Assignments for Object Oriented Programming using Python

Author(s)	Sahana Kumaraswamy / Roy Antony Arnold G
Authorized by	Satheesha B.Nanjappa
Creation/Revision Date	Dec 2015
Version	1.0

Education, Training and Assessment Department
Infosys Limited
Electronics City
Hosur Road
Bangalore – 561 229, India.

Tel: 91 80 852 0261-270
Fax: 91 80 852 0362
www.infosys.com
mailto:ETA@infosys.com

Document Revision History

Version	Date	Author(s)	Reviewer(s)	Description
1.0	Dec 2015	Sahana Kumaraswamy / Roy Antony Arnold G	Dr. P. Suresh / Kalpana Balaraman / K.N.Vani	Initial Draft
1.1	June 2017	K.N.Vani	Dr. Sundaresan Krishnan Iyer	Campus Connect FP 4.1 release

CONTENTS

Assignments for Object Oriented Programming using Python

Context

This document contains assignments to be completed as part of the hands on session for the course – Object Oriented Programming using Python

Guidelines

- The assignments guide has been designed to give hands on experience to map/apply the concepts learnt in the theory session with practical assignments.
- The assignments have been categorized into solved assignments to hand hold a beginner, partially solved assignments to begin trying out on their own and unsolved assignments to let learners write code completely on their own
- These assignments contain coding exercises, debugging exercises, coding standards exercises assignments
- The case study based assignments are threaded, which can be built incrementally. This will help understanding the concepts and building a complete application
- The estimated time would help a learner to solve problems given a deadline
- The assignments need to be completed as instructed by the facilitators
- **Assignments marked as Demo can be used by the faculty during lecture and students need to compile and execute it using Eclipse**

Programming Fundamentals in Python - Assignments

Assignment 1: Using Eclipse IDE to create and execute Python Program - Guided Activity

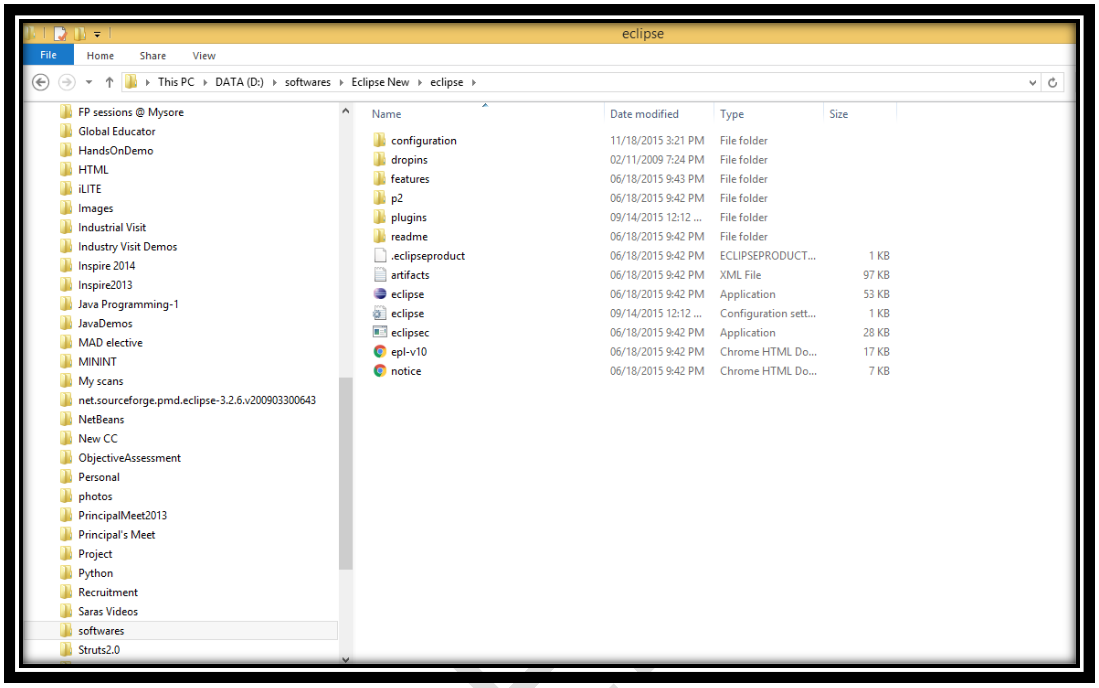
Objective: Learn how to use the Eclipse IDE to create and execute Python programs.

About Eclipse

Eclipse is a multi-language Integrated Development Environment (IDE) and it has an extensible plug-in system. It can be used to develop applications in Python and also other programming languages including C, C++, Ruby, Perl, Python, COBOL, PHP, Java etc. by means of various plug-ins.

Using the Eclipse IDE
Starting Eclipse and Giving Workspace Information

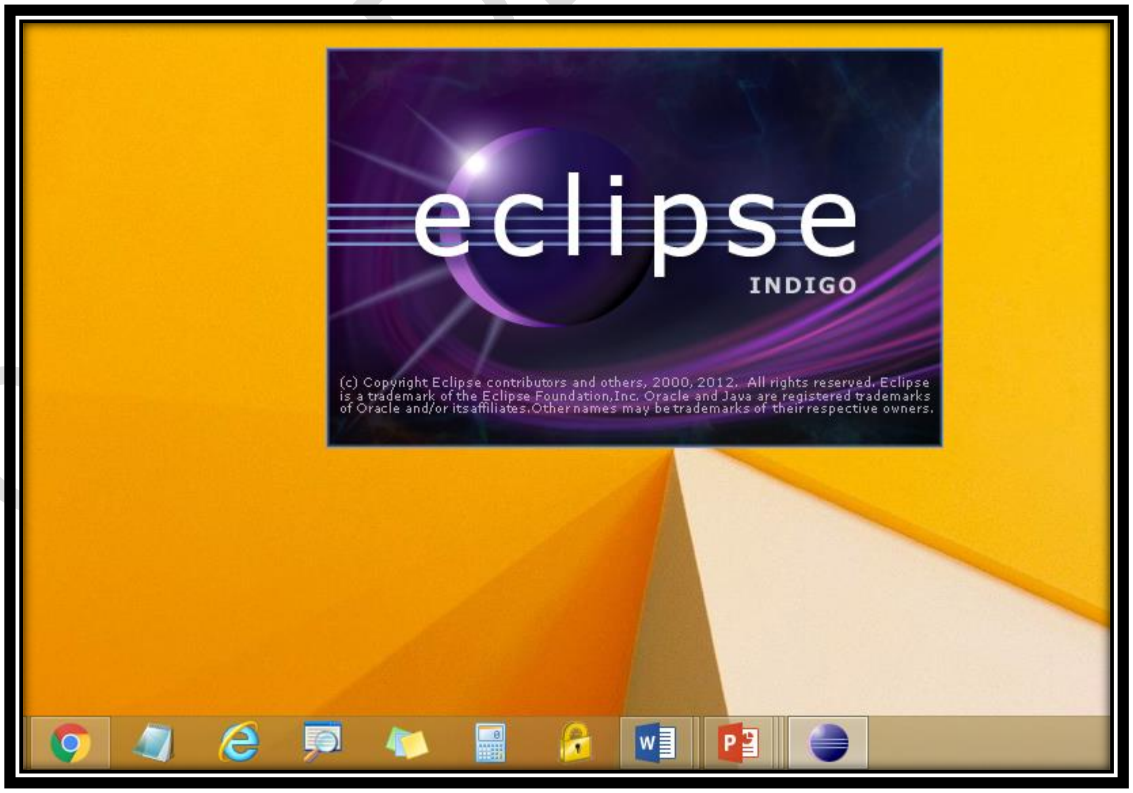
- To start the Eclipse IDE, double click the Eclipse.exe, You may create a shortcut on your desktop



