

NAME - KANWALJIT SINGH (62)
SAP ID – 500044606, Roll - 62
CCVT – 6th SEM

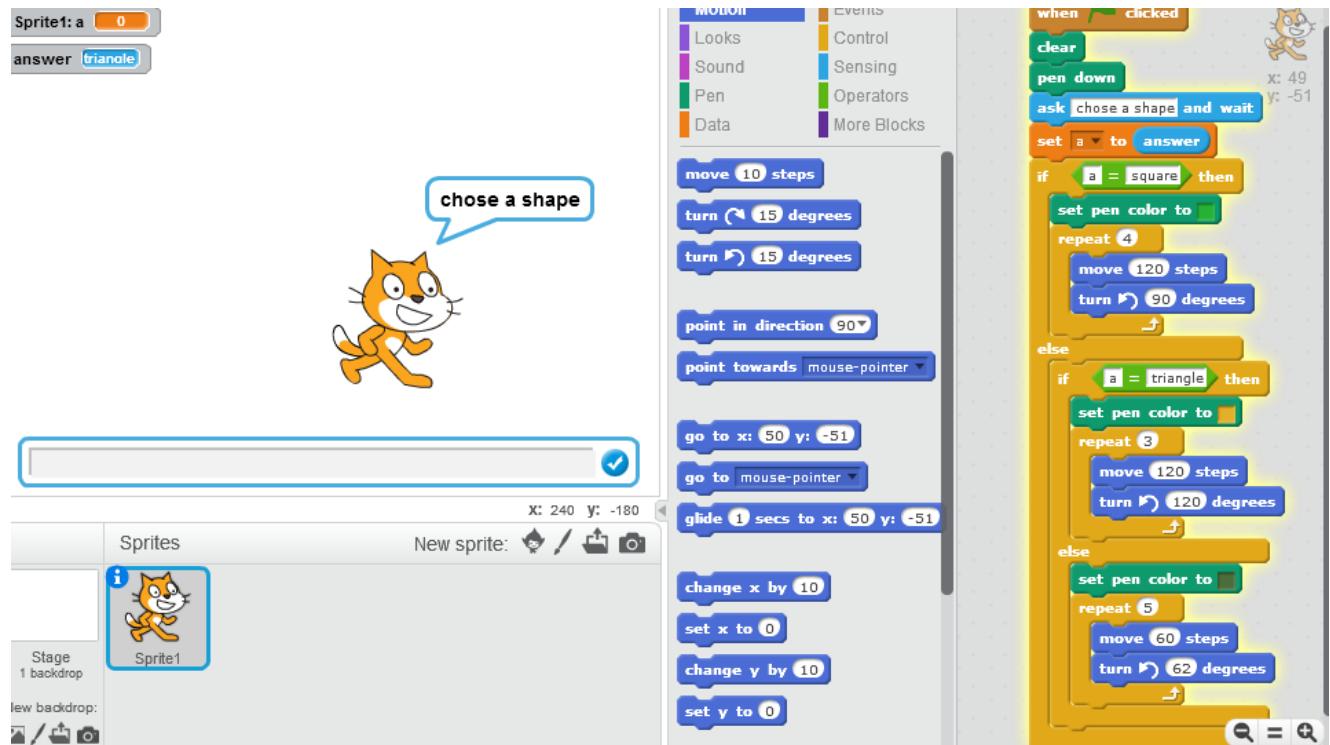
Submitted To: Mr. Deepak Kumar Sharma
SIGNATURE/REMARKS

FP5.0 Module-1 Assignments

Batch Name:	Infosys FP5.0 Summer Internship 2018
Enrollment No:	R110215062
SAP ID:	500044606
NAME:	Kanwaljit Singh
Semester:	VI
Branch:	CSE CCVT

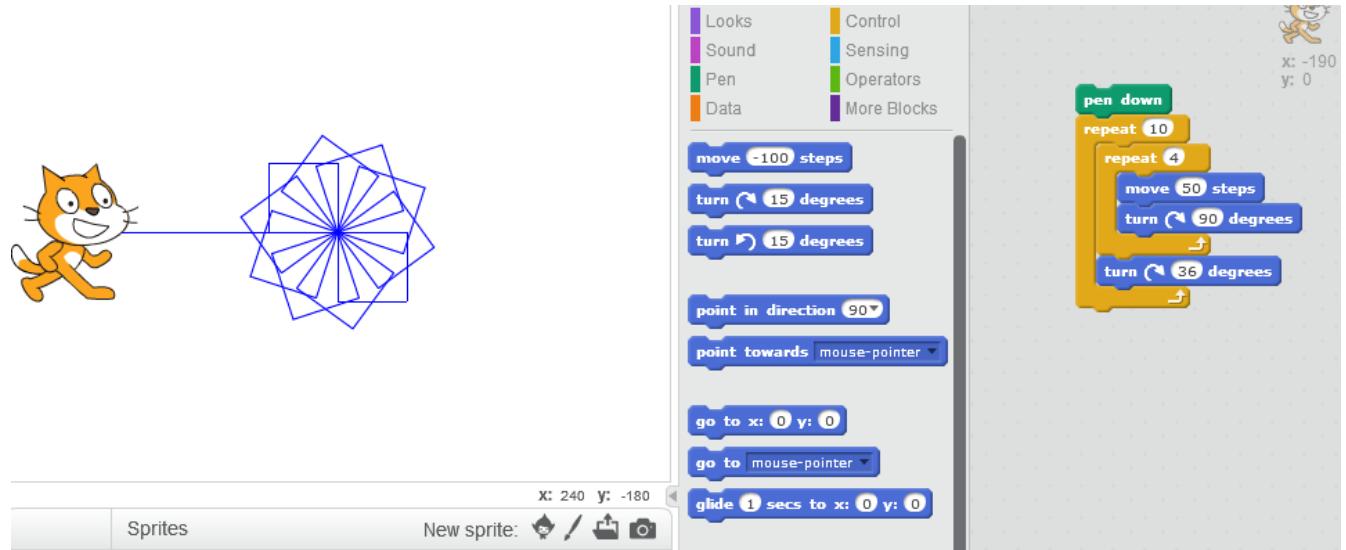
Assignment-1

1. Take a text input from the user as one of the three shape names – "square", "triangle" or "pentagon". Based on the input, draw either a red square, yellow triangle or black pentagon.

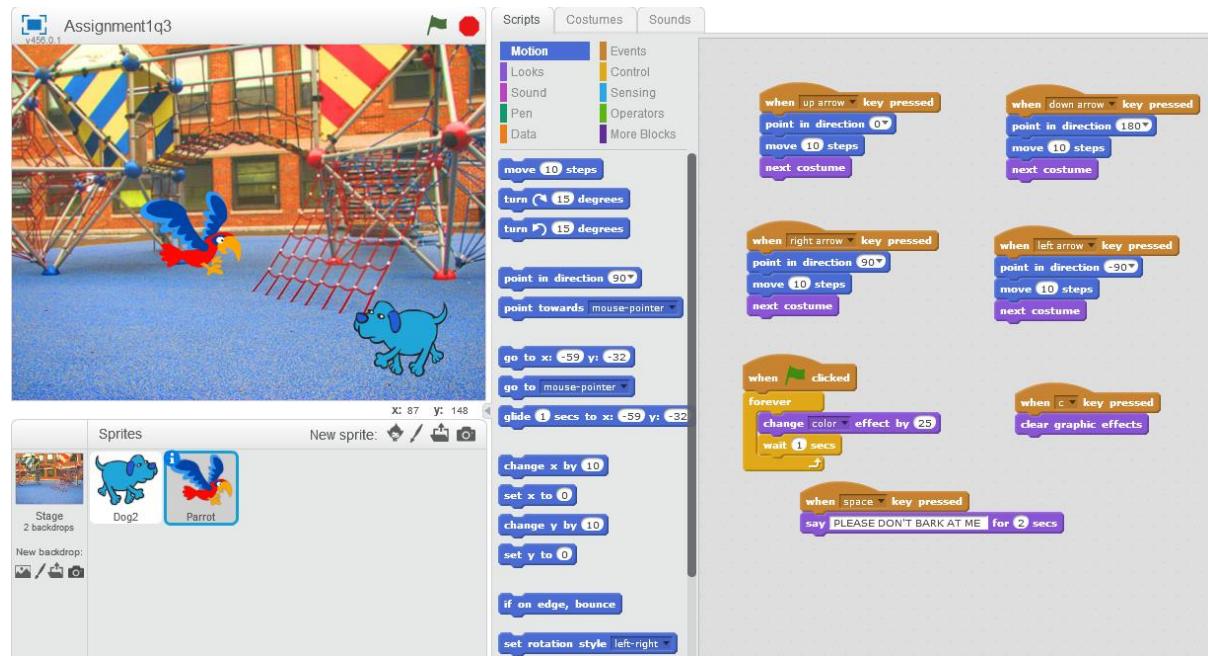


2. Create the following pattern using Scratch.

Create 10 squares each with an inclination of 36 degrees from the preceding square.

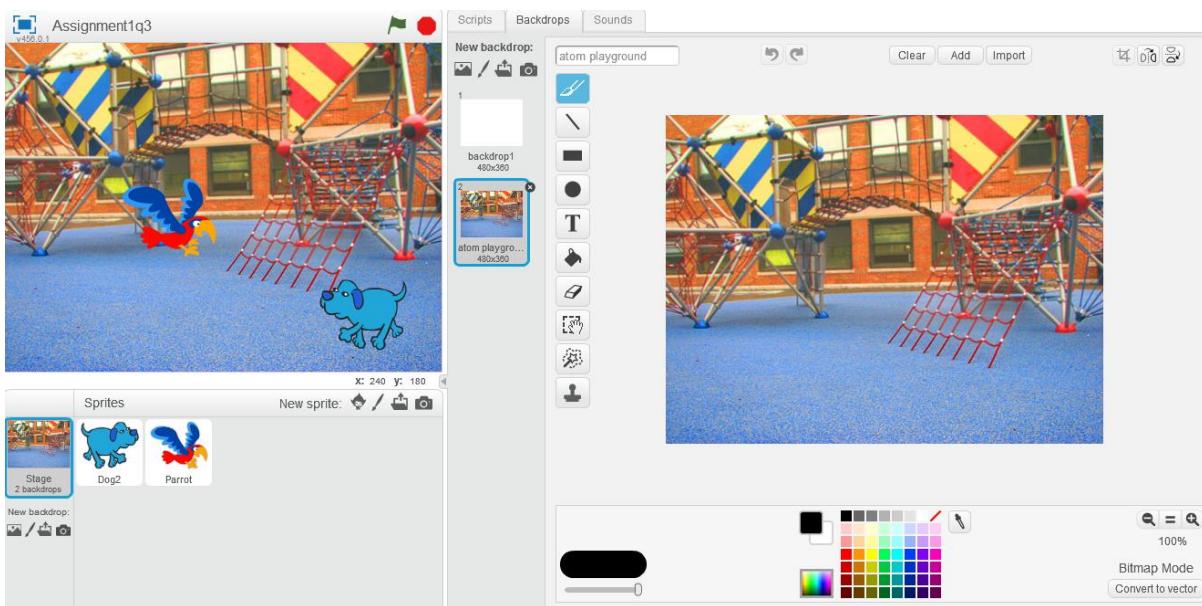
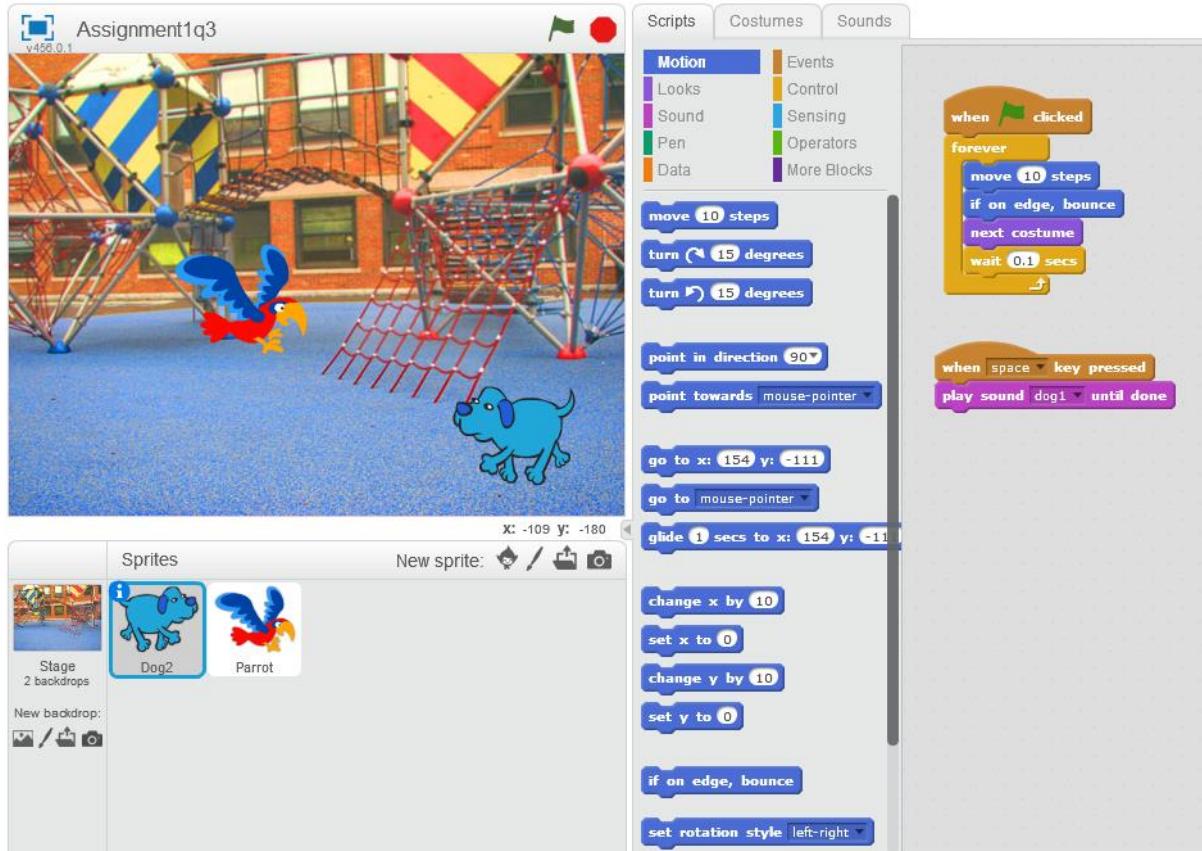


3. Create or import two Sprites and use your imagination to make them do different actions simultaneously. For example: "A bird is flying and a dog is walking at the same time."



