :**

Enter the number OrderDetails to be added  
**1**  
Enter the details of Order 1  
Enter the first name  
**deva**  
Enter the last name  
**muthu**  
Enter the date of birth(dd-mm-yyyy)  
**11-11-1998**  
Enter the age  
**21**  
Enter the street  
**rk**  
Enter the city  
**cbe**  
Enter the Pincode  
**441526**  
Enter the Product Name  
**mouse**  
Enter the Product Brand  
**lg**  
Enter the Product Type  
**mouse**  
Enter the Product Color  
**black**  
Enter the Customer Name  
**kumar**  
Enter the Ordered Date  
**16-03-2019**  
Enter the Delivery Date  
**18-03-2019**  
Enter the Customer name whose details need to be displayed  
**kumar**

**Sample Output :**

Order Details  
Customer Name : deva  
Ordered Date : muthu  
Delivery Date : 21  
Product Details  
Product Name : mouse  
Product Brand : lg  
Product Type : mouse  
Product Color : black  
User Details  
First Name : deva  
Last Name : muthu  
DOB : 11-11-1998  
Age : 21  
Addres of the Order  
Street : rk  
City : cbe  
PIN code : 441526

Top of Form

Bottom of Form

Code files:

Main.py

class User:

# fill your code

def displayUserDetails(self):

# fill your code

class Address:

# fill your code

def displayAddressDetails(self):

# fill your code

class Product:

# fill your code

def displayProductDetails(self):

# fill your code

class RentalOrderDetails(User , Address , Product):

# fill your code

def displayOrderDetails(self , li , cus\_name):

# fill your code

# Main

n = int(input("Enter the number OrderDetails to be added\n"))

c = 1

for i in range(n):

print("Enter the details of Order %d"%c)

fname = input("Enter the first name\n")

lname = input("Enter the last name\n")

dob = input("Enter the date of birth(dd-mm-yyyy)\n")

age = input("Enter the age\n")

street = input("Enter the street\n")

city = input("Enter the city\n")

pin = input("Enter the Pincode\n")

pname = input("Enter the Product Name\n")

pbrand = input("Enter the Product Brand\n")

ptype = input("Enter the Product Type\n")

pcolor = input("Enter the Product Color\n")

cname = input("Enter the Customer Name\n")

odate = input("Enter the Ordered Date\n")

ddate = input("Enter the Delivery Date\n")

# fill your code

**2.OOPS(ABSTRACT)**

**Event(Abstract)**

Pratap, the event organizer was selected to organize an event on the occasion of the Dussehra festival. Pratap was very happy that he got such a big opportunity and was very excited to conduct it. But he had the idea of the consequences he would face during the event. So he notes down all the points that require everyday calculation. And according to that, he starts writing a code to run his event smoothly. Being a student of Computer and Science Engineering, he starts writing code. Let's help Pratap to complete his code. Write a program as per the requirement of Pratap.   
  
Create a class **Event** as an abstract class with the following methods in it.

|  |  |  |
| --- | --- | --- |
| **SL no** | **Method** | **Description** |
| 1 | cal\_total\_spent() | use this method in child classes |
| 2 | cal\_total\_revenue() | use this method in child classes |
| 3 | profit\_or\_loss() | use this method in child classes |

Create the child class**Exhibition** with the following attributes.

|  |  |
| --- | --- |
| **variable** | **Datatype** |
| no\_of\_stalls | int |
| cost\_per\_stall | float |
| no\_of\_entries | int |
| cost\_per\_entry | float |

use \_\_init\_\_() method to initialize the variables.  
use the following methods in **Exhibition** class to perform the various operations.

|  |  |  |
| --- | --- | --- |
| **SL no** | **Method name** | **Description** |
| 1 | cal\_total\_spent() | This method calculates the total amount spent and returns the value to the main function. |
| 2 | cal\_total\_revenue() | This method calculates the total revenue generated and returns the value to the main function. |
| 3 | profit\_or\_loss(total\_spent, total\_revenue) | This method takes ‘total\_spent and total\_revenue’ and compare the values and prints **‘Profit’** if there is profit, and **‘Loss’**if there is a loss, and ‘**No profit, No loss**’ if there is neither a profit nor a loss. |
| 4 | display(total\_spent, total\_revenue) | This method takes ‘total\_spent and total\_revenue’ as arguments and prints the ‘Number of stall, Total spent and Total revenue’. |