FP sessions @ Mysore
Global Educator
HandsOnDemo
HTML
iLITE
Images
Industrial Visit
Industry Visit Demos
Inspire 2014
Inspire2013
Java Programming-1
JavaDemos
MAD elective
MININT
My scans
net.sourceforge.pmd.eclipse-3.2.6.v200903300643
NetBeans
New CC
ObjectiveAssessment
Personal
photos
PrincipalMeet2013
Principal's Meet
Project
Python
Recruitment
Saras Videos
softwares
Struts2.0

Name	Date modified	Type	Size
configuration	11/18/2015 3:21 PM	File folder	
dropins	02/11/2009 7:24 PM	File folder	
features	06/18/2015 9:43 PM	File folder	
p2	06/18/2015 9:42 PM	File folder	
plugins	09/14/2015 12:12 ...	File folder	
readme	06/18/2015 9:42 PM	File folder	
.eclipseproduct	06/18/2015 9:42 PM	ECLIPSEPRODUCT...	1 KB
artifacts	06/18/2015 9:42 PM	XML File	97 KB
eclipse	06/18/2015 9:42 PM	Application	53 KB
eclipse	09/14/2015 12:12 ...	Configuration sett...	1 KB
eclipsec	06/18/2015 9:42 PM	Application	28 KB
epl-v10	06/18/2015 9:42 PM	Chrome HTML Do...	17 KB
notice	06/18/2015 9:42 PM	Chrome HTML Do...	7 KB

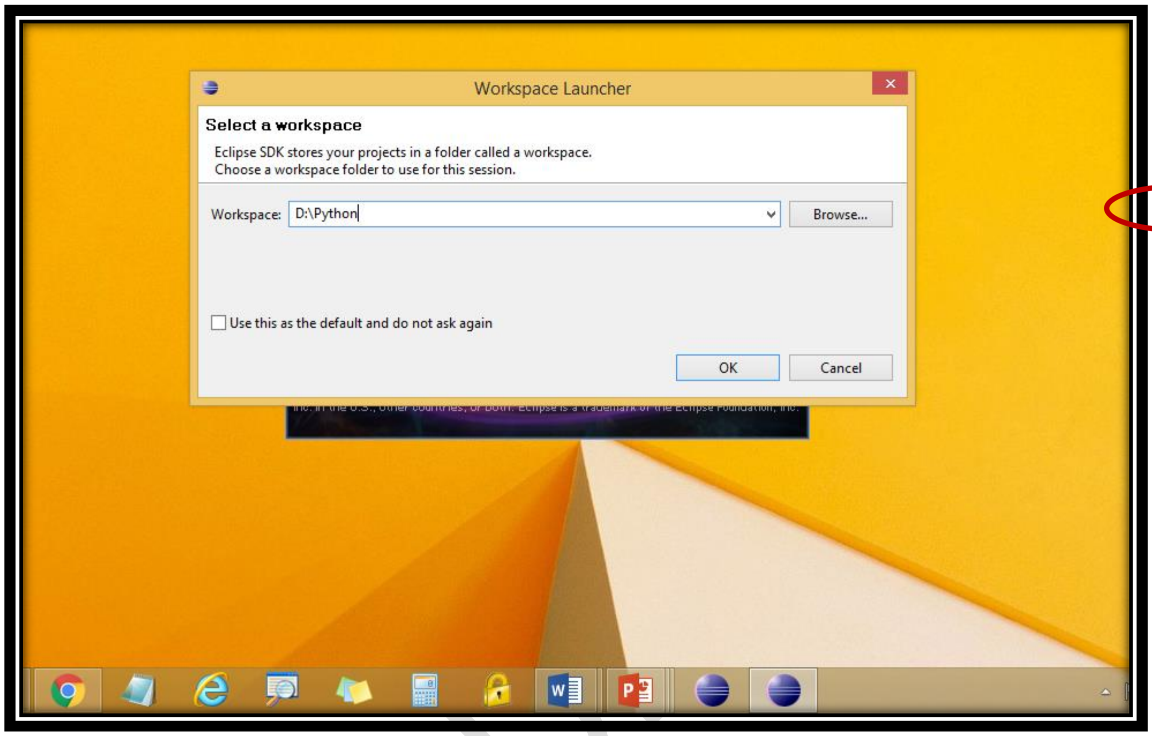
Fig. 1 Starting Eclipse



(c) Copyright Eclipse contributors and others, 2000, 2012. All rights reserved. Eclipse is a trademark of the Eclipse Foundation,Inc. Oracle and Java are registered trademarks of Oracle and/or its affiliates.Other names may be trademarks of their respective owners.

Fig. 2 Eclipse Loading

Once started a prompt will open which will ask you to enter the workspace information as shown in Fig. 3. A workspace is a folder where all the projects created in Eclipse are saved. Create a folder in your desired path and give the correct path as shown below.


Workspace Launcher

Select a workspace

Eclipse SDK stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: D:\Python

Fig. 3 Work Space

Once loaded, you will see the Welcome tab as shown in Fig. 4. Close this Welcome tab to go to the Project Window.