NAME - KANWALJIT SINGH (62)
SAP ID – 500044606, Roll - 62
CCVT – 6th SEM

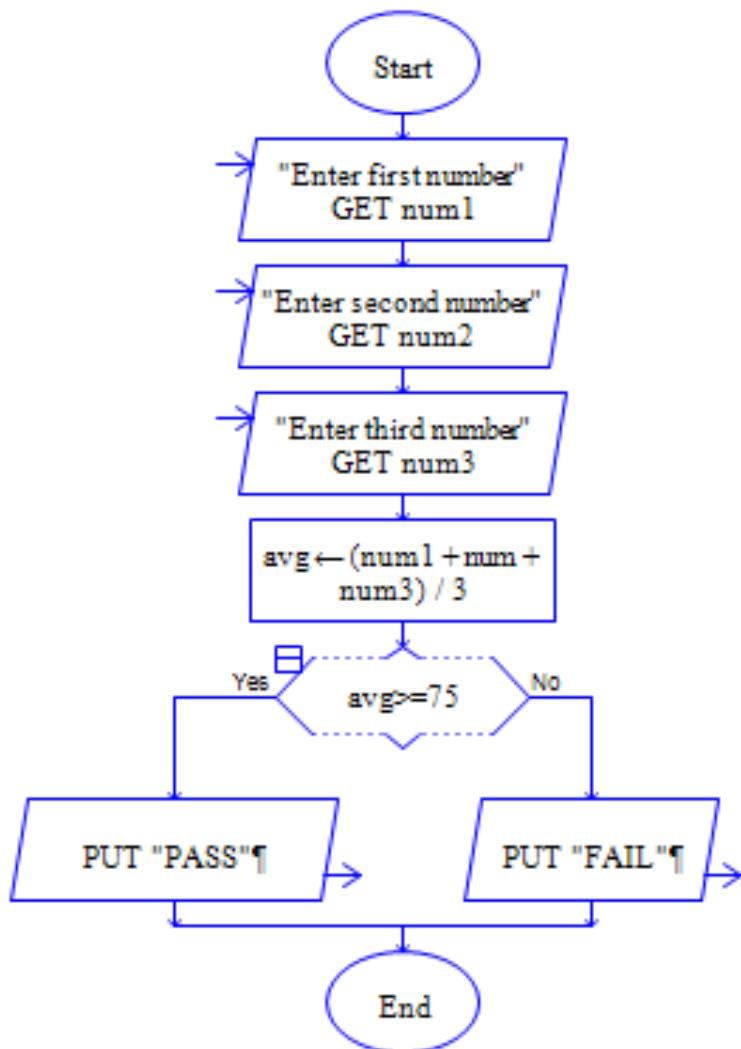
Submitted To: Mr. Deepak Kumar Sharma
SIGNATURE/REMARKS



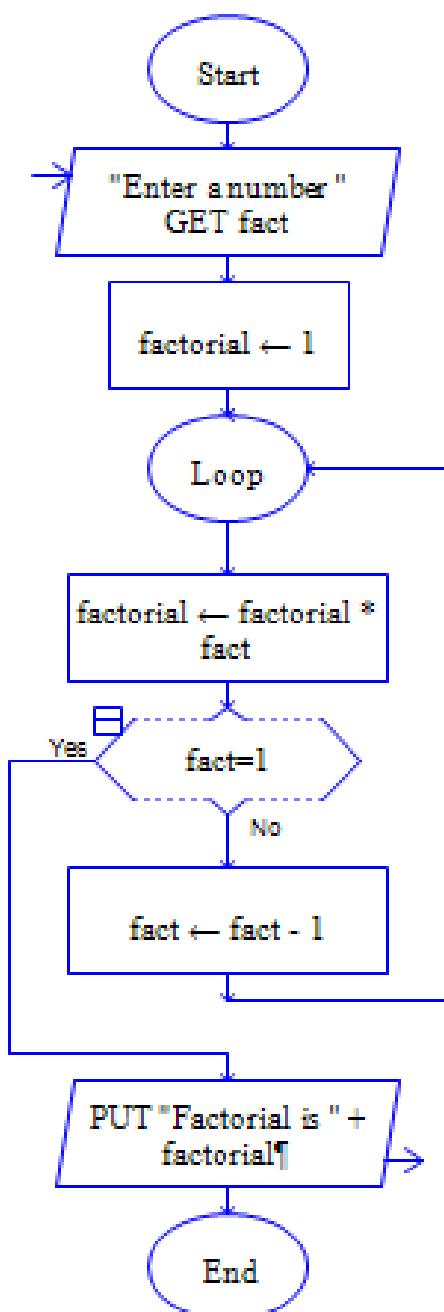
ASSIGNMENT-2

Create flowcharts for the following problems:

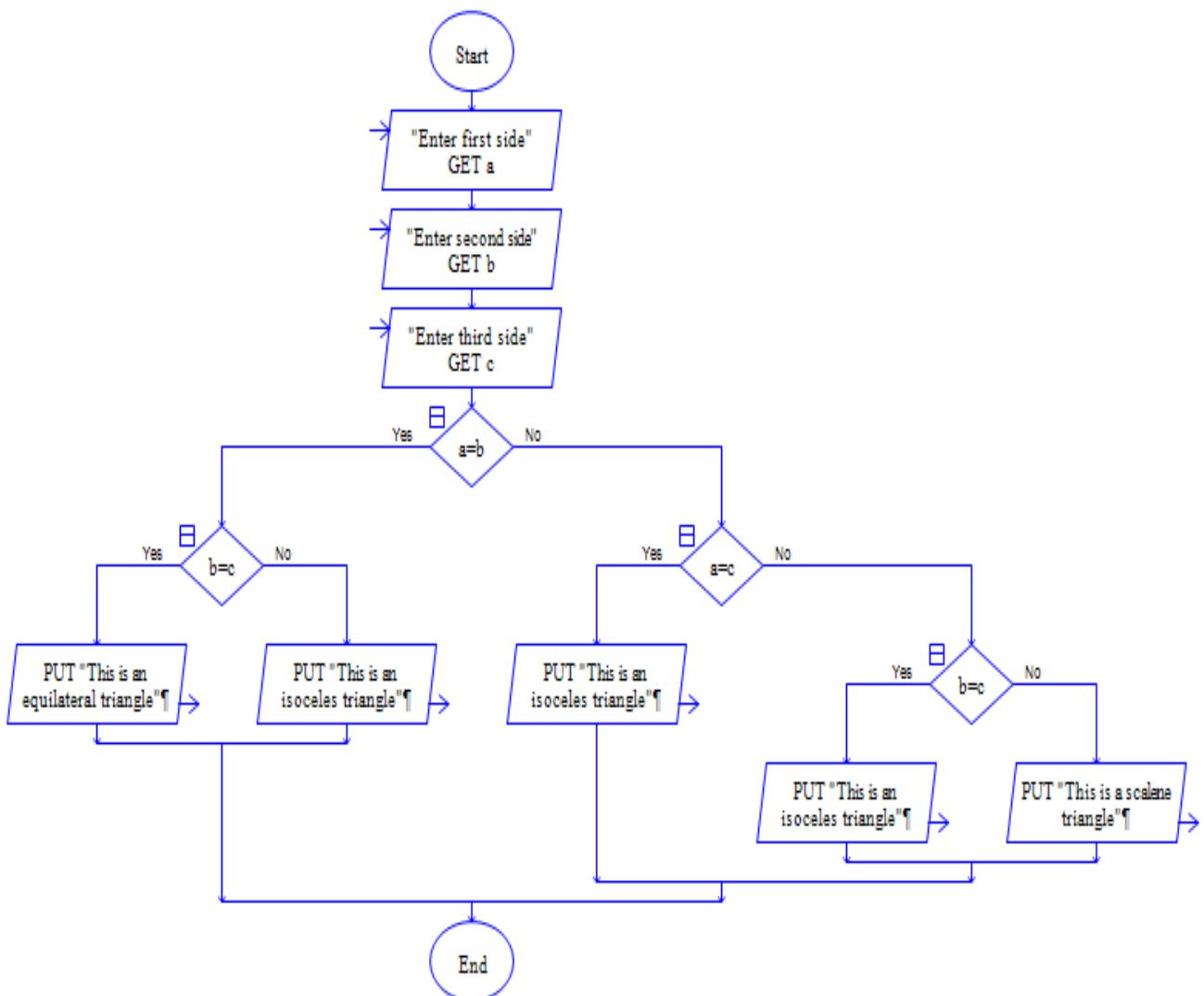
1. Calculate the average of three numbers. If average is greater than or equal to 75, print "Pass", else print "Fail".



2. Calculate and print the factorial of a number.



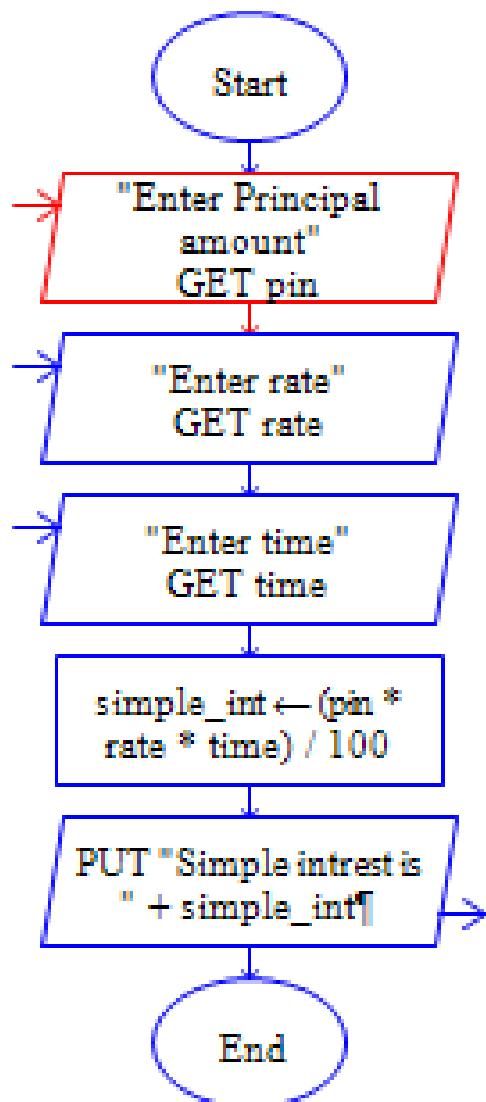
3. Accept the lengths of three sides of a triangle as input from the user. Based on the input, print if the given triangle is "Equilateral", "Isosceles" or "Scalene".



4. Accept the values of principal amount, rate of interest and number of years as an input from the user. Calculate and print the simple interest.

Hint - Formula for simple interest:

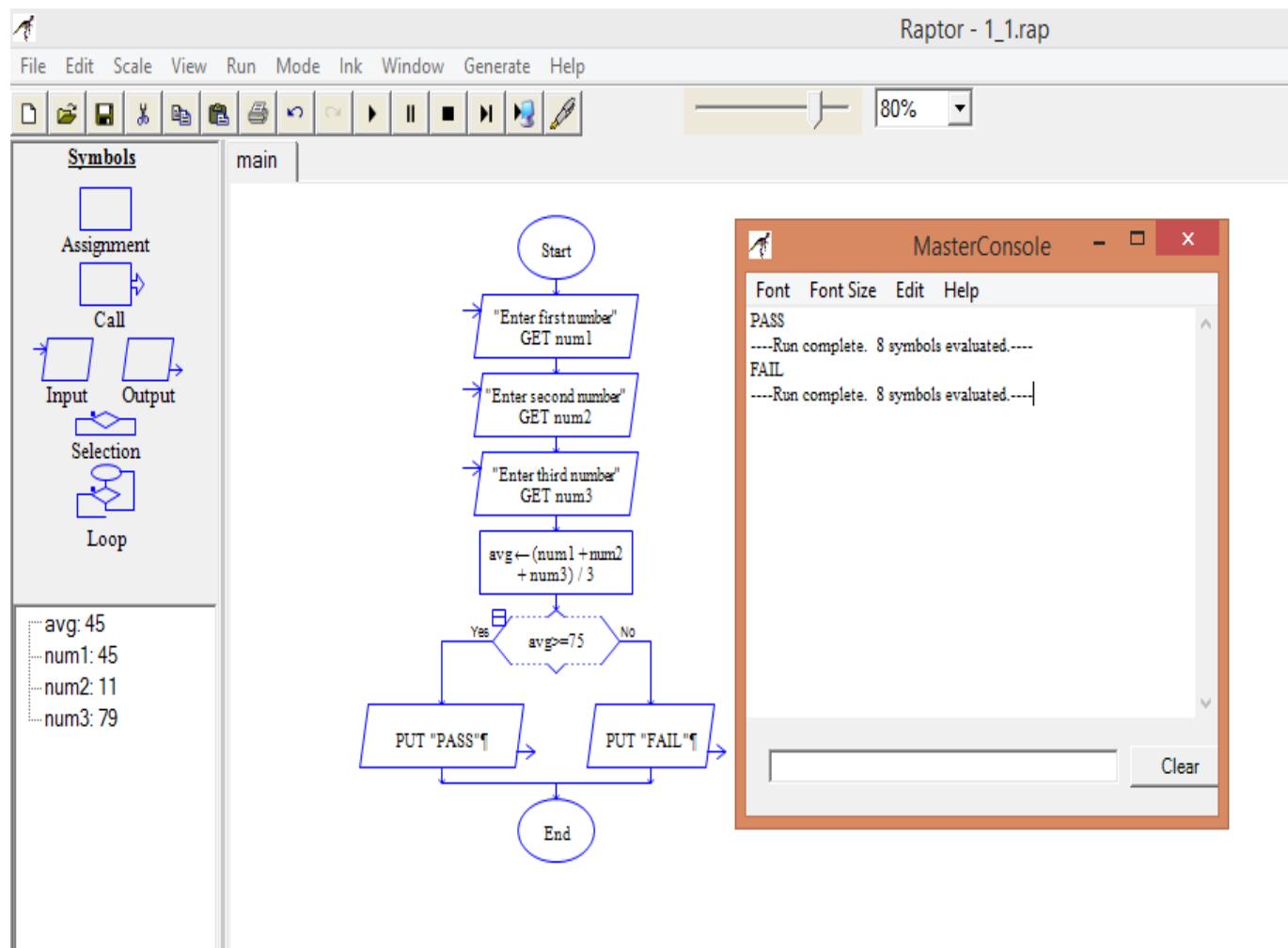
Simple Interest = (principal amount * rate of interest * number of years)/100



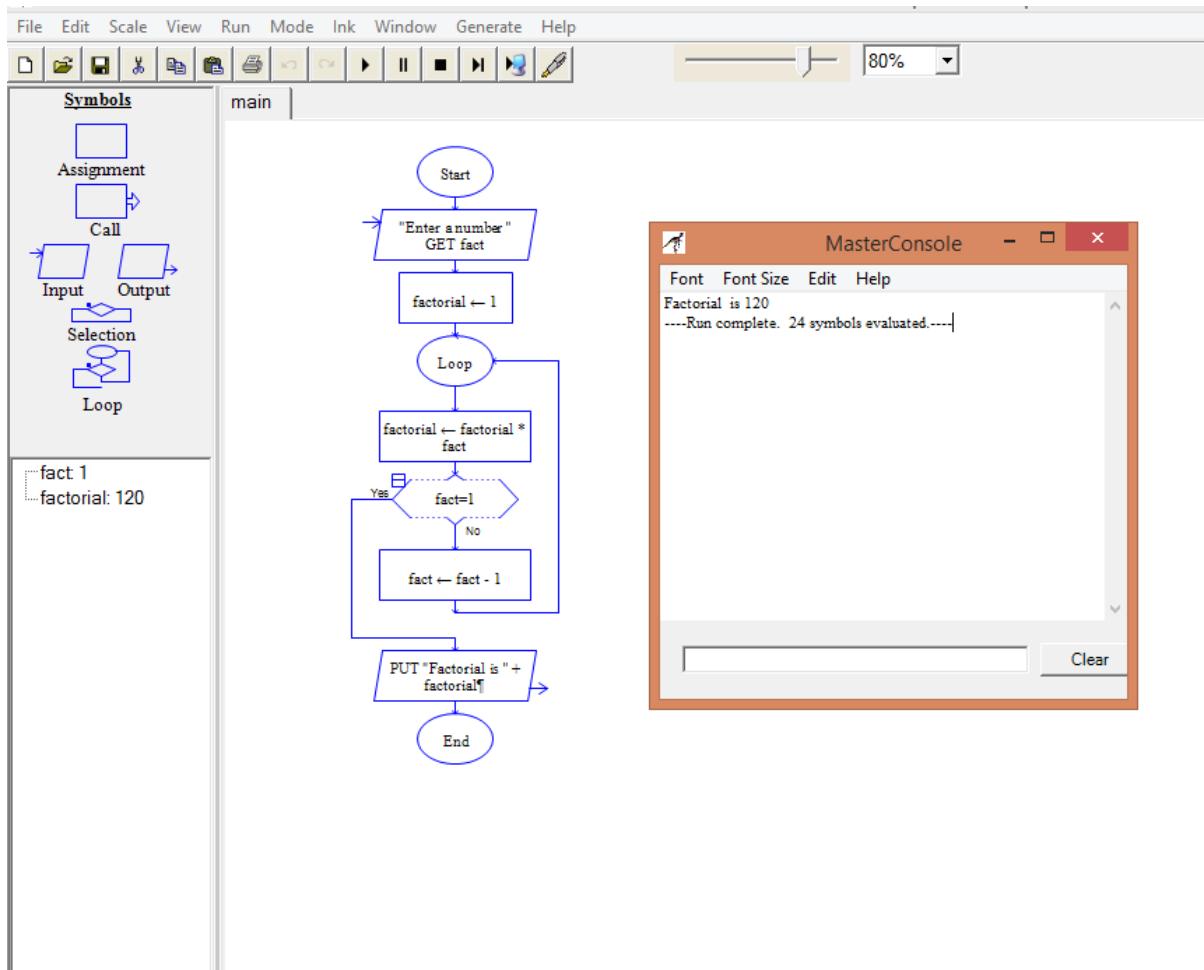
ASSIGNMENT-3

Create flowcharts for the following problems use Raptor tool to create and execute flowcharts for these problems. Observe the output for different set of inputs.