Create another class **StageEvent** with the following attribute

|  |  |
| --- | --- |
| **variable** | **Datatype** |
| no\_of\_events | int |
| cost\_per\_event | float |
| no\_of\_entries\_list | int list |
| cost\_per\_entry\_list | float list |

use \_\_init\_\_() method to initialize the variables.  
use the following methods in **StageEvent** class to perform the various operations.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **SL no** | **Method name** | **Description** |
|  | 1 | cal\_total\_spent() | This method calculates the total amount spent and returns the value to the main function. |
|  | 2 | cal\_total\_revenue() | This method calculates the total revenue generated and returns the value to the main function. |
|  | 3 | profit\_or\_loss(total\_spent, total\_revenue) | This method takes ‘total\_spent and total\_revenue’ and compares the values and prints **‘Profit’** if there is profit, or **‘Loss’**if there is a loss, or ‘**No profit, No loss**’ if there is neither a profit nor a loss. |
|  | 4 | display(total\_spent, total\_revenue) | This method takes ‘total\_spent and total\_revenue’ as arguments and prints the ‘**Number of stall, Total spent and Total revenue**’. |

Use the menu for selecting the event type.  
Menu  
1.Exhibition  
2.Stage Event  
once the event is selected, the corresponding inputs are considered  
Refer sample input-output format for better understanding.

Note: Total cost spent and Total revenue is a float value with one decimal precision.

**[Strictly adhere to the Object-Oriented specifications given in the problem statement.**  
**All text in bold corresponds to the input and rest corresponds to output]**  
  
**Sample Input and Output 1:**

Menu

1.Exhibition

2.stage Event  
Enter Choice  
**1**

Enter number of stalls

**4**

Enter cost per stalls

**23**

Enter number of entries to exhibition

**423**

Enter the entry fee

**2342**

Number of stall is 4

Total cost spent is 92.0

Total revenue is 990666.0

Profit

**Sample Input and Output 2:**  
Menu

1.Exhibition

2.stage Event  
Enter Choice  
**2**

Enter number of Events

**3**

Enter cost per Event

**15000**

Enter the number of entries for 1 event

**1000**

Enter the entry fee

**10**

Enter the number of entries for 2 event

**150**

Enter the entry fee

**15**

Enter the number of entries for 3 event

**200**

Enter the entry fee  
**10**

Number of events is 3

Total cost spent is 45000.0

Total revenue is 14250.0

Loss

Top of Form

Bottom of Form

Code template

**Main.py**

from Exhibition import Exhibition

from StageEvent import StageEvent

print("Menu\n1.Exhibition\n2.Stage Event")

n = int(input("Enter Choice\n"))

#Fill your code here

**StageEvent.py**

from abc import ABCMeta

from Event import Event

class StageEvent(Event):

def \_\_init\_\_(self,no\_of\_events, cost\_per\_event, no\_of\_entries, cost\_per\_entry):

self.no\_of\_events = no\_of\_events

self.cost\_per\_event = cost\_per\_event

self.no\_of\_entries = no\_of\_entries

self.cost\_per\_entry = cost\_per\_entry

def cal\_total\_spent(self):

#Fill your code here

def cal\_total\_revenue(self):

#Fill your code here

def display(self, total\_spent, total\_revenue):

#Fill your code here

def profit\_or\_loss(self, total\_spent, total\_revenue):

#Fill your code here

**Event.py**

from abc import ABCMeta

class Event:

\_\_metaclass\_\_=ABCMeta

def cal\_total\_spent(self):

pass

def cal\_total\_revenue(self):

pass

def profit\_or\_loss(self):

pass

**Exibition.py**

from abc import ABCMeta

from Event import Event

class Exhibition(Event):

def \_\_init\_\_(self,no\_of\_stalls,cost\_per\_stall, no\_of\_entries, cost\_per\_entry):

self.no\_of\_stalls = no\_of\_stalls

self.cost\_per\_stall = cost\_per\_stall

self.no\_of\_entries = no\_of\_entries

self.cost\_per\_entry = cost\_per\_entry

def cal\_total\_spent(self):

#Fill your code here

def cal\_total\_revenue(self):

#Fill your code here

def display(self, total\_spent, total\_revenue):

#Fill your code here

def profit\_or\_loss(self, total\_spent, total\_revenue):

#Fill your code here

**Folder hierarchy to be followed:**

**PythonAssignments\** Day1\_20Feb \Q1.py to Q3.py

\ Day2\_21Feb \ Q1.py to Q6.py

\Day3\_22Feb \ Q1.py to Q4.py

\Day4\_1 Mar \ Q1.py to Q3.py

\Day5\_2 Mar \ Q1.py

\Day6\_7\_Mar7,8 \ Q1.py to Q8.py

\Day8\_Mar9 \ Q1.py to Q3.py

\Day9\_10\_Mar14\_15 \Q1.py to Q2.py

\Day11\_Mar20 \Q1.py to Q2.py

\Day12\_Mar21 \Q1.py to Q2.py

\Day13\_Mar28 \Q1.py

\Day16\_Apr 4 \Q1.py to Q2.py