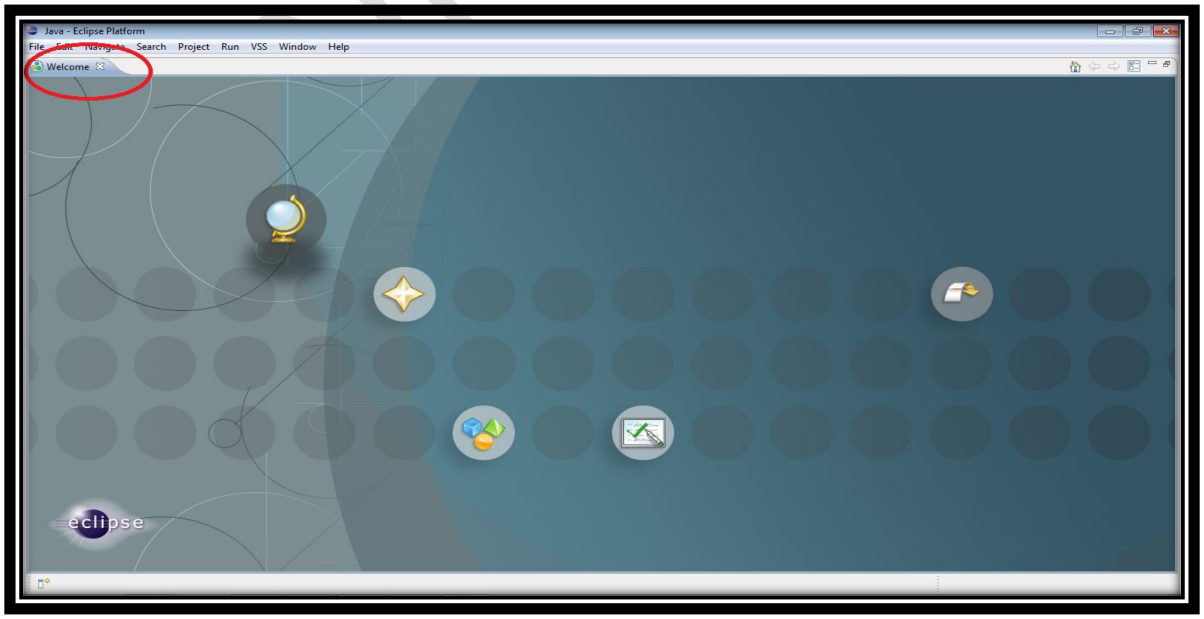


Fig. 4 Welcome Screen

Next go to the Window menu and select the perspective as Python perspective as shown in Fig. 5.1 and Fig 5.2. (Window → Open Perspective → PyDev)

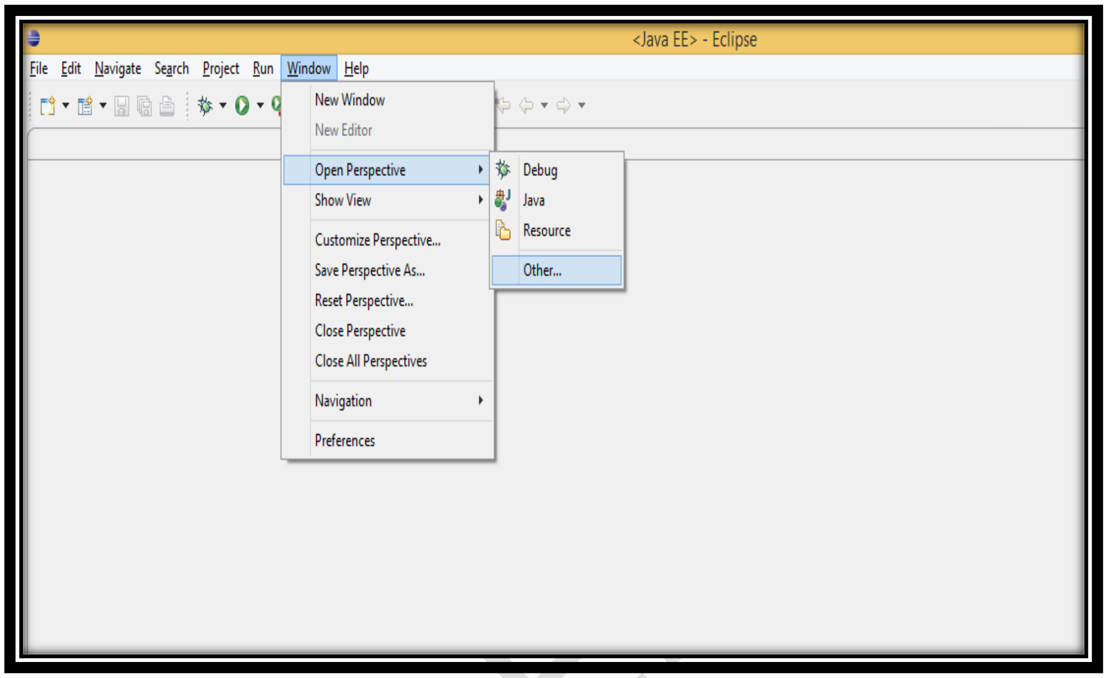


Fig. 5.1 Changing Perspective

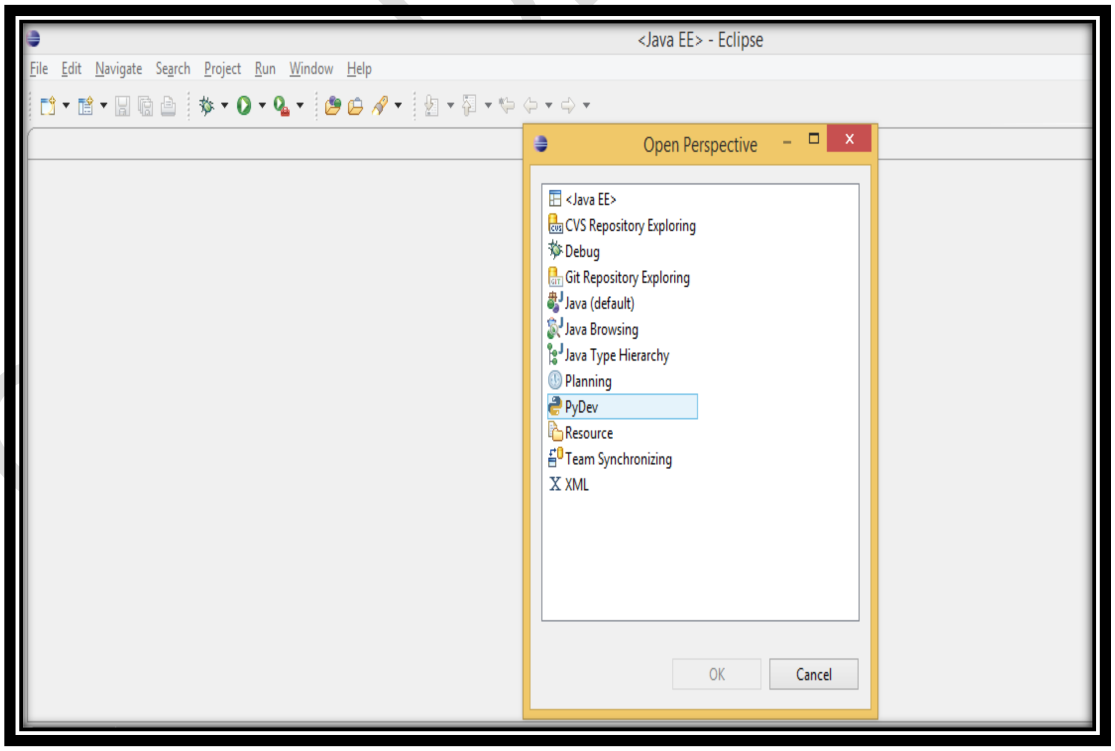


Fig. 5.2 Changing Perspective

Selecting the PyDev perspective will give you the screen as shown in Fig. 6.