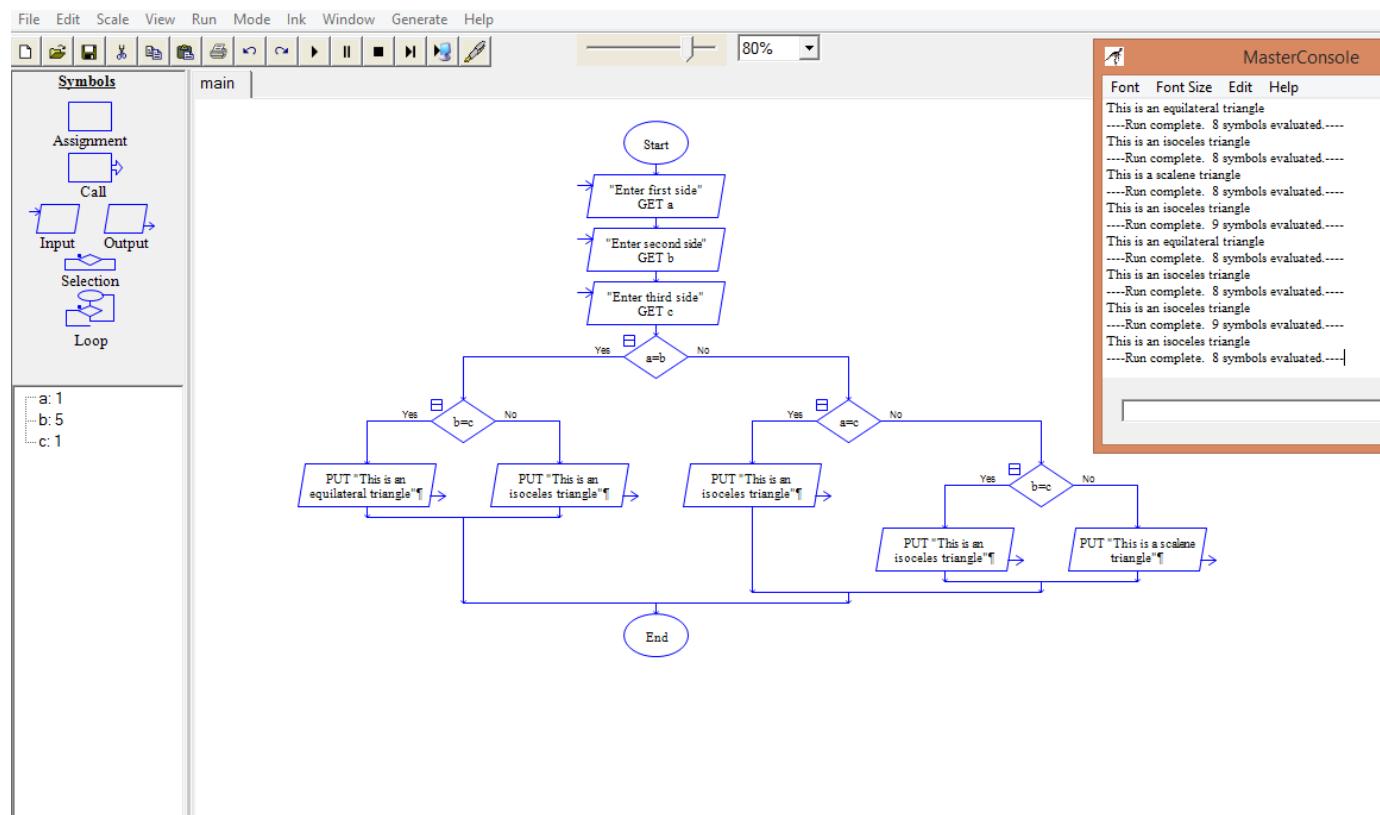
1. Calculate the average of three numbers. If average is greater than or equal to 75, print "Pass", else print "Fail".



2. Calculate and print the factorial of a number.



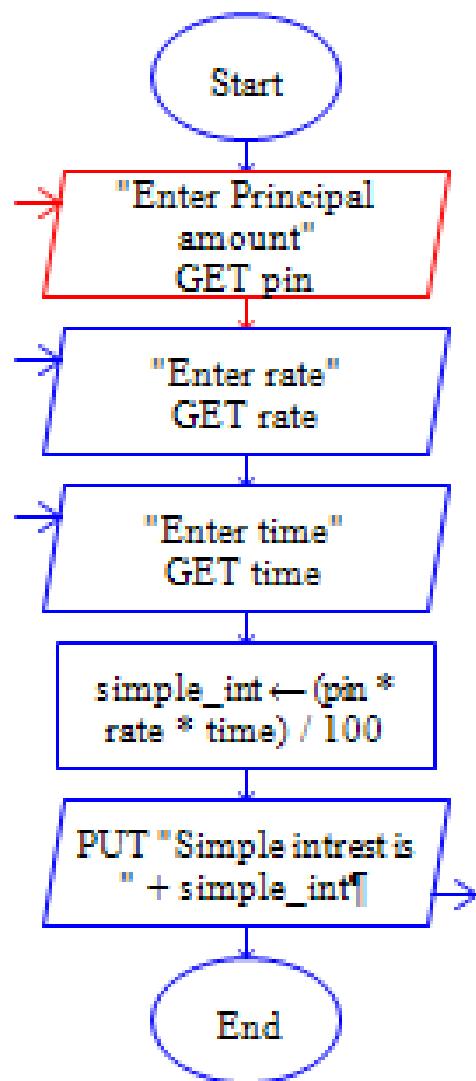
3. Accept the lengths of three sides of a triangle as input from the user. Based on the input, print if the given triangle is "Equilateral", "Isosceles" or "Scalene".



4. Accept the values of principal amount, rate of interest and number of years as an input from the user. Calculate and print the simple interest.

Hint - Formula for simple interest:

Simple Interest = (principal amount * rate of interest * number of years)/100



ASSIGNMENT- 4

Write Pseudo Code:

1. to check whether a given number is even or odd.

```
Start
input number
if(number/2) then
    Display "Number is even "
    else display "Number is odd "
    end if-else
end
```

2. to print the multiples of 3 between 1 to 20.

```
start
Display message "Multiples of 3 in 1 to 20 are :"
for(i=3;i<20;i=i+3)
    display value of i
    end for loop
end
```

3.to find factorial of a given number.

```
start
fact=1
Input num
while(num<1)
    fact*=num
    num--
end while
Display "factorial of given number is " + fact
end
```

4. to calculate ‘x’ to the power of ‘n’ using a while loop. Assume both ‘x’ and ‘n’ are positive whole numbers.

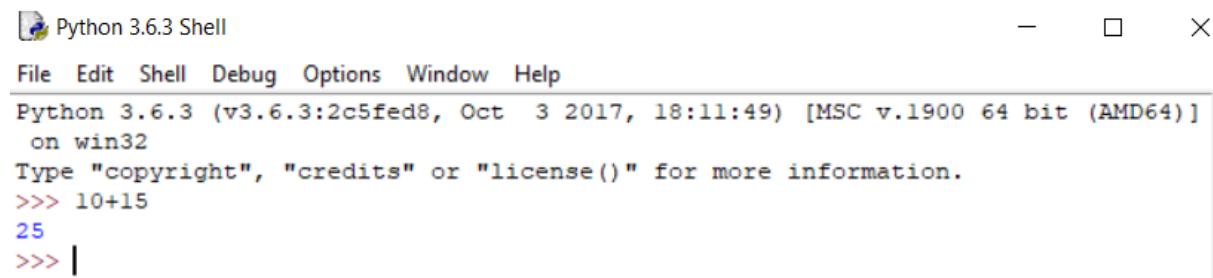
```
start
pow=1
Input x and n
for(i=1; i<=n;i++)
    pow=x*pow
    end for loop
Display "X to the power N is " + pow
end
```

ASSIGNMENT- 5

Open the Python IDLE and execute the following commands. Observe the output.

- 1)
 $10 + 15$
- 2)
`print("Hello World")`
- 3)
 $45 - 34$
- 4)
 $8 * 2$
- 5)
`print("Rahul's age is", 45)`
- 6)
`print("I have", 10, "mangoes and", 12, "bananas")`

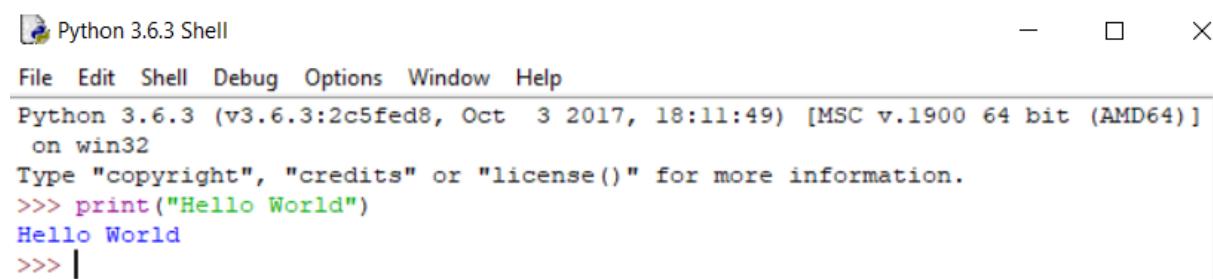
Duration : 10 mins



Python 3.6.3 Shell

File Edit Shell Debug Options Window Help

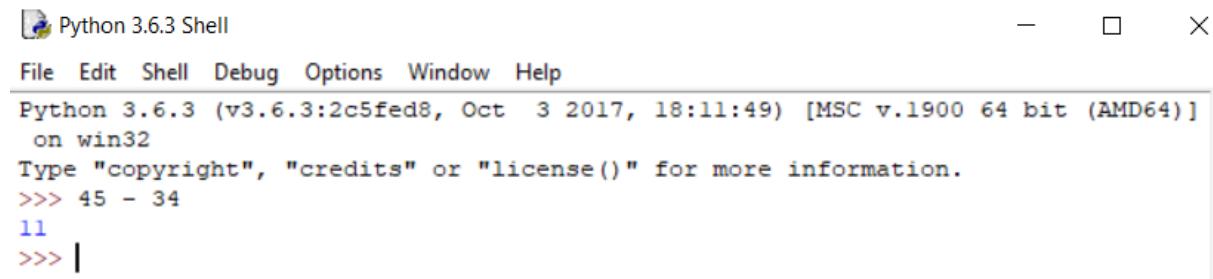
```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 18:11:49) [MSC v.1900 64 bit (AMD64)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> 10+15
25
>>> |
```



Python 3.6.3 Shell

File Edit Shell Debug Options Window Help

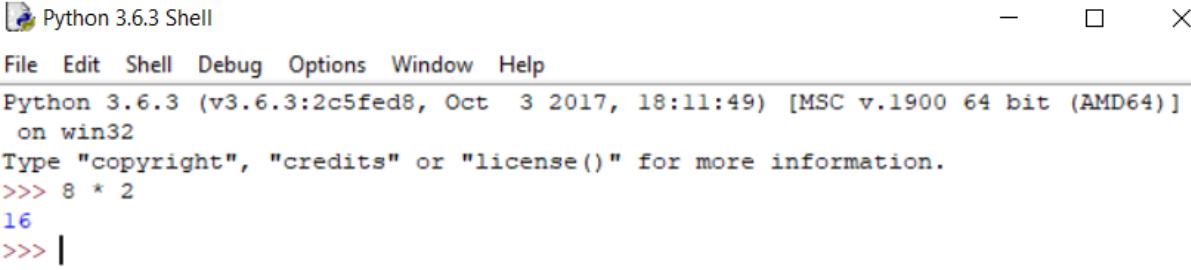
```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 18:11:49) [MSC v.1900 64 bit (AMD64)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("Hello World")
Hello World
>>> |
```



Python 3.6.3 Shell

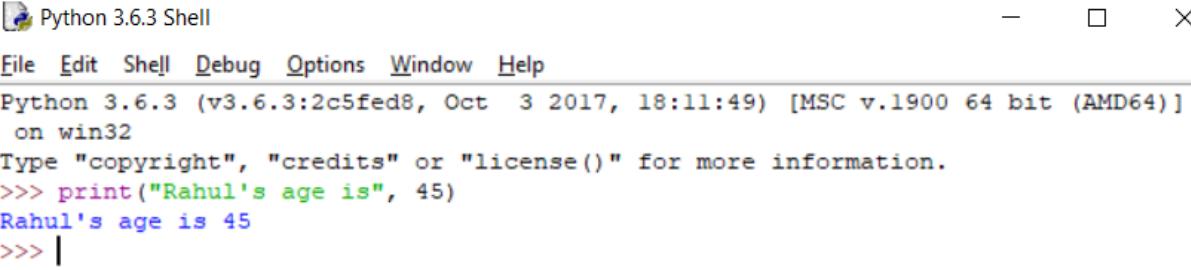
File Edit Shell Debug Options Window Help

```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 18:11:49) [MSC v.1900 64 bit (AMD64)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> 45 - 34
11
>>> |
```



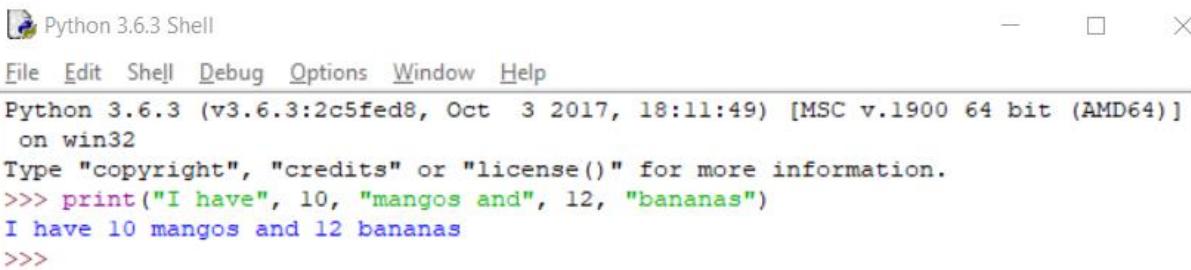
Python 3.6.3 Shell window showing the Python 3.6.3 interpreter. The command `>>> 8 * 2` is entered and results in `16`.

```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 18:11:49) [MSC v.1900 64 bit (AMD64)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> 8 * 2
16
>>> |
```



Python 3.6.3 Shell window showing the Python 3.6.3 interpreter. The command `>>> print("Rahul's age is", 45)` is entered and results in `Rahul's age is 45`.

```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 18:11:49) [MSC v.1900 64 bit (AMD64)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("Rahul's age is", 45)
Rahul's age is 45
>>> |
```



Python 3.6.3 Shell window showing the Python 3.6.3 interpreter. The command `>>> print("I have", 10, "mangos and", 12, "bananas")` is entered and results in `I have 10 mangos and 12 bananas`.

```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 18:11:49) [MSC v.1900 64 bit (AMD64)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("I have", 10, "mangos and", 12, "bananas")
I have 10 mangos and 12 bananas
>>> |
```

Assignment – 6

Open Python IDLE and execute the following commands. Observe the output.