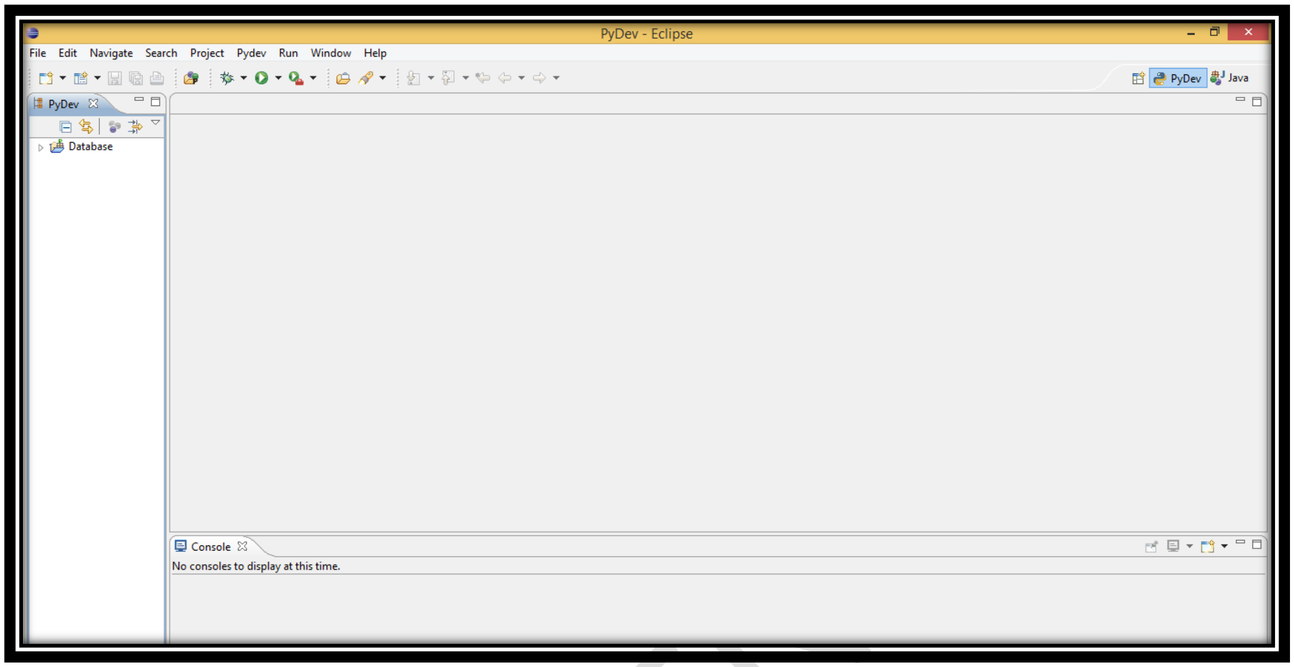

PyDev - Eclipse
File Edit Navigate Search Project Pydev Run Window Help
PyDev
Java
Database
Console
No consoles to display at this time.

Fig. 6 Project Explorer

Creating a New Project

- In Eclipse a workspace can contain one or more projects and each project usually contains a Python application. Each application may contain one or more Python files.
- To create a new project, go to File → New → Project as shown in Fig. 7.

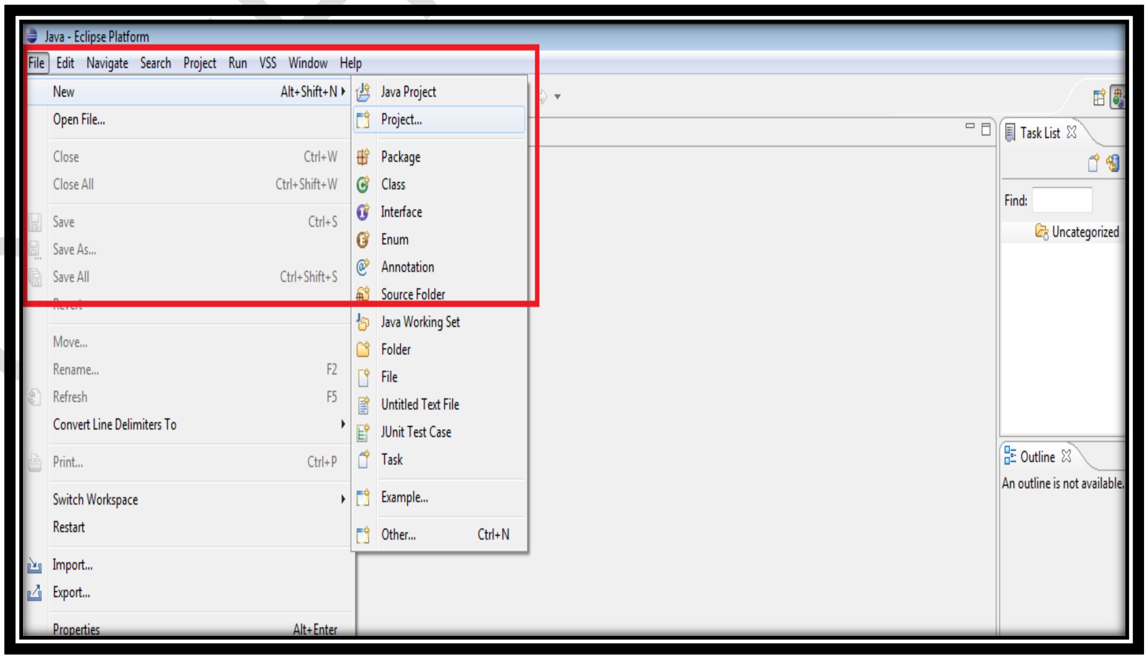


Fig. 7 New Project

- Select the option PyDev Project under the PyDev category as shown in Fig. 8.

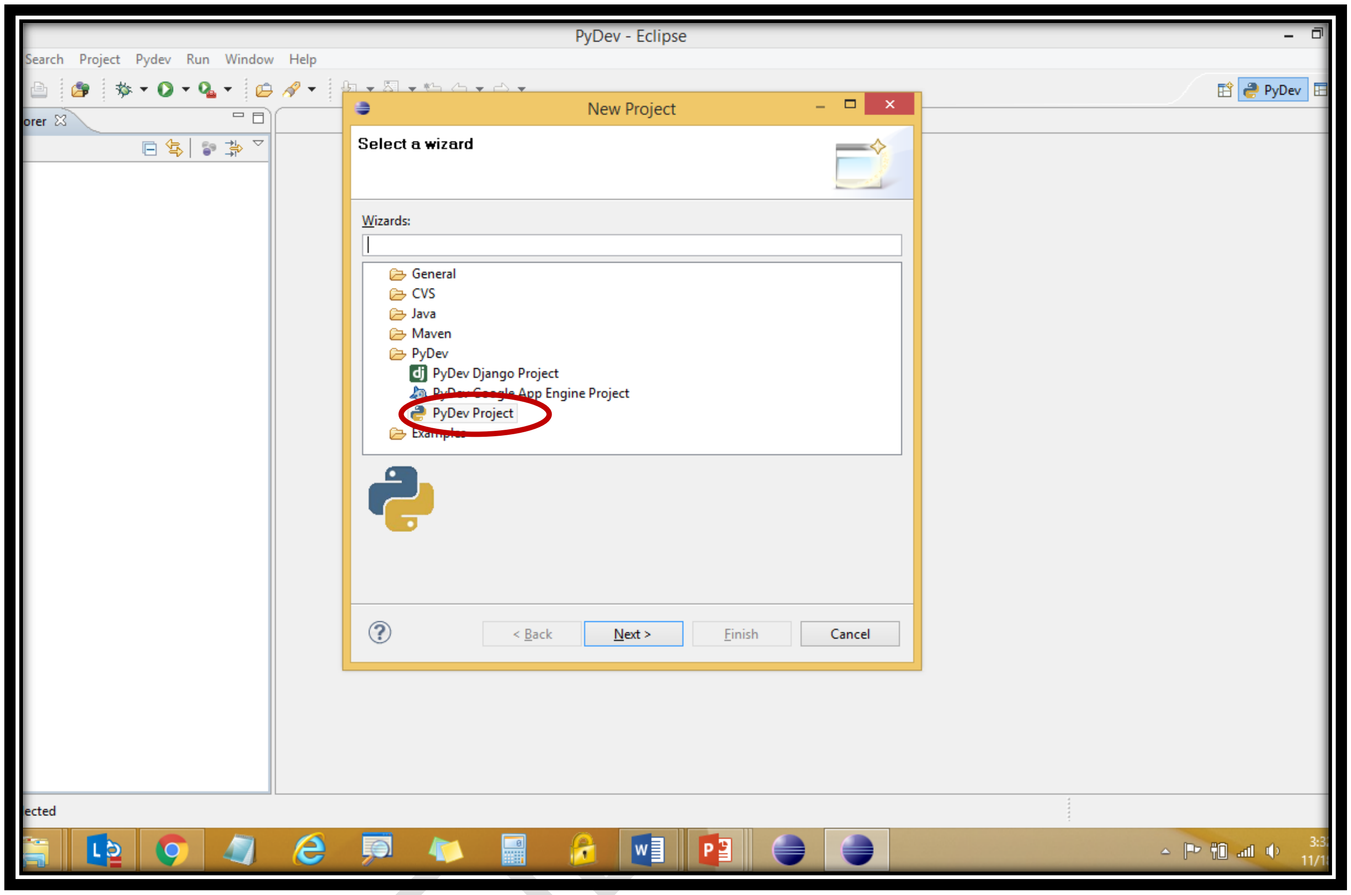
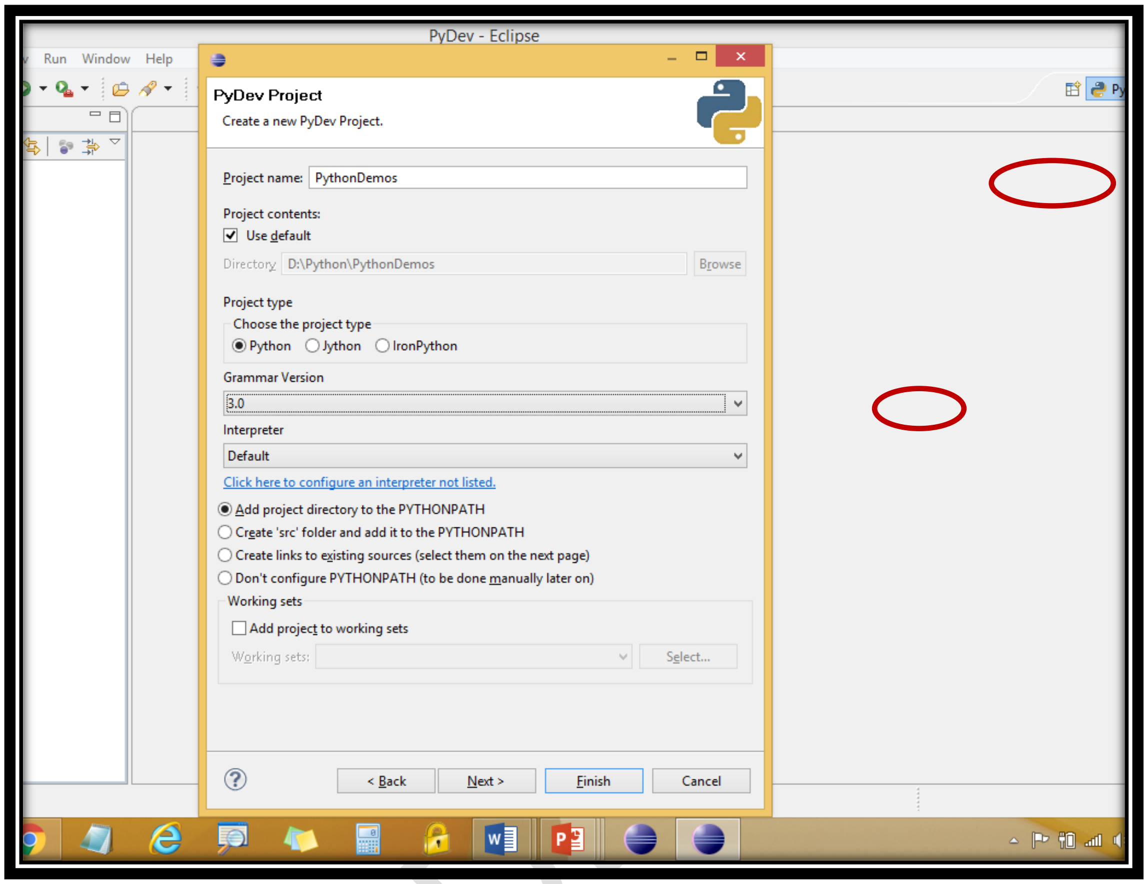


Fig. 8 Project Type – PvDev

- Enter a relevant project name, Set Grammer Version to 3.0
- Select the interpreter from drop down menu if listed or click on "*click here to configure an interpreter not listed"*
- You may configure the interpreter through *quick auto config option* or *manual config* by manually entering the path of the Python interpreter
- After configuring interpreter, click on finish button as shown in Fig. 9.