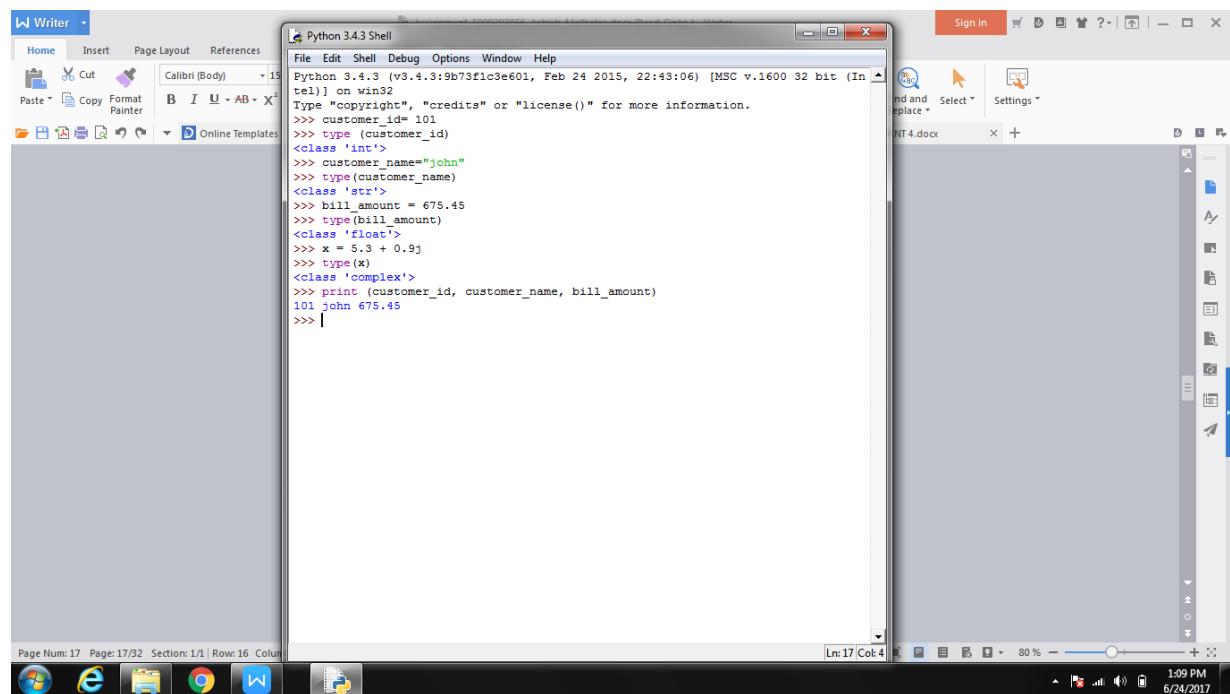
1. emp_number = 1233
2. Print ("Employee Number:", emp_number)
3. emp_salary = 16745.50
4. emp_name = "Jerry Squaris"
5. Print ("Employee Salary and Name:", emp_salary, emp_name)
6. emp_salary = 23450.34
7. Print ("Updated Employee Salary:", emp_salary)

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (In tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> emp_number=1233
>>> print("Employee number:",emp_number)
Employee number: 1233
>>> emp_salary = 16745.50
>>> emp_name = "Jerry Squaris"
>>> print("Employee salary and name:",emp_salary ,emp_name)
Employee salary and name: 16745.5 Jerry Squaris
>>> emp_salary = 23450.34
>>> print ("Updated Employee Salary:", emp_salary)
Updated Employee Salary: 23450.34
>>> |
```

Assignment – 7

Execute the following Python statements in IDLE and observe the output:

- customer_id = 101
- Type (customer_id)
- customer_name = "John"
- Type (customer_name)
- bill_amount = 675.45
- Type (bill_amount)
- $x = 5.3 + 0.9j$
- type(x)
- Print (customer_id, customer_name, bill_amount)



Assignment – 8

In a retail application, shopkeeper wants to keep a track of following details of a customer. Sample values are provided.

- bill_id = 101
- customer_id = 1001
- customer_name = "Rahul"
- if_minor = False
- bill_amount = 2000.50

Write a python program to store the details and display them.

The screenshot shows a Windows desktop environment. In the center, a terminal window titled 'Assignment8.py' is open, displaying Python code and its execution. The code prompts for bill_id, customer_id, customer_name, and bill_amount, then prints the results. The terminal window is part of a larger application window titled 'W Writer'. To the right of the terminal, a Microsoft Word document titled 'NT4.docx' is visible. The taskbar at the bottom shows several icons, including Python and other application icons. The system tray indicates the date as 6/24/2017 and the time as 1:15 PM.

```
Assignment8.py - D:\New folder\Assignment8.py (3.4.3)
File Edit Format Run Options Window Help
Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52) [MSC v.1900 32 bit (Intel)]
Type "copyright", "credits" or "license()" for more information.
>>> bill_id = input("Enter bill id: ")
Enter bill id: 101
>>> customer_id = input("Enter customer id: ")
Enter customer id: 1001
>>> customer_name = input("Enter customer name: ")
Enter customer name: Rahul
>>> bill_amount = input("Enter the bill amount: ")
Enter the bill amount: 2000.50
>>> print(bill_id, customer_id, customer_name, bill_amount)
101 1001 Rahul 2000.50
>>>
```

Assignment – 9

Execute the following commands and observe the usage of different types of commenting styles.

```
i = 10 # creates an integer variable. This is a single line comment.  
print("i =", i) # prints 10  
'''
```

Below code creates a Boolean variable in Python
(This is a multiple line comment)

```
'''
```

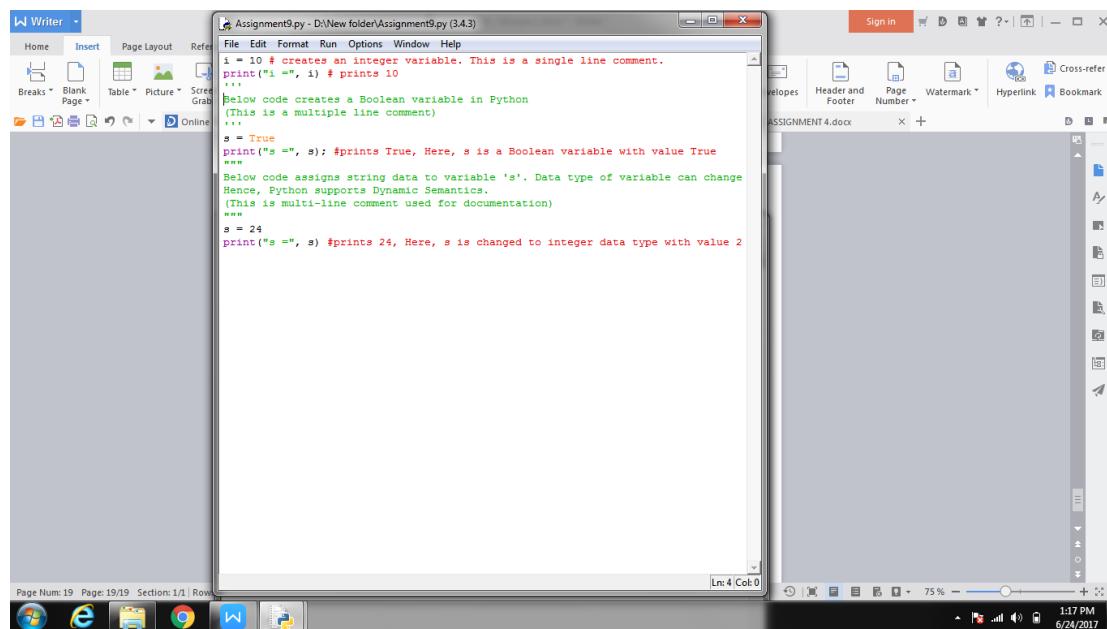
```
s=True  
print("s =", s) #prints True, Here, s is a Boolean variable with value True  
'''
```

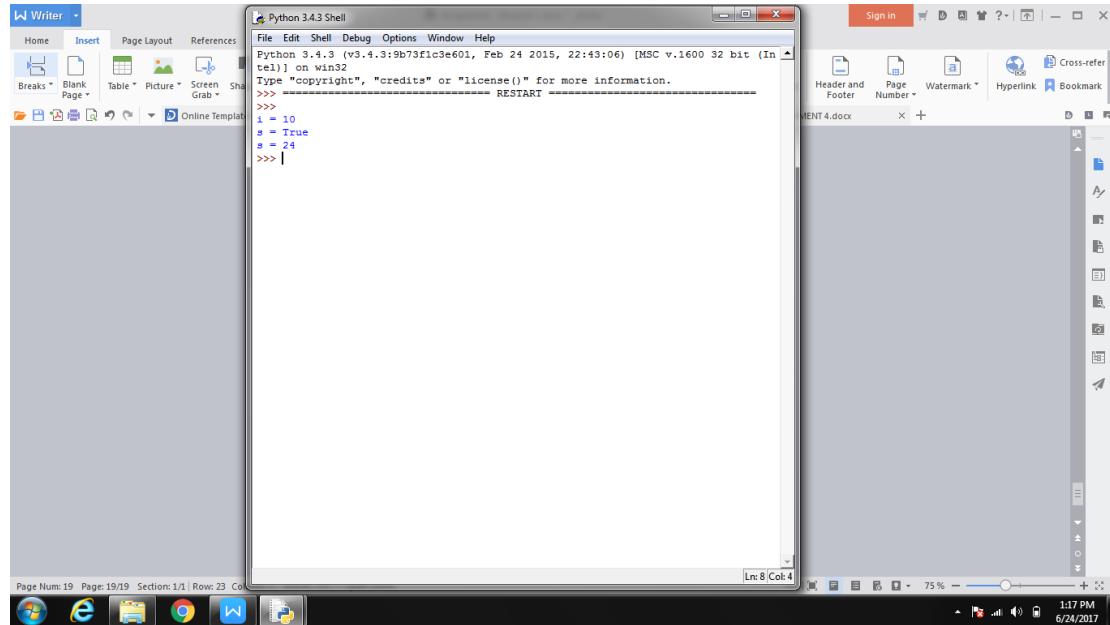
Below code assigns string data to variable 's'. Data type of variable can change during execution,

Hence, Python supports Dynamic Semantics.
(This is multi-line comment used for documentation)

```
'''
```

```
s = 24  
print("s =", s) #prints 24, Here, s is changed to integer data type with  
value 24
```



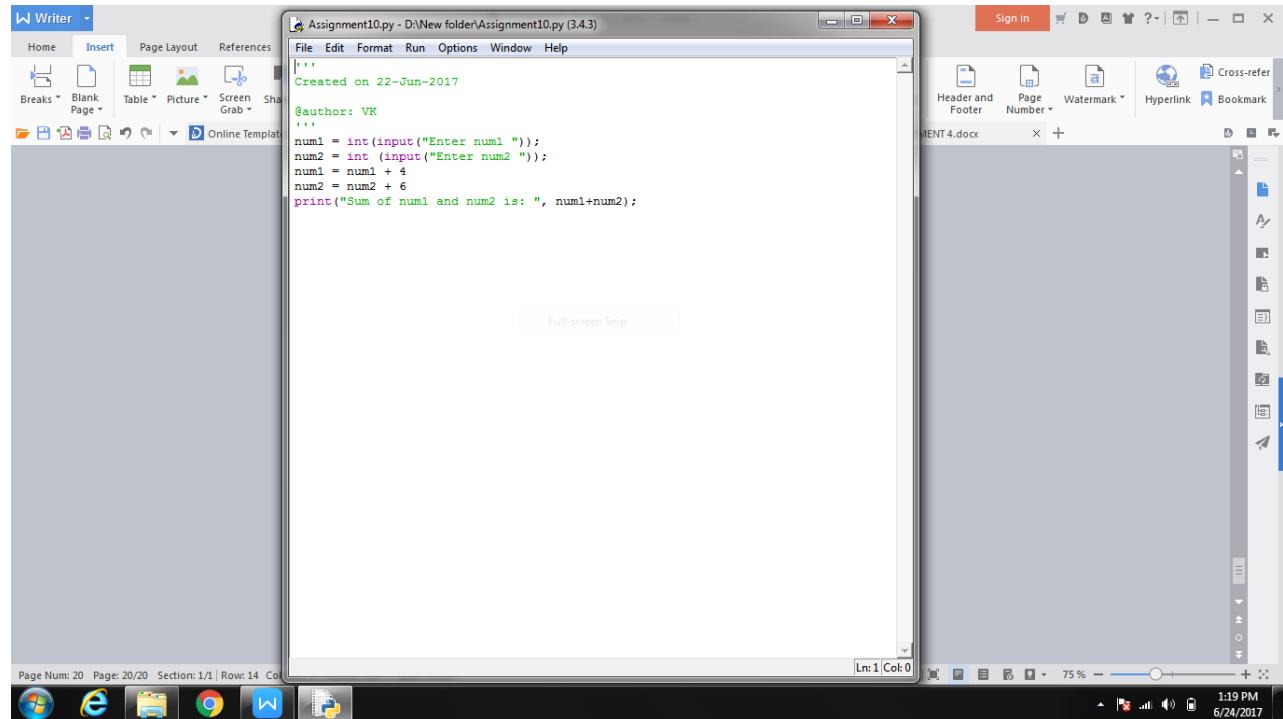


Assignment – 10

Write a Python program for the following requirements:

- Prompt the user to input two numbers num1 and num2
- Increment num1 by 4 and num2 by 6
- Find and print the sum of new values of num1 and num2

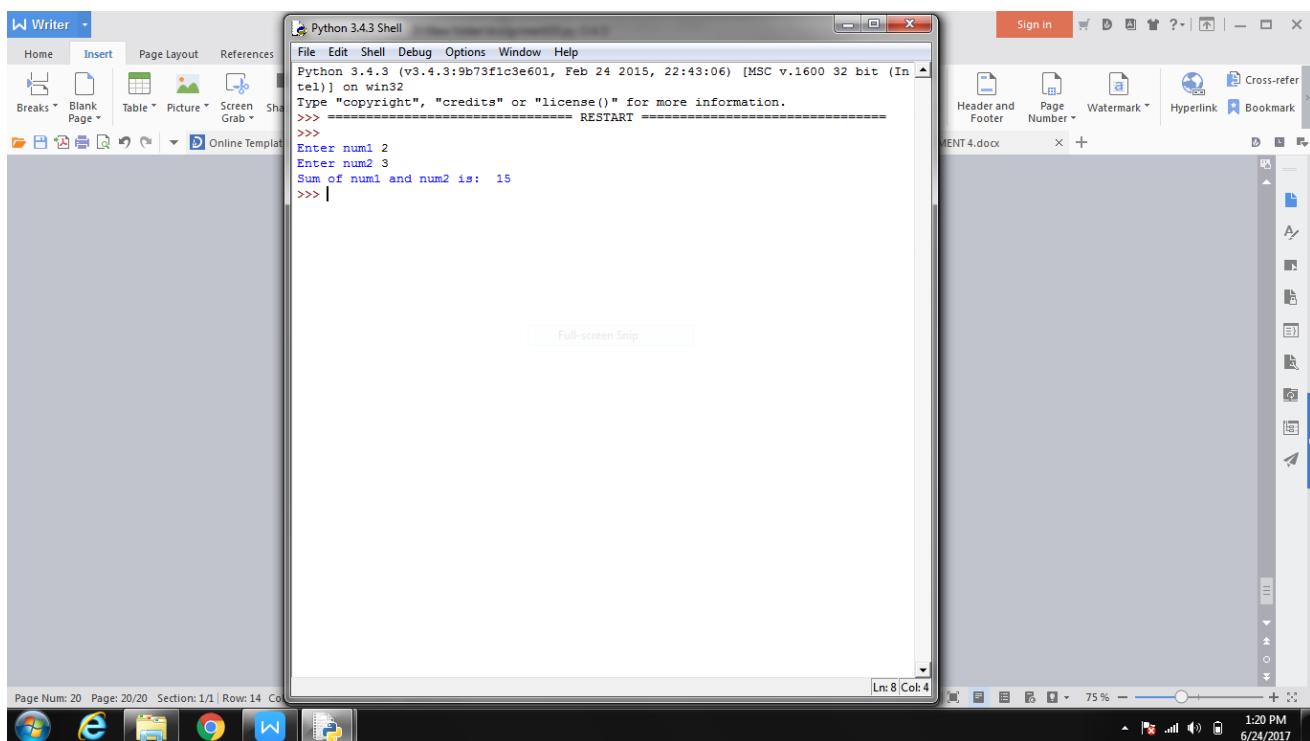
Hint – Use type casting for converting the input into an integer.



The screenshot shows a Windows desktop environment. In the center, there is a WPS Writer window titled "Assignment10.py - D:\New folder\Assignment10.py (3.4.3)". The window contains the following Python code:

```
'''Created on 22-Jun-2017@author: VK'''num1 = int(input("Enter num1 "));num2 = int (input("Enter num2 "));num1 = num1 + 4num2 = num2 + 6print("Sum of num1 and num2 is: ", num1+num2);
```

To the right of the WPS Writer window, there is a Microsoft Word document titled "MENT 4.docx". The status bar at the bottom of the screen displays "Page Num: 20 Page: 20/20 Section: 1/1 Row: 14 Col: 1". The system tray icons include the Start button, Task View, Edge browser, Google Chrome, File Explorer, and Python icon. The system clock shows "1:19 PM" and the date "6/24/2017".



Assignment – 11

- 1) Consider two variables 'a' and 'b' in Python such that $a = 4$ and $b = 5$. Swap the values of 'a' and 'b' without using a temporary variable. Print the values of 'a' and 'b' before and after swapping.
- 2) Consider the scenario of processing marks of a student in ABC Training Institute. John, the student of fifth grade takes exams in three different subjects. Create three variables to store the marks obtained by John in three subjects. Find and display the average marks scored by John. Now change the marks in one of the subjects and observe the output. Did the value of average change?
- 3) Given the value of radius of a circle, write a Python program to calculate the area and perimeter of the circle. Display both the values.
- 4) The finance department of a company wants to compute the monthly pay of its employees. Monthly pay should be calculated as mentioned in the formula below. Display all the employee details.

Monthly Pay = Number of hours worked in a week * Pay rate per hour *
No. of weeks in a month

A1)

The screenshot shows the LiClipse IDE interface. The PyDev Package Explorer view on the left lists various Python files and other project files like assignments.zip and bowling.txt. The central editor window contains the following code:

```
a=4
b=3
print("value of a and b before swapping",a,b)
a=a+b
b=a-b
print("value of a and b after swapping",a,b)
```

The Console view at the bottom shows the output of running the script:

```
<terminated> Assignment11.py [C:\Python34\python.exe]
value of a and b before swapping 4 3
value of a and b after swapping 3 4
```

A2)

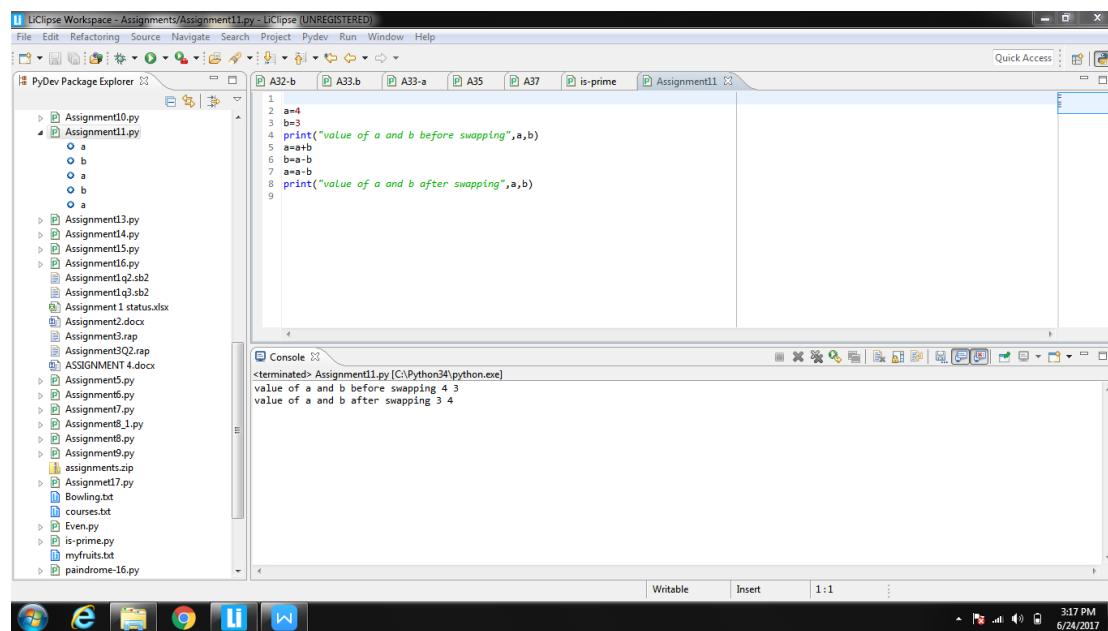
The screenshot shows the LiClipse IDE interface. The PyDev Package Explorer view on the left lists various Python files and other project files like assignments.zip and bowling.txt. The central editor window contains the following code:

```
sub1=int(input("enter the marks in sub1"));
sub2=int(input("enter the marks in sub2"));
sub3=int(input("enter the marks in sub3"));
result=(sub1+sub2+sub3)/3
print(result)
sub1=int(input("change the marks in sub1"));
print("changed average is",result)
```

The Console view at the bottom shows the output of running the script:

```
<terminated> Assignment11.py [C:\Python34\python.exe]
enter the marks in sub1 20
enter the marks in sub2 20
enter the marks in sub3 20
17.333333333333332
change the marks in sub1
changed average is 17.333333333333332
```

A3)



The screenshot shows the Eclipse IDE interface with the PyDev perspective. The Project Explorer on the left lists various Python files and other project-related files. The central editor window contains the following Python code:

```

1 a=4
2 b=3
3 print("value of a and b before swapping",a,b)
4 a=a+b
5 a=a-b
6 a=a-b
7 print("value of a and b after swapping",a,b)
8
9

```

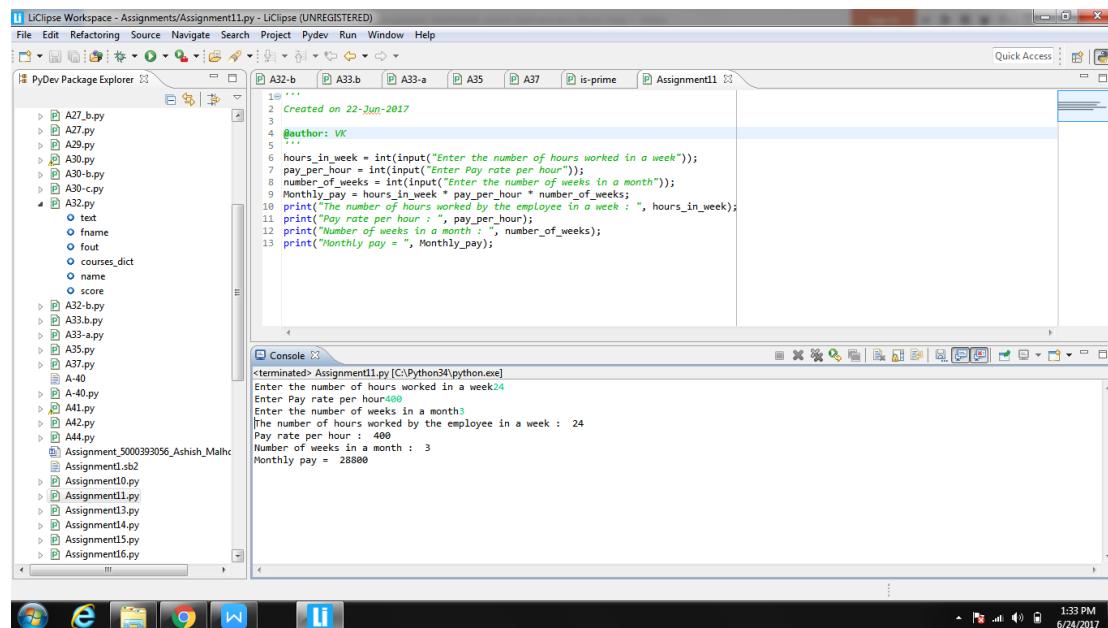
The Console window at the bottom shows the output of running the script:

```

<terminated> Assignment11.py [C:\Python34\python.exe]
value of a and b before swapping 4 3
value of a and b after swapping 3 4

```

A4)



The screenshot shows the Eclipse IDE interface with the PyDev perspective. The Project Explorer on the left lists various Python files and other project-related files. The central editor window contains the following Python code:

```

1 /**
2 * Created on 22-Jun-2017
3 */
4 #author: VK
5
6 hours_in_week = int(input("Enter the number of hours worked in a week"));
7 pay_per_hour = int(input("Enter Pay rate per hour"));
8 number_of_weeks = int(input("Enter the number of weeks in a month"));
9 Monthly_pay = hours_in_week * pay_per_hour * number_of_weeks;
10 print("The number of hours worked by the employee in a week : ", hours_in_week);
11 print("Pay rate per hour : ", pay_per_hour);
12 print("Number of weeks in a month : ", number_of_weeks);
13 print("Monthly pay = ", Monthly_pay);

```

The Console window at the bottom shows the output of running the script:

```

<terminated> Assignment11.py [C:\Python34\python.exe]
Enter the number of hours worked in a week24
Enter Pay rate per hour400
Enter the number of weeks in a month3
The number of hours worked by the employee in a week : 24
Pay rate per hour : 400
Number of weeks in a month : 3
Monthly pay = 28800

```

Assignment – 12

Identify the sections of the given program where the coding standards are not followed and correct them.

1. itemNo=1005
2. unitprice = 250
3. quantity = 2
4. amount=quantity*unitprice
5. print("Item No:", itemNo)
6. print("Bill Amount:", amount)

Ans) Lines which were corrected due to incorrect coding standard are marked in bold and comment has been attached explaining the coding standard which are being used.

```
item_no = 1005      # (variables should be in complete lowercase)
unit_price = 250    # (variables should have _ in between words for clear
understanding)
quantity = 2
amount = quantity * unit_price    # (leave a space before and after
binary operator for clear understanding)
Print("Item No:", item_no)
Print("Bill Amount:", amount)
```

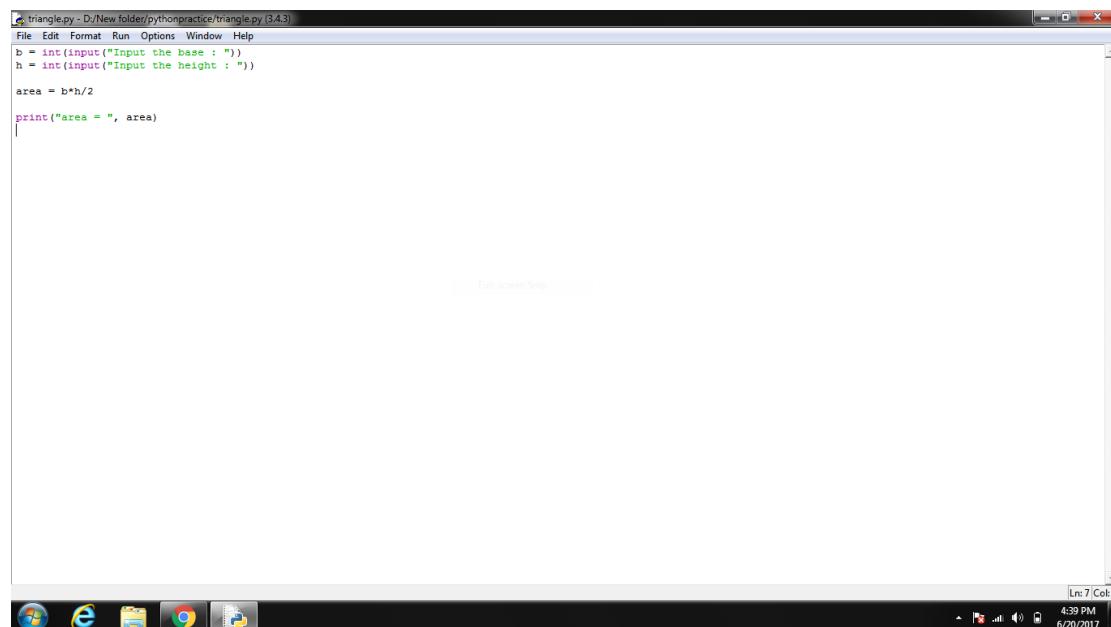
Assignment – 13

Create a new file in Python IDLE as "triangle.py"

- Write a Python program to calculate and print the area of the triangle.
Prompt the user to input the values for base and height of the triangle.
- Execute the program (use 'Run Module' under 'Run' tab) and observe the output.
- Close the file, open it again and execute it once more with different values. Observe the output.

Hint – Use type casting for converting the input into an integer.

Area of a triangle = $\frac{1}{2} * \text{base} * \text{height}$



A screenshot of the Python IDLE interface. The window title is "triangle.py - D:/New folder/pythonpractice/triangle.py [3.4.3]". The menu bar includes File, Edit, Format, Run, Options, Window, and Help. The code editor contains the following Python script:

```
b = int(input("Input the base : "))
h = int(input("Input the height : "))

area = b*h/2
print("area = ", area)
```

The status bar at the bottom right shows "Ln: 7 Col: 0", "4:39 PM", and "6/20/2017". The taskbar at the bottom of the screen shows icons for various applications, including the Python IDLE icon.

NAME - KANWALJIT SINGH (62)
SAP ID – 500044606, Roll - 62
CCVT – 6th SEM

Submitted To: Mr. Deepak Kumar Sharma
SIGNATURE/REMARKS

```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
Input the base : 3
Input the height : 4
area = 6.0
>>>
```

This screenshot shows a Python 3.4.3 Shell window. The user has inputted the base and height of a triangle, and the program has calculated the area as 6.0. The window title is 'Python 3.4.3 Shell'. The status bar at the bottom right shows the date and time as 6/20/2017 and 4:37 PM respectively.

```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
Input the base : 3
Input the height : 4
area = 6.0
>>> ===== RESTART =====
>>> 7
>>> ===== RESTART =====
>>>
Input the base : 8
Input the height : 6
area = 24.0
>>> |
```

This screenshot shows another instance of a Python 3.4.3 Shell window. The user has inputted different values for the base and height, resulting in an area of 24.0. The window title is 'Python 3.4.3 Shell'. The status bar at the bottom right shows the date and time as 6/20/2017 and 4:38 PM respectively.

Assignment – 14

1) Consider the scenario of retail store management again. The store provides discount for all bill amounts based on the criteria below:

Bill Amount	Discount %
≥ 1000	5
$\geq 500 \text{ and } < 1000$	2
$> 0 \text{ and } < 500$	1

Write a Python program to find the net bill amount after discount. Observe the output with different values of bill amount.

Assume that bill amount will be always greater than zero.

2) Extend the above program to validate the customer id. Customer ids in the range of 101 and 1000 (both inclusive) should only be considered valid.

Note: Display appropriate error messages wherever applicable.

Ans

The screenshot shows the LiClipse IDE interface. The top menu bar includes File, Edit, Refactoring, Source, Navigate, Project, PyDev, Run, Window, Help, and Quick Access. The left sidebar contains a PyDev Package Explorer with various Python files listed. The main editor window displays the following Python code:

```
1 bill_id = 1001;
2 customer_id = 1001;
3 bill_amount = 1200.0;
4 discount_amount = 0.0;
5 print("Bill id: ", bill_id);
6 print("Customer id: ", customer_id);
7 print("Bill amount: ", bill_amount);
8 if customer_id >= 101 and customer_id <= 1000:
9     if bill_amount >= 1000:
10         discount_amount = bill_amount - bill_amount * 5 / 100;
11     else:
12         if bill_amount >= 500 and bill_amount < 1000:
13             discount_amount = bill_amount - bill_amount * 2 / 100;
14         if bill_amount > 0 and bill_amount < 500:
15             discount_amount = bill_amount - bill_amount * 1 / 100;
16     else:
17         print("Invalid user");
18     print("Discounted Bill Amount: ", discount_amount);
19
20
```

The bottom console window shows the execution results:

```
<terminated> Assignment14.py [C:\Python34\python.exe]
Bill id: 1001
Customer id: 1001
Bill amount: 1200.0
Invalid user
Discounted Bill Amount:  0.0
```

Assignment – 15

Create four string variables a, b, c, d to store the following values and display them:

- My city is Mexico
- Raghu is my friend's brother
- My favorite programming language is "Python"
- Python is a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than possible in languages such as "C++" or "Java".

The screenshot shows the LiClipse IDE interface. The top menu bar includes File, Edit, Refactoring, Source, Navigate, Search, Project, PyDev, Run, Window, and Help. The title bar reads "LiClipse Workspace - Assignments/Assignment15.py - LiClipse (UNREGISTERED)". The left sidebar is the PyDev Package Explorer, listing various Python files and other project files like Assignment2.docx and Assignment4.docx. The main editor window contains the following Python code:

```
1 a="My city is Mexico"
2 b="Raghu is my friends brother"
3 c="My favorite programming Language is Python"
4 d="Python is a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design
5 philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer
6 lines of code than possible in languages such as C++ or Java."
7 print(a)
8 print(b)
9 print(c)
10 print(d)
11 input("Press enter to exit")
```

The bottom right corner of the editor shows the status bar with "Writable", "Insert", "1:1", "3:37 PM", and "6/24/2017". Below the editor is a Console window showing the execution of Assignment15.py and its output:

```
Assignment15.py[C:\Python34\python.exe]
My city is Mexico
Raghu is my friends brother
My favorite programming Language is Python
Python is a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design
philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer
lines of code than possible in languages such as C++ or Java.
Press enter to exit
```

Assignment – 16

- Accept a string as an input from the user. Check if the accepted string is palindrome or not.
- If the string is palindrome, print "String is palindrome", otherwise print "String is not palindrome".
- Also print the actual and the reversed strings.

Note – Ignore the case of characters.

Hint – A palindrome string remains the same if the characters of the string are reversed.

The screenshot shows the LiClipse IDE interface. The left pane displays a file tree with various Python files and a Word document. The central pane shows a code editor for a file named 'palindrome-16.py' containing the following Python code:

```
1@def is_palindrome(s):
2    if len(s) < 1:
3        return True
4    else:
5        if s[0] == s[-1]:
6            return is_palindrome(s[1:-1])
7        else:
8            return False
9 a=str(input("Enter string:"))
10 if(is_palindrome(a)==True):
11     print("String is a palindrome!")
12 else:
13     print("String isn't a palindrome!")
```

The right pane shows a 'Console' window with the output of running the script. The command 'Enter string:ra' was entered, and the response 'String is a palindrome!' is displayed.

ASSIGNMENT -17

Accept two strings 'string1' and 'string2' as an input from the user. Generate a resultant string, such that it is a

concatenated string of all upper case alphabets from both the strings in the order they appear. Print the actual and

the resultant strings.

Note: Each character should be checked if it is a upper case alphabet and then it should be concatenated to the

resultant string.

The screenshot shows the LiClipse IDE interface. The left pane displays the PyDev Package Explorer with various Python files and other project files like 'Assignment 1 status.xlsx'. The central editor pane contains the following Python code:

```
1 #!/usr/bin/python
2 # Created on 23-Jun-2017
3 #
4 #author: VK
5 #
6 i = 0
7 accepted_string = "An apple a day keeps a doctor away";
8 new_string = accepted_string.replace(" ", "")
9 result = ""
10 while i < len(new_string):
11     if(i % 2 == 0):
12         result = result + new_string[i];
13     i = i + 1;
14 print(result);
```

The right pane shows the Console output, which displays the result of running the script: 'Apedyepaotruwy'. The bottom status bar shows the date and time as 3:41 PM 6/24/2017.

ASSIGNMENT 18

Ques Given a string containing both upper and lower case alphabets. Write a Python program to count the number of occurrences of each alphabet(case insensitive) and display the same

The screenshot shows the LiClipse IDE interface. The top menu bar includes File, Edit, Refactoring, Source, Navigate, Search, Project, Pydev, Run, Window, Help, and Quick Access. The toolbar below has icons for New, Open, Save, Cut, Copy, Paste, Find, Replace, and others. The left sidebar displays a project structure under 'Assignments' with files like 'eclipse', 'Fibonacci', 'py.py', 'Foundation Program 5', 'pythonpractice', 'A18.py', 'ASSIGNMENT 4.docx', 'Even.py', 'pandrome-16.py', 'prime.py', 'string.py', 'Sum.py', and 'C:\Python34\python.e'. The main workspace shows the Python code for 'A18.py':

```
1@def count_dict(mystring):
2    d = {}
3    # count occurrences of character
4    for w in mystring:
5        d[w] = mystring.count(w)
6    # print the result
7    for k in sorted(d):
8        print (k + ':' + str(d[k]))
9
10 mystring='qwertyyawertyyyyy'
11 count_dict(mystring)
12 |
```

The 'Console' tab at the bottom shows the output of running the script:

```
<terminated> A18.py [C:\Python34\python.exe]
e: 2
q: 2
r: 2
t: 1
w: 2
y: 5
```

The status bar at the bottom right shows the date and time: 6/21/2017 4:13 PM.

ASSIGNMENT 19

Ques Write a Python program to accept a string 'accepted_string'. Generate resultant string 'resultant_string' such that 'resultant_string' should contain all characters at the even position of 'accepted_string'(ignoring blank spaces). Display 'resultant_string' in reverse order

The screenshot shows the LiClipse IDE interface. The top menu bar includes File, Edit, Refactoring, Source, Navigate, Search, Project, Pydev, Run, Window, Help, and Quick Access. The left sidebar is the PyDev Package Explorer, listing various Python files and other project files like assignments.zip and Bowling.txt. The central workspace contains a code editor window titled 'A33-a' with the following Python code:

```
i = 0
accepted_string = "An apple a day keeps a doctor away"
new_string = accepted_string.replace(" ", "")
result = ""
while i < len(new_string):
    if(i % 2 == 0):
        result = result + new_string[i];
    i = i + 1;
print(result);
print(result[::-1])
```

Below the code editor is a 'Console' window showing the terminal output:

```
<terminated> Assignment19.py [C:\Python34\python.exe]
Aapedyepaoatrwy
ywrtoapeydepaA
```

The bottom status bar displays the date and time: 4:44 PM 6/24/2017.

ASSIGNMENT -20

Ques Write a Python program to generate 'n' Fibonacci numbers where 'n' is accepted as an input from the user.

Store the generated Fibonacci numbers in a list and display the output

The screenshot shows the LiClipse IDE interface. The left pane displays a file tree with various projects and files, including 'Assignments', 'Fibonacci', and 'a15'. The main workspace shows a Python script named 'py.py' with the following code:

```
1 #This program calculates the Fibonacci sequence
2 a = 0
3 b = 1
4 count = 0
5 max_count = 20
6 while count < max_count:
7     count = count + 1
8     #we need to keep track of a since we change it
9     old_a = a
10    old_b = b
11    a = old_b
12    b = old_a + old_b
13#notice that the , at the end of a print statement keeps it
14 # from switching to a new line
15 print (old_a),
```

The bottom pane, titled 'Console', shows the output of running the script:

```
<terminated> py.py [C:\Python34\python.exe]
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
```

ASSIGNMENT -21

Ques: The "Variety Retail Store" sells different varieties of Furniture to the customers. The list of furniture available with its

respective cost is given below:

The furniture and its corresponding cost should be stored as a list. A customer can order any furniture in any quantity (the name and quantity of the furniture will be provided). If the required furniture is available in the furniture list(given above) and quantity to be purchased is greater than zero, then bill amount should be calculated. In case of invalid values for furniture required by the customer and quantity to be purchased, display appropriate error message and consider bill amount to be 0. Initialize required furniture and quantity with different values and test the results.

Write a Python program to calculate and display the bill amount to be paid by the customer based on the furniture bought and quantity purchased.

Hint – Create two diffrent lists for 'Furniture' and 'Cost'. Indices of two lists should be matched to retrieve the cost

of a particular furniture.

The screenshot shows the LiClipse IDE interface. The left pane displays the PyDev Package Explorer with various Python files and a Microsoft Word document. The central pane shows the code editor with Assignment21.py, which contains a script to calculate the total cost of furniture based on input. The right pane shows the Console window with the output of running the script. The bottom status bar indicates the date and time.

```
1 furniture = ["Sofa Set", "Dining Table", "T.V. Stand", "Cupboard"];
2 cost = [20000, 8500, 4599, 13920]
3 i = 0;
4 amount = 0;
5 item = input("Enter the Furniture you wanted to buy ")
6 quantity = int(input("Enter the quantity of the Furniture "))
7
8
9 while i < 4:
10     if item in furniture[i]:
11         if quantity > 0:
12             index_value = furniture.index(item)
13             cost_furniture = cost[index_value]
14             amount = quantity * cost_furniture
15         if quantity < 0:
16             print("Invalid Input")
17         i = i + 1;
18
19 print("Total cost is : ", amount)
```

<terminated> Assignment21.py [C:\Python34\python.exe]
Enter the Furniture you wanted to buy cupboard
Enter the quantity of the Furniture 20
Total cost is : 0

ASSIGNMENT -22

Ques-Consider the list of courses opted by a Student "John" and available electives at ABC Training Institute:

```
courses = ("Python Programming", "RDBMS", "Web Technology", "Software Engg.")
```

```
electives = ("Business Intelligence", "Big Data Analytics")
```

Write a Python Program to satisfy business requirements mentioned below:

1. List the number of courses opted by John.
2. List all the courses opted by John.
3. John is also interested in elective courses mentioned above. Print the updated tuple including electives.

The screenshot shows the Eclipse IDE interface with the following details:

- File Structure (PyDev Package Explorer):** Shows various Python files (Assignment13.py, Assignment14.py, etc.) and other files like assignments.zip and assignments.rar.
- Code Editor (Assignment22.py):** Contains the following Python code:

```
1 courses=["Python Programming", "RDBMS", "Web Technologies", "Software Engg."]
2 electives=["Business Intelligence", "Big-data analytics"]
3
4 print('Number of courses opted by John: ', len(courses))
5 print('Courses opted by John: ', courses)
6 courses.append(elective[0])
7 courses.append(elective[1])
8 print('Updated course list: ', courses)
```
- Console Output:** Displays the execution results:

```
<terminated> Assignment22.py [C:\Python34\python.exe]
Number of courses opted by John: 4
Courses opted by John: ['Python Programming', 'RDBMS', 'Web Technologies', 'Software Engg.']
Updated course list: ['Python Programming', 'RDBMS', 'Web Technologies', 'Software Engg.', 'Business Intelligence', 'Big-data analytics']
```

ASSIGNMENT -23

Ques: Given below is a dictionary 'customer_details' representing customer details from a Retail Application. Customer Id is the key and Customer Name is the value.

```
customer_details = { 1001 : "John", 1004 : "Jill", 1005: "Joe", 1003 : "Jack" }
```

Write Python code to perform the operations mentioned below:

- Print details of customers.
- Print number of customers.
- Print customer names in ascending order.
- Delete the details of customer with customer id = 1005 and print updated dictionary.
- Update the name of customer with customer id = 1003 to "Mary" and print updated dictionary.
- Check whether details of customer with customer id = 1002 exists in the dictionary.

The screenshot shows the Eclipse IDE interface with the following details:

- PyDev Package Explorer:** Shows various Python files in the workspace, including Assignment13.py, Assignment14.py, Assignment15.py, Assignment16.py, Assignment17.py, Assignment18.py, Assignment19.py, Assignment20.py, Assignment21.py, Assignment22.py, Assignment23.py, Assignment24.py, Assignment25.py, Assignment26.py, Assignment27.py, Assignment28.py, Assignment3.rap, Assignment3Q2.rap, Assignment4.docx, Assignment5.py, Assignment6.py, Assignment7.py, Assignment8.py, Assignment9.py, Assignment10.py, Assignment11.py, Assignment12.py, Assignment13.py, Assignment14.py, Assignment15.py, Assignment16.py, Assignment17.py, and Bowling.txt.
- Editor:** The active editor window (A33-a) contains the following Python code:

```
1 /**
2 * Created on 24-Jun-2017
3 *
4 * @author: VK
5 */
6 i = 0
7 customer_details = { 1001 : "John", 1004 : "Jill", 1005: "Joe", 1003 : "Jack" }
8 print("Customer details: ", customer_details, "\n");
9 print("Number of customers: ", len(customer_details), "\n");
10 lst = customer_details.values();
11 print("Customer name in ascending order: ", lst, "\n")
12 del customer_details[1005];
13 print("Updated customer_details: ", customer_details, "\n")
14 customer_details[1003] = "Mary"
15 print("Second Update of customer_details: ", customer_details, "\n")
16 while i < len(customer_details):
17     if 1002 in customer_details:
18         print("Key 1002 exist")
19     else:
20         print("Key 1002 does not exist")
21     i = i + 1
```

- Console:** The console output shows the execution of the code and its results:

```
<terminated> Assignment23.py [C:\Python34\python.exe]
Customer name in ascending order: dict_values(['John', 'Jack', 'Jill', 'Joe'])

Updated customer_details: {1001: 'John', 1003: 'Jack', 1004: 'Jill'}

Second Update of customer_details: {1001: 'John', 1003: 'Mary', 1004: 'Jill'}

Key 1002 does not exist
Key 1002 does not exist
Key 1002 does not exist
```

ASSIGNMENT -24

Ques: Consider a scenario from ABC Training Institute. The given table shows the marks scored by students of grade XI in Python Programming course.

Write a Python program to meet the requirements mentioned below:

- Display the name and marks for every student.
- Display the top two scorers for the course.
- Display class average of this course.

Hint- Implement the solution using a dictionary.

The screenshot shows the LiClipse IDE interface. The top menu bar includes File, Edit, Refactoring, Source, Navigate, Search, Project, Pydev, Run, Window, Help, and Quick Access. The left sidebar is the PyDev Package Explorer, listing various Python files and a status.xlsx file. The central workspace shows a code editor for Assignment24.py with the following content:

```
1
2 import operator
3 student_details = {"John" : 86.5, "Jack" : 91.2, "Jill" : 84.5, "Harry" : 72.1, "Joe" : 80.5}
4 print("Name and marks of the students are: ", student_details)
5 maximum = max(student_details[key] for key in student_details)
6 print(maximum, student_details[maximum])
7 average = sum(student_details.values()) / len(student_details)
8 print("Average Marks is: ", average)
```

The bottom console window shows the output of running the script:

```
<terminated> Assignment24.py [C:\Python34\python.exe]
Name and marks of the students are: {'Joe': 80.5, 'John': 86.5, 'Jill': 84.5, 'Jack': 91.2, 'Harry': 72.1}
Jack 91.2
Average Marks is: 82.96
```

ASSIGNMENT- 25

Ques: Consider the scenario from "Variety Retail Store" discussed in 'List' section. The list of furniture available with its

respective cost is given below:

A customer can order any furniture in any quantity. If the required furniture is available in the furniture list(given above) and quantity to be purchased is greater than zero, then bill amount should be calculated. In case of invalid values for furniture required by the customer and quantity to be purchased, display appropriate error message and consider bill amount to be 0. Initialize required furniture and quantity with different values and test the results.

Calculate and display the bill amount to be paid by the customer based on the furniture bought and quantity purchased. Implement the given scenario using:

1)List of tuples

2)Dictionary

The screenshot shows the LiClipse IDE interface. The top menu bar includes File, Edit, Refactoring, Source, Navigate, Search, Project, PyDev, Run, Window, Help, and Quick Access. The left sidebar displays the PyDev Package Explorer with various Python files and a status.xlsx file. The main workspace shows the code for Assignment25.py. The code defines four tuples (t1-t4) representing furniture items and their costs, creates a tuple list, initializes variables for item and amount, and uses a while loop to calculate the total cost based on user input. The bottom console window shows the execution of the script, where the user enters 'Sofa set' and '20000', resulting in a total cost of 0.

```
1 t1 = ("Sofa Set", 20000);
2 t2 = ("Dining Table", 8500);
3 t3 = ("TV Stand", 4599);
4 t4 = ("Cupboard", 13920);
5 tuple_list = [t1, t2, t3, t4]
6 i = 0;
7 amount = 0;
8 item = input("Enter the Furniture you wanted to buy ")
9 quantity = int(input("Enter the quantity of the Furniture "))
10
11
12 while i < 4:
13     if item in tuple_list[i]:
14         if quantity > 0:
15             cost_furniture = tuple_list[i][2]
16             amount = quantity * cost_furniture
17         if quantity < 0:
18             print("Invalid Input")
19         i = i + 1;
20
21 print("Total cost is : ", amount)
22
```

<terminated> Assignment25.py [C:\Python34\python.exe]
Enter the Furniture you wanted to buy Sofa set
Enter the quantity of the Furniture 20000
Total cost is : 0

ASSIGNMENT- 26

Ques: Consider a scenario from ABC Training Institute. Given below are two Sets representing the names of students

enrolled for a particular course:

```
java_course = {"John", "Jack", "Jill", "Joe"}
```

```
python_course = {"Jake", "John", "Eric", "Jill"}
```

Write a Python program to list the number of students enrolled for:

- 1) Python course
- 2) Java course only
- 3) Python course only
- 4) Both Java and Python courses
- 5) Either Java or Python courses but not both

Ans.

The screenshot shows the Eclipse IDE interface with the following details:

- File Structure (PyDev Package Explorer):** Shows various Python files (Assignment13.py, Assignment14.py, etc.) and other files like assignments.zip and Bowling.txt.
- Code Editor (Assignment26.py):** Contains the following Python code:

```
1'''  
2 Created on 24-Jun-2017  
3  
4 @author: VK  
5 ...  
6  
7 java_course = {"John", "Jack", "Jill", "Joe"}  
8 python_course = {"Jake", "John", "Eric", "Jill"}  
9 print("Number of students enrolled for python course: ", len(python_course))  
10 print("Number of students enrolled for Java course only: ", len(java_course - python_course))  
11 print("Number of students enrolled for Python course only: ", len(python_course - java_course))  
12 print("Number of students enrolled for both Python and course: ", len(python_course & java_course))  
13 print("Number of students enrolled for either Python or Java but not both: ", len(python_course ^ java_course))  
14 print("Number of students enrolled for either Python or Java: ", len(python_course | java_course))
```
- Console Output:** Displays the execution results:

```
<terminated> Assignment26.py [C:\Python34\python.exe]  
Number of students enrolled for python course: 4  
Number of students enrolled for Java course only: 2  
Number of students enrolled for Python course only: 2  
Number of students enrolled for both Python and course: 2  
Number of students enrolled for either Python or Java but not both: 4  
Number of students enrolled for either Python or Java: 6
```

ASSIGNMENT- 27

Ques:

Using functions, re-write and execute Python program to:

- 1.Add natural numbers upto n where n is taken as an input from user.
- 2.Print Fibonacci series till nth term (Take input from user).

The screenshot shows the LiClipse IDE interface. The top menu bar includes File, Edit, Refactoring, Source, Navigate, Search, Project, Pydev, Run, Window, Help, and Quick Access. The left sidebar contains a PyDev Package Explorer with various Python files and projects like A30-c, A41, A42, string, A27, *A27_b, and A44. The main editor window displays the following Python code:

```
# Python program to display the Fibonacci sequence up to n-th term using recursive functions
def recur_fibo(n):
    """Recursive function to
    print Fibonacci sequence"""
    if n <= 1:
        return n
    else:
        return(recur_fibo(n-1) + recur_fibo(n-2))
# Change this value for a different result
nterms = 10
# uncomment to take input from the user
#nterms = int(input("How many terms? "))
# check if the number of terms is valid
if nterms <= 0:
    print("Please enter a positive integer")
else:
    print("Fibonacci sequence:")
    for i in range(nterms):
        print(recur_fibo(i))
```

The bottom right corner of the IDE shows the system tray with icons for battery, signal, volume, and date/time (6:57 PM, 6/21/2017). The bottom taskbar has icons for Start, Internet Explorer, File Explorer, Google Chrome, and LiClipse.

The Console tab shows the output of the executed code:

```
<terminated> A27_b.py [C:\Python34\python.exe]
Fibonacci sequence:
0
1
1
2
3
5
8
13
21
34
```

ASSIGNMENT -28

At an airport, a traveler is allowed entry into the flight only if he clears the following checks:

- 1.Baggage Check
- 2.Immigration Check
- 3.Security Check

The logic for the check methods are given below:

check_baggage (baggage_weight)

- .returns True if

baggage_weight

is greater than or equal to 0 and less than or equal to 40. Otherwise returns False.

check_immigration (expiry_year)

- .returns True if

expiry_year

is greater than or equal to 2001 and less than or equal to 2025. Otherwise returns False.

check_security(noc_status)

- .returns True if

noc_status

is 'valid' or 'VALID', for all other values return False.

traveler()

- .Initialize the traveler Id and traveler name and invoke the functions check_baggage(), check_immigration() and check_security() by passing required arguments.
- .Refer the table below for values of arguments.

• If all values of check_baggage(), check_immigration() and check_security() are true,

display traveler_id and traveler_name

display "Allow Traveler to *fly*!"

Otherwise,

display traveler_id and traveler_name

display "Detain Traveler for Re-checking!"

Invoke the traveler() function. Modify the values of different variables in traveler() function and observe the output.