Project name: PythonDemos

Project contents:
☑ Use default

Directory: D:\Python\PythonDemos Browse

Project type
Choose the project type
● Python ○ Jython ○ IronPython

Grammar Version
3.0

Interpreter
Default

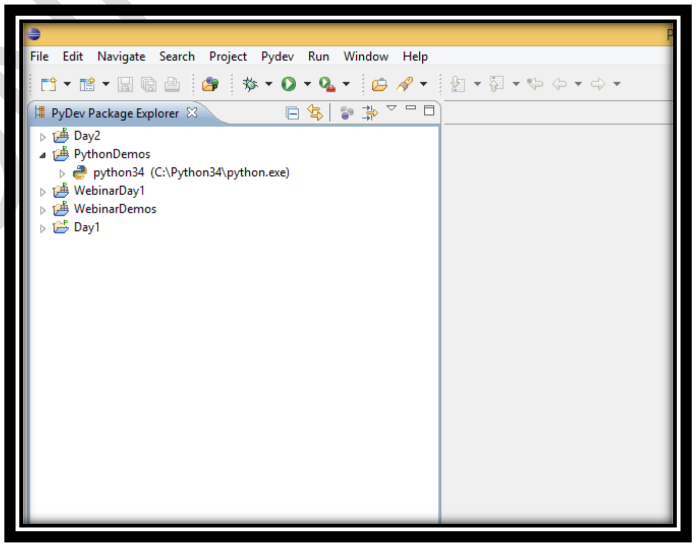
Click here to configure an interpreter not listed.

● Add project directory to the PYTHONPATH
○ Create 'src' folder and add it to the PYTHONPATH
○ Create links to existing sources (select them on the next page)
○ Don't configure PYTHONPATH (to be done manually later on)

Working sets
☐ Add project to working sets
Working sets: Select...

Fig. 9 Project Details

- This will create a project and you can see the same in the package explorer along with the path of the interpreter configured in the last step as shown in Fig. 10.



```
▷ Day2
◢ PythonDemos
    ▷ python34 (C:\Python34\python.exe)
▷ WebinarDay1
▷ WebinarDemos
▷ Day1
```

Fig. 10 Project Directory Structure

Writing Your First Python Program

- To write your first Python program you must first create a PyDev Module. Right click on the project and go to New → PyDev MOdule as shown in Fig. 11.

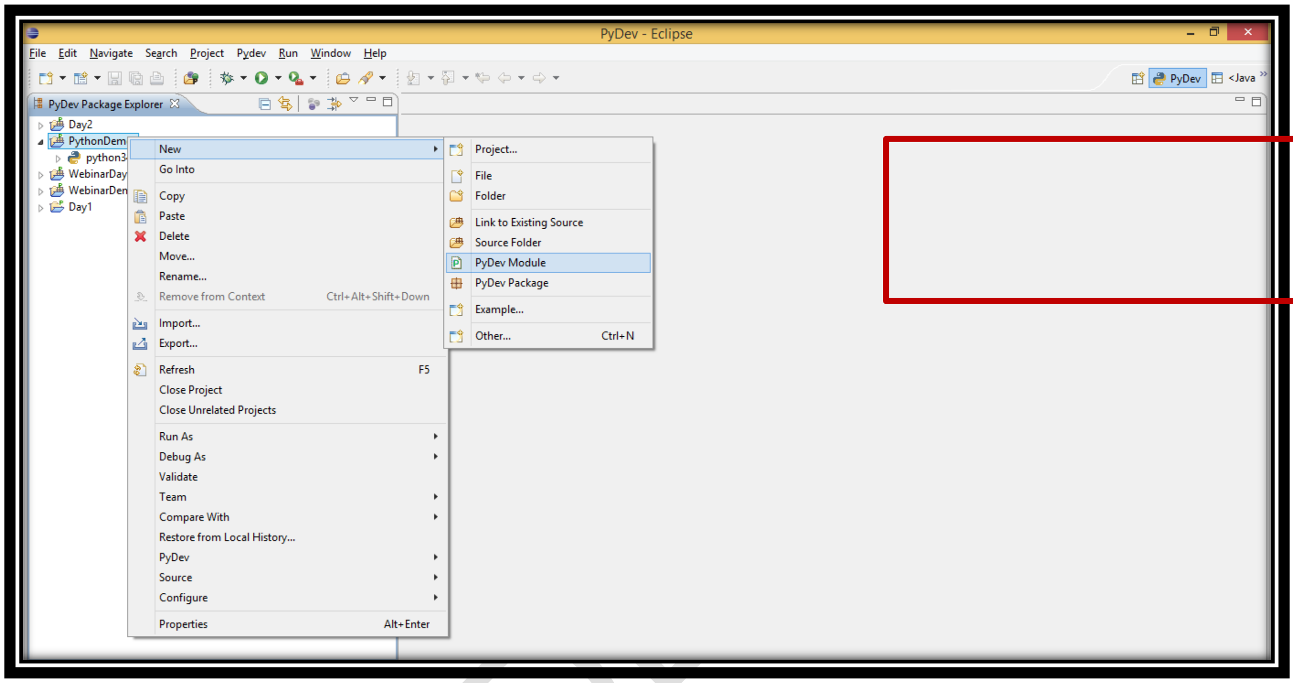
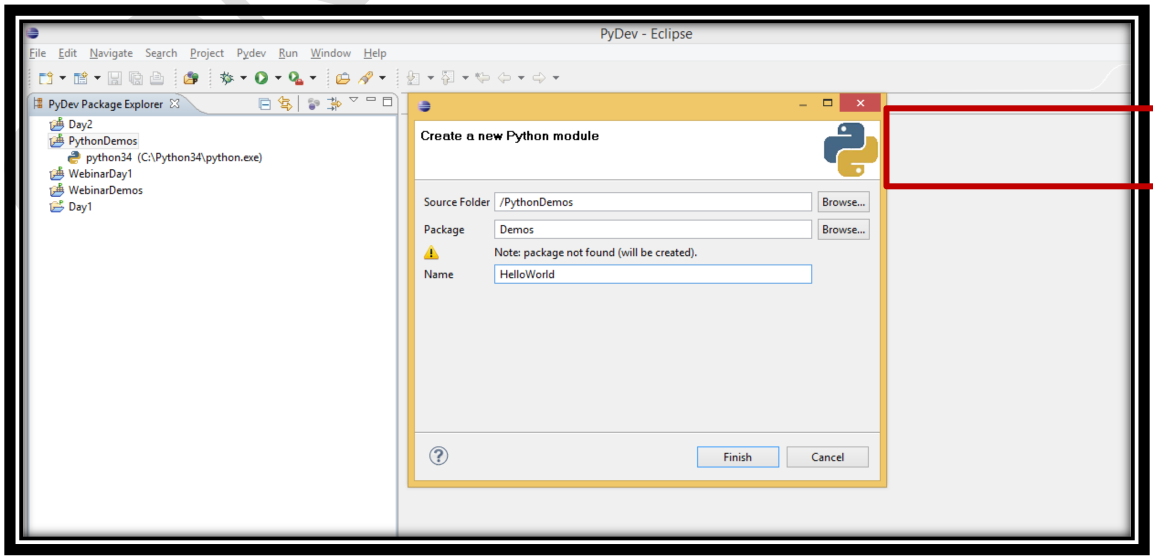


Fig. 11 New PyDev Module

- Enter the Package name and PyDev Module name as shown in Fig. 12 and then click on finish.
  Packages are like directories and PyDev modules are like files. Packages will be explained later.



Create a new Python module

Source Folder /PythonDemos

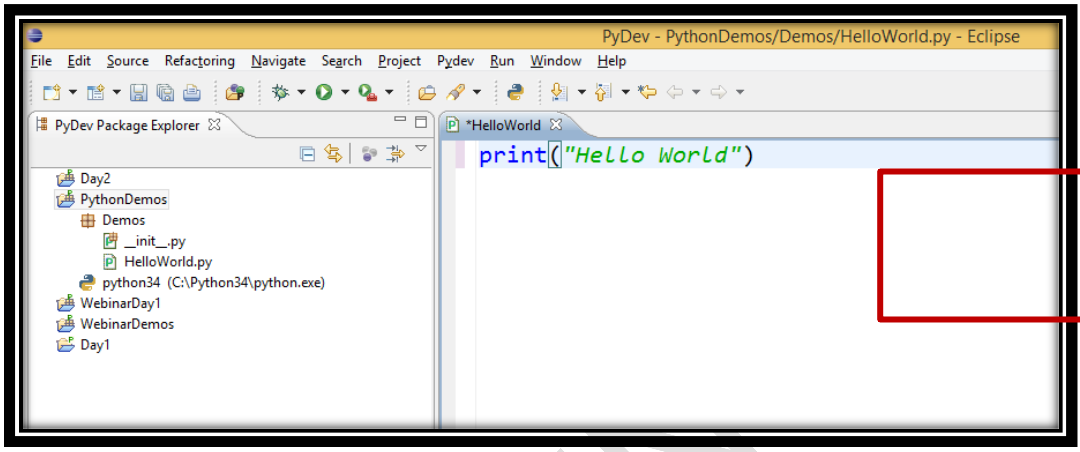
Package Demos

Note: package not found (will be created).

Name HelloWorld

Fig. 12 Enter Package and Module Details

- This will add a Package and a PyDev module in the package to the project as shown in Fig. 13.
- You can now write your Python program as shown in Fig.13
- You can also create new PyDev modules in the package created

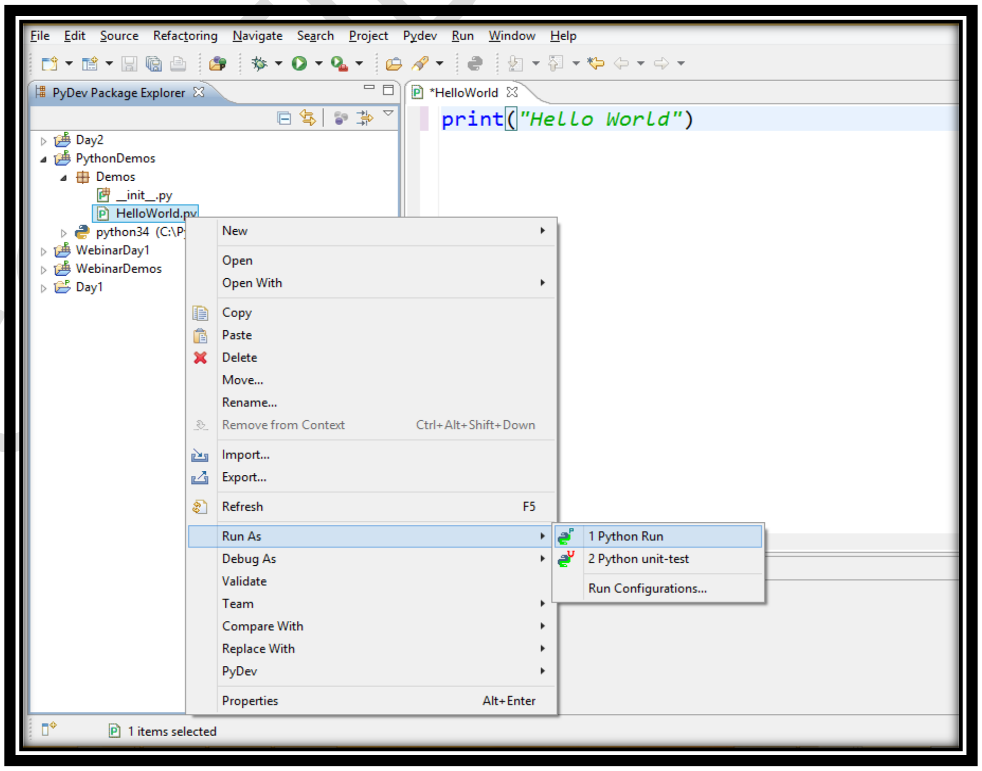

PyDev - PythonDemos/Demos/HelloWorld.py - Eclipse
File Edit Source Refactoring Navigate Search Project Pydev Run Window Help
PyDev Package Explorer
*HelloWorld
Day2
PythonDemos
Demos
__init__.py
HelloWorld.py
python34 (C:\Python34\python.exe)
WebinarDay1
WebinarDemos
Day1

```
print("Hello World")
```