```
def check_baggage(baggage_weight):
    if(baggage_weight >= 0 and baggage_weight <= 40):
        return True
    return False
```

```
def check_immigration(expiry_year):
    if(expiry_year >= 2001 and expiry_year <= 2025):
        return True
    return False
```

```
def check_security(noc_status):
    if(noc_status == "valid" or noc_status == "VALID"):
        return True
    return False
```

```
def traveler():
    traveler_id = 1001
    traveler_name = "John"
```

NAME - KANWALJIT SINGH (62)
SAP ID – 500044606, Roll - 62
CCVT – 6th SEM

Submitted To: Mr. Deepak Kumar Sharma
SIGNATURE/REMARKS

```
baggage_weight = 35
expiry_year = 2019
noc_status = "VALID"
baggage_status = check_baggage(baggage_weight)
immigration_status = check_immigration(expiry_year)
security_status = check_security(noc_status)
print("Traveler ID: ", traveler_id)
print("Traveler name: ", traveler_name)
if(baggage_status and immigration_status and security_status):
    print("Allow Traveler to fly")
    return
print("Display Traveler for Rechecking!")
return

traveler()
```

```
Traveler ID: 1001
Traveler name: John
Allow Traveler to fly
```

ASSIGNMENT-29

Ques: Consider the pseudo code for generating Fibonacci series using Recursion:

FIBO (number)

```
if (number = 0) then
    return (0)
else if (number = 1) then
    return (1)
else
    return FIBO(number - 1) + FIBO(number - 2)
end if
```

Write a program in Python to implement the same using Recursion and execute it in Eclipse.
Print appropriate error message if the user enters negative number as input

The screenshot shows the LiClipse IDE interface. The top menu bar includes File, Edit, Refactoring, Source, Navigate, Search, Project, PyDev, Run, Window, Help, and Quick Access. The left sidebar is the PyDev Package Explorer, displaying a project structure with packages like Assignments, Fibonacci, and files such as A18.py, A27.b.py, A27.py, A29.py, A30.py, A30-b.py, A30-c.py, A40.py, A40-py, A41.py, A42.py, A44.py, and A44.py. The main workspace shows a Python script named A29.py with the following code:

```
1 def recur_fibo(n):
2     if n <= 1:
3         return n
4     else:
5         return(recur_fibo(n-1) + recur_fibo(n-2))
6 # take input from the user
7 nterms = int(input("How many terms? "))
8 # check if the number of terms is valid
9 if nterms <= 0:
10     print("Please enter a positive integer")
11 else:
12     print("Fibonacci sequence:")
13     for i in range(nterms):
14         print(recur_fibo(i))
```

The bottom console window shows the output of running the script:

```
<terminated> A29.py [C:\Python34\python.exe]
How many terms? 6
Fibonacci sequence:
0
1
1
2
3
5
```

The system tray at the bottom right shows the date and time as 7:15 PM 6/21/2017.

ASSIGNMENT-30

Ques: Write a Python program to implement the following (Use Recursion):

1. Print 'n' multiples of 3, where 'n' is taken as an input from the user. The multiples should be printed from to last.
2. Reverse a string. Print the original and reversed string.
3. Check if the given string is palindrome. If yes, print "String is palindrome" otherwise print "String is not palindrome"

The screenshot shows the LiClipse IDE interface. The left pane displays a file tree for a project named 'Assignment'. Inside 'Assignment', there are several subfolders like 'eclipse', 'Fibonacci', 'py', 'Foundation Program 5', 'pythonpractice', 'A18.py', 'A30.py', 'ASSIGNMENT 4.docx', 'Even.py', 'palindrome-16.py', 'prime.py', 'string.py', 'Sum.py', and 'C:\Python34\python.exe'. Below the file tree, the 'PyDev Pac' view is open, showing various Python packages. The main workspace shows a code editor with Python code for printing multiples of 3. The code is as follows:

```
3 # define a function
4 def print_factors(n):
5     """ This function takes a number and prints the factors
6     of that number
7     """
8     print("The multiples of 3 are:")
9     for i in range(1, n + 1):
10         print(3*i)
11
12     # change the value for a different result.
13
14 # uncomment the following line to take input from the user
15 num = int(input("Enter a number: "))
16
17 print_factors(num)
```

Below the code editor is a 'Console' view. It shows the command `<terminated> A30.py [C:\Python34\python.exe]` followed by the output of the program. The user enters '10' and the program prints the multiples of 3 from 3 to 30.

```
Enter a number: 10
The multiples of 3 are:
3
6
9
12
15
18
21
24
27
30
```

LiClipse Workspace - Assignments/A30-b.py - LiClipse (UNREGISTERED)

```
1 def reverse(string):
2     if len(string) == 0:
3         ...
4     else:
5         return reverse(string[1:]) + string[0]
6     a = str(input("Enter the string to be reversed: "))
7     print(a)
8     print(reverse(a))
```

Console <terminated> A30-b.py [C:\Python34\python.exe]
<terminated> A30-b.py [C:\Python34\python.exe]
Enter the string to be reversed: namita
namita
atiman

Writable Insert 8:18 4:29 PM 6/21/2017

LiClipse Workspace - Assignments/A30-c.py - LiClipse (UNREGISTERED)

```
18# Display whether or not a string is a palindrome.
19# @param string the string to check
20# @return True if the string is a palindrome, False otherwise
21#
22def isPalindrome(string):
23    if len(string) < 1:
24        return True
25    if string[0] == string[len(string) - 1]:
26        return isPalindrome(string[1:len(string) - 1])
27    else:
28        return False
29#
30# Call the main function.
31main()
```

Console <terminated> A30-c.py [C:\Python34\python.exe]
<terminated> A30-c.py [C:\Python34\python.exe]
Enter a string: raar
That's a palindrome.

Writable Insert 4:31 PM 6/21/2017

ASSIGNMENT-31

Ques: Write a Python program to:

- 1.read a file
- 2.add backslash (\) before every double quote in the filee contents.
- 3.write it to another file in the same folder.
4. print the content of both the files

```
file1 = open("TestFile1.txt", "r")
print("File opened successfully")
content = file1.read()
print("File read successfully")
file1.close()
print("File closed successfully")
print()
content_to_write = ""
for ch in content:
    if(ch==""):
        content_to_write = content_to_write + '/'
    else:
        content_to_write = content_to_write + ch
print()
file2 = open("TestFile2.txt", "w+")
print("File opened successfully")
file2.write(content_to_write)
print("File written successfully")
file2.close()
```

```
print("File closed successfully")
```

ASSIGNMENT -32

Ques : Consider a file 'courses.txt' in D Drive with the following details:

Write a program to read the file and store the courses in Python variables as a:

1)

Dictionary (Sample - {0: 'Java', 1: 'Python', 2:'Javascript' 3: 'PHP'})

2)

List (Sample -

['Java', 'Python', 'Javascript', 'PHP'])

The screenshot shows the LiClipse IDE interface. The left pane displays a PyDev Package Explorer with several Python files in the 'pythonpractice' package. The right pane shows the code editor for 'A32.py' and the 'Console' output. The code reads 'courses.txt' and prints a dictionary mapping course names to scores.

```
1 # creating a dictionary from a text file
2 # key value pairs occupy a line and are separated by a space
3 # data for the test file of name_course_score pairs
4 text = """
5 Java 0
6 Python 1
7 Javascript 2
8 PHP 3
9
10 fname = "courses.txt"
11 # write the test file
12 fout = open(fname, "w")
13 fout.write(text)
14 fout.close()
15 # read the test file in and convert to a dictionary
16 courses_dict = {}
17 for line in open(fname):
18     name, score = line.split()
19     courses_dict[name] = int(score)
20 print(courses_dict)
```

<terminated> A32.py [C:\Python34\python.exe]
({'Java': 0, 'Python': 1, 'Javascript': 2, 'php': 3})

2)

The screenshot shows the LiClipse IDE interface. The top menu bar includes File, Edit, Refactoring, Source, Navigate, Search, Project, PyDev, Run, Window, Help, and Quick Access. The left sidebar is the PyDev Package Explorer, displaying a project structure with files like A18.py, A27_b.py, A27.py, A29.py, A30.py, A30-b.py, A30-c.py, A32.py, A32-b.py, A-40, A41.py, A42.py, and A44.py. The central workspace contains two tabs: A32 and courses.txt. The A32 tab displays the following Python code:

```
1 # text for the test
2 test_text = """
3 Java
4 Python
5 Javascript
6 PHP
7 fname = "courses.txt"
8 #write the multilng text to a test file
9 fout = open(fname, "w")
10 fout.write(test_text)
11 fout.close()
12 # read the file back as a list of lines
13 fin = open(fname, "r")
14 data_list = fin.readlines()
15 fin.close()
16
17 # optionally strip the trailing newline char '\n'
18 data_list = [item.rstrip('\n') for item in data_list]
19 print (data_list)
20
21
22
```

The bottom tab, courses.txt, shows the output of the script: ['Java', 'Python', 'Javascript', 'PHP']. The status bar at the bottom right indicates the time as 12:08 PM and the date as 6/23/2017.

ASSIGNMENT-33

Ques: Consider a file 'student_details.txt' in D Drive with the details of students in ABC institute – student id and name:

Write a program to read the file and store the student records in Python variable as:

1) List of lists

2) List of dictionaries

The screenshot shows the LiClipse IDE interface. The PyDev Package Explorer view on the left displays several Python files, including A32.py, A32-b.py, A33-a.py, and Assignment11.py. The A33-a.py file is open in the editor, containing the following code:

```
1 # creating a dictionary from a text file
2 # key value pairs occupy a line and are separated by a space
3 # data for the test file of name course_score pairs
4 text = """
5 Rahul 101
6 julie 102
7 helena 103
8 kelly 104"""
9 fname = "courses.txt"
10 frite = open(fname, "w")
11 fout.write(text)
12 fout.close()
# read the test file in and convert to a dictionary
13 courses_dict = {}
14 for line in open(fname):
15     name, score = line.split()
16     courses_dict[name] = int(score)
17
18 print(courses_dict)
```

The Console view at the bottom shows the output of running the script:

```
<terminated> A33-a.py [C:\Python34\python.exe]
{'Rahul': 101, 'kelly': 104, 'julie': 102, 'helena': 103}
```

2)

The screenshot shows the LiClipse IDE interface. The left pane displays a PyDev Package Explorer with various Python files in the 'Assignments' project. The central pane shows the code for 'A33.b.py'. The code reads a file named 'courses.txt' containing names and scores, and writes the contents to another file. The right pane shows the 'Console' output, which displays the names and scores from the file. The bottom status bar shows the date and time.

```
1 # text for the test
2 test_text = """
3 Rahul 101
4 Julie 102
5 helena 103
6 kelly 104"""
7 fname = "courses.txt"
8 #write the multilining text to a test file
9 fout = open(fname, "w")
10 fout.write(test_text)
11 fout.close()
12
13 # read the file back as a list of lines
14 fin = open(fname, "r")
15 data_list = fin.readlines()
16 fin.close()
17 # optionally strip the trailing newline char '\n'
18 data_list = [item.rstrip('\n') for item in data_list]
19 print(data_list)
20
21
22
```

Console > A33.b.py [C:\Python34\python.exe]
['Rahul 101', 'Julie 102', 'helena 103', 'kelly 104']

ASSIGNMENT-34

Ques: Write a Python program to count the words in the file using a dictionary (use space as a delimiter). Find unique

words and the count of their occurrences(ignoring case). Write the output in another file "words.txt" at the same location.

```
file = open("D://rhyme.txt", "r")
content = file.read().lower()
file.close()
```

```
unique_words = set(content.split())
```

```
words_count = {}
```

```
for word in unique_words:
```

```
    count = 0
```

```
    for word1 in content.split():
```

```
        if(word1 == word):
```

```
            count = count + 1
```

```
    words_count[word.capitalize()] = count
```

```
file = open("D://words.txt", "w+")
```

```
file.write("Number of words: %d\n\n" % (len(content.split())))
```

```
file.write("Unique Words\t\tOccurrences\n\n")
```

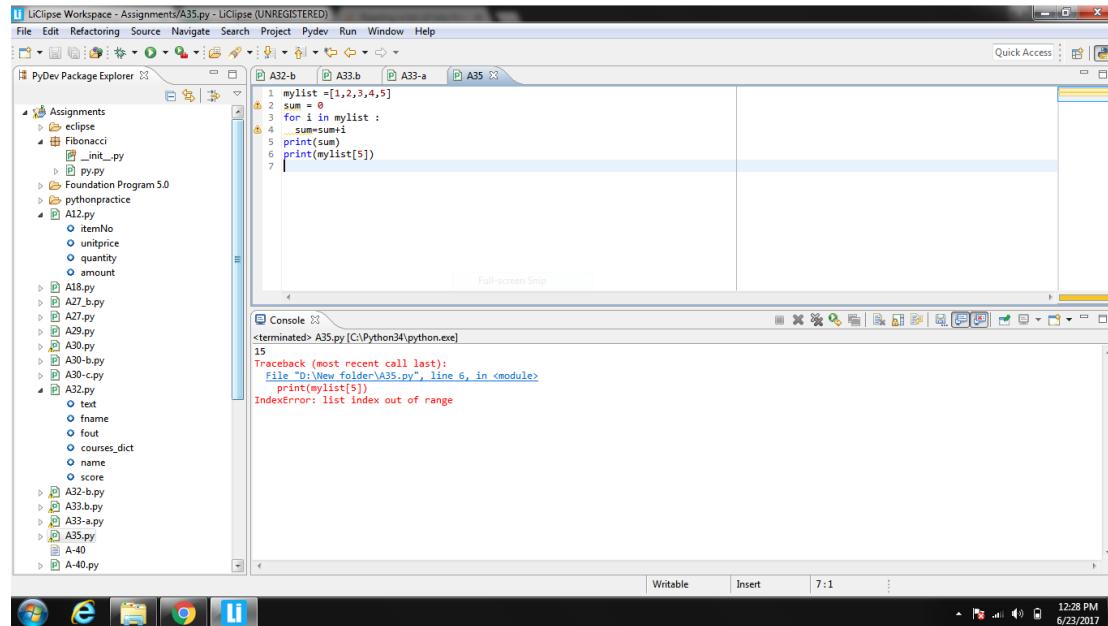
```
for word, count in words_count.items():
```

```
    file.write("%s\t\t%d\n" % (word, count))
```

```
file.close()
```

ASSIGNMENT-35

Ques: Rewrite the code to handle the exceptions raised. Print appropriate error messages wherever applicable



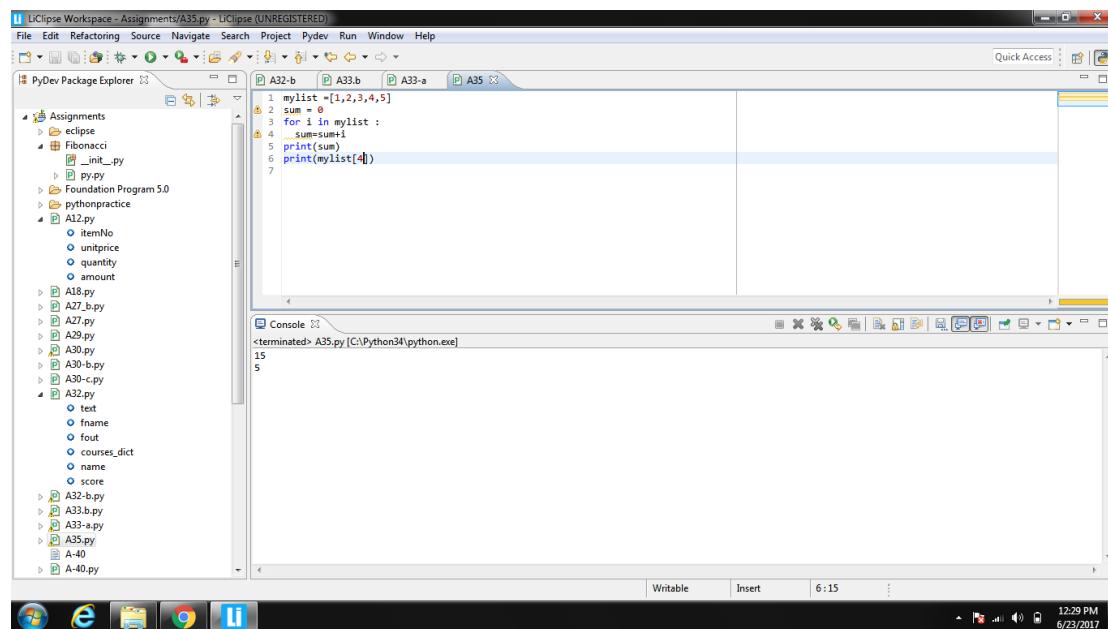
The screenshot shows the LiClipse IDE interface. The PyDev Package Explorer on the left lists various Python files under the 'Assignments' project. The A35.py file is open in the editor, containing the following code:

```
1 mylist =[1,2,3,4,5]
2 sum = 0
3 for i in mylist :
4     sum+=i
5     print(sum)
6 print(mylist[5])
7
```

The console window at the bottom shows the output of running the script:

```
<terminated> A35.py [C:\Python34\python.exe]
15
Traceback (most recent call last):
  File "D:\New folder\A35.py", line 6, in <module>
    print(mylist[5])
IndexError: list index out of range
```

The status bar at the bottom right indicates the time as 12:28 PM and the date as 6/23/2017.



The screenshot shows the same LiClipse IDE interface after the code has been modified. The A35.py file now contains:

```
1 mylist =[1,2,3,4,5]
2 sum = 0
3 for i in mylist :
4     sum+=i
5     print(sum)
6 print(mylist[4])
7
```

The console window shows the output:

```
<terminated> A35.py [C:\Python34\python.exe]
15
5
```

The status bar at the bottom right indicates the time as 12:29 PM and the date as 6/23/2017.

ASSIGNMENT-36

Ques: You have already created a Python program to implement the following in file handling section:

- 1.read a file.
- 2.add backslash (\) before every double quote in the file contents.
- 3.write it to another file in the same folder.
- 4.print the contents of both the files.

try:

```
file1 = open("TestFile1.txt", "r")
content = file1.read()
file1.close()
content_to_write = ""
for ch in content:
    if(ch=="'"):
        content_to_write = content_to_write + '/'"
    else:
        content_to_write = content_to_write + ch
print()
file2 = open("TestFile2.txt", "w+")
file2.write(content_to_write)
file2.close()
except IOError:
    print("Particular file could not be opened or read or written")
```

```
else :  
    print("Operation performed successfully")
```

ASSIGNMENT 37

Ques You have already executed the Python program given below in Functions section:

- Add natural numbers up to n where n is taken as an input from user.

Do appropriate exception handling in the code and observe the output by providing invalid input values.

The screenshot shows the LiClipse IDE interface. The left pane displays the PyDev Package Explorer with a project structure under 'Assignments'. The right pane shows the code editor for file A37.py, which contains the following Python code:

```
1 # Python Program - Find Sum of Natural Numbers
2
3 while True:
4     print("Enter '0' for exit.")
5     num = int(input("Upto which number ? "))
6     if num == 0:
7         break
8     elif num < 1:
9         print("Please, enter a positive number...")
10    else:
11        sum = 0
12        while num > 0:
13            sum += num
14            num -= 1
15        print("Sum = ", sum)
```

Below the code editor is the 'Console' view, which shows the execution of the program. It prompts the user to enter a number, receives the input '4', and then prints the sum as 10. It also handles an invalid input '-9' by prompting for a positive number.

ASSIGNMENT- 38

Ques: Refer to the following assignment which you have already executed in Functions section. Modify your code to implement Exception Handling and display appropriate error message wherever applicable. At an airport, a traveler is allowed entry into the flight only if he clears the following checks:

1. Baggage Check

2. Immigration Check

3. Security Check

The logic for the check methods are given below:

check_baggage (baggage_weight)

• returns True if

baggage_weight

is greater than or equal to 0 and less than or equal to 40. Otherwise returns

False.

check_immigration (expiry_year)

• returns True if

expiry_year

is greater than or equal to 2001 and less than or equal to 2025. Otherwise returns

False.

check_security(noc_status)

• returns True if

noc_status

is 'valid' or 'VALID', for all other values return False.

traveler()

• Initialize the traveler Id and traveler name and invoke the functions check_baggage(), check_immigration() and

check_security() by passing required arguments.

• Refer the table below for values of arguments.

• If all values of check_baggage(), check_immigration() and check_security() are true,

display traveler_id and traveler_name

display "Allow Traveler to *fly*!"

Otherwise,

display traveler_id and traveler_name

display "Detain Traveler for Re-checking!"

Invoke the traveler() function. Modify the values of different variables in traveler() function and observe the output.

```
def check_baggage(baggage_weight):  
    if(baggage_weight >= 0 and baggage_weight <= 40):  
        return True  
    return False
```

```
def check_immigration(expiry_year):  
    if(expiry_year >= 2001 and expiry_year <= 2025):  
        return True  
    return False
```

```
def check_security(noc_status):  
    if(noc_status == "valid" or noc_status == "VALID"):  
        return True  
    return False
```

```
def traveler():

    traveler_id = 1008

    traveler_name = "John"

    baggage_weight = "forty"

    expiry_year = 2022

    noc_status = "NOT VALID"

    try:

        baggage_status = check_baggage(baggage_weight)

        immigration_status = check_immigration(expiry_year)

        security_status = check_security(noc_status)

    except TypeError:

        print("Baggage weight and expiry year must be natural numbers")

    else:

        print("Traveler ID: ", traveler_id)

        print("Traveler name: ", traveler_name)

        if(baggage_status and immigration_status and security_status):

            ("Allow Traveler to fly")

            return

        print("Display Traveler for Rechecking!")

        return

traveler()
```

```
Baggage weight and expiry year must be natural numbers
>>> |
```

ASSIGNMENT-39

Ques: Create a module "number_checker.py" which has following 2 functions:

- `is_prime(num)` : this function returns true if the input number is prime
- `is_even(num)`: this function returns true if the input number is even
- Create another Python module "test_module.py".
- Invoke the functions "is_prime(num)" and "is_even(num)" in "test_module.py".
- Observe the results.

Hint: Import

"number_checker.py" module in "test_module.py" before using it's functions

```
def is_prime(num):  
    for i in range(2,num):  
        if(num % i == 0):  
            return False  
    return True
```

```
def is_even(num):  
    return (num % 2 == 0)
```

```
import number_checker  
  
num = int(input("Enter a number:- "))  
  
if(number_checker.is_prime(num)):  
    print("%d is a prime number" % num)  
  
else:  
    print("%d is not a prime number" % num)  
  
if(number_checker.is_even(num)):
```

```
print("%d is a even number" % num)
else:
    print("%d is not a even number" % num)

Enter a number:- 2
2 is a prime number
2 is a even number
>>> |
```

ASSIGNMENT- 40

Write a Python program to randomly print any of the below numbers:

100,200,300,400,500,600,700,800,900,1000

Execute the program 10 times and verify if the number generated in every output is one out of the numbers given in the list above.

```
import random

#Print any of 100,200,300,400,500,600,700,800,900,1000
x = random.randrange(100,1000,100)
print(x)

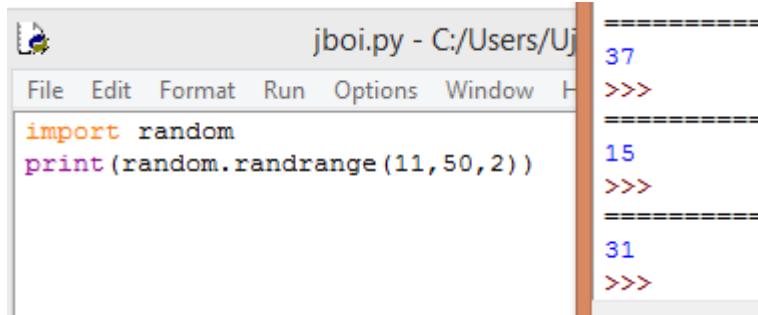
print()

#Print odd number between 10 & 50
x = random.randrange(11,50,2)
print(x)

800

31
>>> |
•
```

Write a Python program to print a random odd numbers between 10 and 50.



```
jboi.py - C:/Users/Uj
File Edit Format Run Options Window Help
=====
37
>>>
=====
15
>>>
=====
31
>>>
```

ASSIGNMENT-41

Write a Python program for rolling a dice on clicking enter key. The program should run infinitely until user enters 'q'.

```
import random

ch = input("Press any key (+ Enter) to roll a dice (q to quit): ")
while(ch.lower() != 'q'):
    print("Dice rolled: ", random.randint(1,6))
    print()
    ch = input("Press any key (+ Enter) to roll a dice again (q to quit): ")

Press any key (+ Enter) to roll a dice (q to quit):
Dice rolled:  1

Press any key (+ Enter) to roll a dice again (q to quit):
Dice rolled:  4

Press any key (+ Enter) to roll a dice again (q to quit):
Dice rolled:  3

Press any key (+ Enter) to roll a dice again (q to quit):
Dice rolled:  2

Press any key (+ Enter) to roll a dice again (q to quit): q
>>> |
```

ASSIGNMENT-42

If area of one wall of a cubical wooden box is 16 units, write a Python program to display the volume of the box.

Note:

Area of a cube with side 'a' is ' a^{**2} '.

Volume of the cube can be computed as ' a^{**3} '.

Hint: Make use of 'sqrt' and 'pow' functions from math module.

Duration : 10 mins

```
import math
area=16
side=math.sqrt(area)
volume=math.pow(side,3)
print(volume)
```

64.0

ASSIGNMENT-43

The ABC Institute offers vocational courses to students in multiple areas e.g. theatre, classical singing, traditional dance forms, Bollywood dance, literature and so on. A student can enroll for zero to all courses. Write a Python function that takes the number of courses as an input and returns the total number of different course combinations, a student can opt for. (Make use of functions available in math module

```
import math
num=int(input("Enter number of courses"))
print(math.factorial(num))
```

Enter number of courses4

24

ASSIGNMENT-44

Execute the following code and observe the output.

- 1.
- import time
- 2.
- print(time.time())
- 3.
- print(time.localtime())
- 4.
- print(time.localtime(time.time()))
- 5.
- print(time.asctime())
- 6.
- mytime = (2016,7,27,15,45,23,0,0,0)
- 7.
- print(time.localtime(time.mktime(mytime)))

```
>>> import time
>>> time.time()
1498675762.6218405
>>> time.localtime()
time.struct_time(tm_year=2017, tm_mon=6, tm_mday=29, tm_hour=0, tm_min=19, tm_se
c=36, tm_wday=3, tm_yday=180, tm_isdst=0)
>>> time.localtime(time.time())
time.struct_time(tm_year=2017, tm_mon=6, tm_mday=29, tm_hour=0, tm_min=20, tm_se
c=15, tm_wday=3, tm_yday=180, tm_isdst=0)
>>> time.asctime()
'Thu Jun 29 00:20:32 2017'
>>> mytime=(2017,6,29,0,20,0,3,180,0)
>>> print(time.localtime(time.mktime(mytime)))
time.struct_time(tm_year=2017, tm_mon=6, tm_mday=29, tm_hour=0, tm_min=20, tm_se
c=0, tm_wday=3, tm_yday=180, tm_isdst=0)
>>> |
```

ASSIGNMENT-45

Consider a Python string:

```
cust_details = "Hello John, your customer id is j181"
```

1)

Find, if the name of the customer is preceded by a pattern "Hello " or "hello " (Observe a space after the word)?

If pattern is found, print the searched result.

2)

Find, if the given string ends with a pattern containing only one alphabet followed by three numbers? If pattern

is found, print the searched result.

3)

Replace the word starting with "j" followed by three numbers to only the number(remove the alphabet).

4)

Replace the word "id" with "ID".

The output of the above code is "

Hello John, your customer ID is 181

```
import re
str="Hello John,your customer id is j181"
match= re.search('Hello ',str)
print(match.group())

match= re.search('j\d{3}',str)
print(match.group())

print(re.sub('j(\d{3})',r'\1',str))
print(re.sub('id','ID',str))
```

```
Hello
j181
Hello John,your customer id is 181
Hello John,your customer ID is j181
```

"

ASSIGNMENT-46

Consider a scenario of managing student details in ABC Training Institute. Write a Python program to implement the

business requirements mentioned below:

a)

Accept student_id and validate whether it contains only digits.

b)

If student_id is valid, accept student_name from the user and validate whether it contains only alphabets.

c)

If student_name is valid, accept fees_amount paid by the student:

1.

Decimal point is optional in fees_amount(can have maximum one decimal point)

2.

Only two digits are allowed after decimal point

d)

If invalid data is entered in any of the above steps, display appropriate error messages. Else, create an email_id

for student as

student_name@ABC.com

. Assume there are no duplicate names.

e)

Perform above validations using Regular Expressions and print details of the student: student_id, student_name, fees_amount, email_id

Duration : 20 mins

```
stu_id=input("enter student id")
if(re.search('\d', stu_id)):
    stu_name=input("Enter student name")
    if(re.search('\D', stu_name)):
        fee_amt=input("Enter fee amount")
        if(re.search('\d|\.\d{2}',fee_amt)):
            email=stu_name + "@ABC.com"
        else:
            print("FEE must not have more than two decimal values")
    else:
        print("student name must be alphabets")
else:
    print("student id must be numbers")

print(stu_id,stu_name,fee_amt,email)
```

```
enter student id101
Enter student nameujjawal
Enter fee amount151544.12
101 ujjawal 151544.12 ujjawal@ABC.com
```

ASSIGNMENT-47

Consider a string:

my_string = "Strings are amongst the most popular data types in Python. We can create the strings by enclosing

characters in quotes. Python treats single quotes the same as double quotes."

1) Write a Python program to count the number of occurrences of word "String" in the given string ignoring the case.

2) Write a function "count_words" to print the count of occurrences of a word:

- a) which end with "on". (e.g. Python)
- b) which have "on" in between the first and last characters (e.g. amongst)

Duration : 10 mins

```
my_string = "Strings are amongst the most popular data types in Python . We can create the strir
print(my_string.count("string" or "String"))
t=my_string.split()
c=0
for i in range(0,len(t)):
    if(t[i].endswith("on")):
        c+=1
print("NUmber of word ending with 'on' are ",c)
1
NUmber of word ending with 'on' are  2
```

ASSIGNMENT-48

Consider the price list of various items in the Retail Store:

item_price = [1050, 2200, 8575, 485, 234, 150, 399]

Customer John wants to know the:

1.

Price of costliest item sold in retail store

2.

Number of items in the Retail store

3.

Prices of items in increasing order

4.

Prices of items in descending order

Implement the above mentioned business requirements using built-in List functions.

Hint –

- 1) Use max, len, sort functions of math module.
- 2) For question 3 and 4, display the list in increasing/decreasing order.

```
tem_price = [1050, 2200, 8575, 485, 234, 150, 399]
print(max(tem_price))
print(len(tem_price))
print(sorted(tem_price))
print(sorted(tem_price,reverse = True))

8575
7
[150, 234, 399, 485, 1050, 2200, 8575]
[8575, 2200, 1050, 485, 399, 234, 150]
```

ASSIGNMENT-49

Built-in function in Python which accepts any data structure (list, string, tuple, dictionary and set) and returns a

sorted list.

```
def my_sort(data,order=True):  
    if(order):  
        return sorted(data)  
    else:  
        return sorted(data, reverse=True)  
tem_price = [1050, 2200, 8575, 485, 234, 150, 399]  
print(my_sort(tem_price, False))  
print(my_sort(tem_price))
```

```
[8575, 2200, 1050, 485, 399, 234, 150]  
[150, 234, 399, 485, 1050, 2200, 8575]
```

****END OF MODULE-1****