Fig. 13 Sample Program

Executing Your Python Program

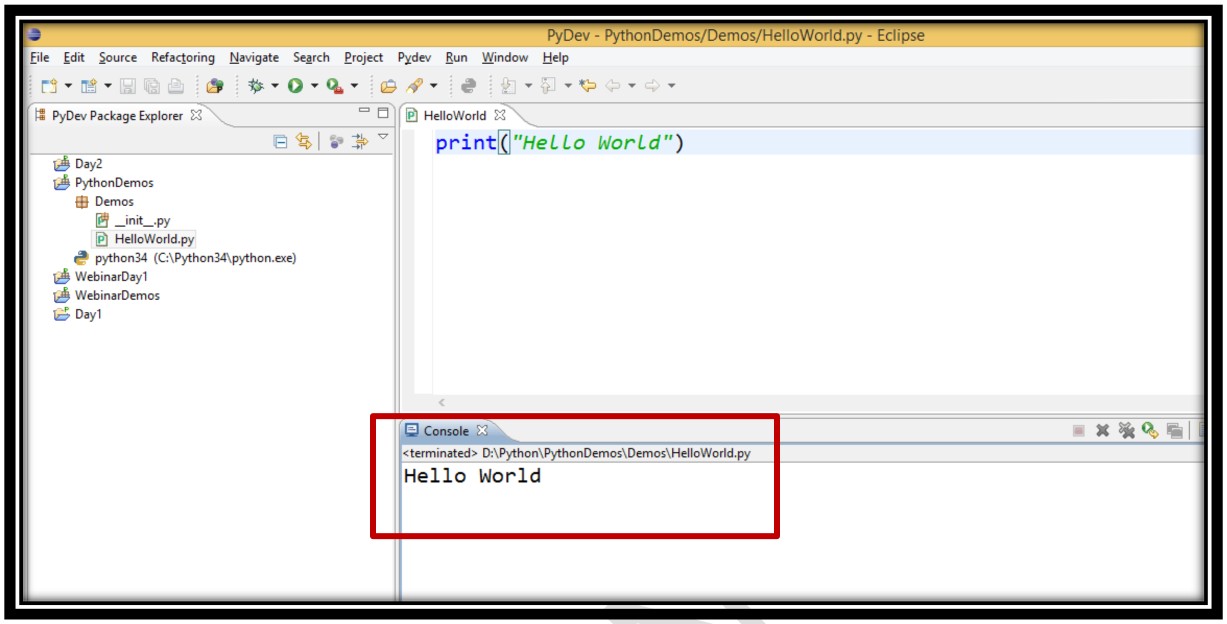
- To execute a PyDev Module, right click on the module-name in the project explorer and go to Run As → Python Run as shown in Fig. 14.



```
print("Hello World")
```

Fig. 14 Executing the Program

- The output of the program will be displayed in the Console Tab as shown in Fig.15.


PyDev - PythonDemos/Demos/HelloWorld.py - Eclipse
File Edit Source Refactoring Navigate Search Project Pydev Run Window Help
PyDev Package Explorer
Day2
PythonDemos
Demos
__init__.py
HelloWorld.py
python34 (C:\Python34\python.exe)
WebinarDay1
WebinarDemos
Day1
HelloWorld
Console

```python
print("Hello World")
```

```
<terminated> D:\Python\PythonDemos\Demos\HelloWorld.py
Hello World
```

Fig. 15 Output Window

Estimated time: 15 min

Summary of this assignment: You have now learnt how to create projects, how to create and execute a simple Python program using Eclipse IDE.

Assignment 2: Observations from a real world problem - Guided Activity

Objective: Given a real world problem, be able to understand the need for programming fundamentals such as identifiers, variables, data types etc

Problem Description: A retail store management wants to automate the process of generating the bill amount for its customers. As an initial step, they want to initialize the bill details of a customer as given below:
Bill id should be 1001, customer id should be 101 and bill amount should be 199.99.
After initializing, all the values must be displayed in the format given below:

bill_id: 1001
customer_id: 101
bill_amount: Rs.199.99

Analyze the above problem statement and answer the following questions:

1. What do you think is needed to write a program to implement the solution for the above problem statement?

Estimated time: 5 minutes

Summary of this assignment: In this assignment, you have understood the need of a high level programming language and programming fundamentals such as identifiers, variables, data types, operators etc

Assignment 3: Programming constructs in Python - Guided Activity

Objective: Given a real world problem, be able to identify the data types of the variables required to solve it

Problem Description: For the previous assignment, identify the data types that may be used to represent bill id, customer id and bill amount.

Estimated time: 5 minutes

Summary of this assignment: In this assignment, you have learnt to identify the data type for the variables based on the real world problem

Assignment 4: Programming constructs in Python - Quiz

Objective: Given a real world problem, be able to identify the variables and operators required to solve the problem and implement it using a high level programming language like Python

Problem Description: Predict the output of following Python Statements.

```
print ("Value of e is %0.1f" , 2.713 )
print ("Value of e is %0.1f" %2.713 )
print ("Programming in Python: Version %d" %3.5 )
print ("%20s : %d" % ("Python 3.0 is also known as Python",3000.57 ))
x=2
print ("{0:2d} {1:3d} {2:4d}".format(x, x*x, x*x*x))
```

Output:

```
Value of e is %0.1f 2.713
Value of e is 2.7
Programming in Python: Version 3
Python 3.0 is also known as Python 3000
 2   4    8
```

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have learnt to make use of format specifiers for different type of data variables based on the real world problem

Assignment 5: Programming constructs in Python - Demo

Objective: Given a real world problem, be able to identify the variables and operators required to solve the problem and implement it using a high level programming language like Python

Problem Description: Write a Pseudocode and Python program to implement the real world problem discussed in Programming constructs in Python Assignment 2. Compile and execute the program using Eclipse IDE.

You may want to refer to Programming constructs in Python Assignment 1 to understand how Eclipse IDE may be used for writing and executing a Python program.

INITIALIZE_VARIABLES_DISPLAY

```
1. bill_id = 1001
2. customer_id = 101
3. bill_amount = 199.99
4. display "Bill Id:", bill_id
5. display "Customer Id:", customer_id
6. display "Bill Amount:Rs.", bill_amount
```

```
bill_id = 1001
customer_id = 101
bill_amount = 199.99
print("Bill Id:%d" %bill_id)
print("Customer Id:%d" %customer_id)
print("Bill Amount:Rs.%f" %bill_amount)
```

Output:

```
Bill Id: 1001
Customer Id: 101
Bill Amount:Rs. 199.990000
```

Estimated time: 5 minutes

Summary of this assignment: In this assignment, you have understood how to implement the solution for a simple real world problem using variables and operators

Assignment 6: Programming constructs in Python – Hands on practice

Objective: Given a real world problem, be able to identify the variables and operators required to solve the problem and implement it using a high level programming language like Python

Problem Description: The finance department of a company wants to calculate the monthly pay of one of its employee. Monthly pay should be calculated as mentioned in the below formula and display all the employee details.

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

Note:
The number of hours worked by the employee in a week should be considered as 40,
Pay rate per hour should be considered as Rs.400 and
Number of weeks in a month should be considered as 4

Write Pseudo code and Python program in Eclipse to implement the above real world problem.

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have learnt to implement the solution for a simple real world problem using variables and operators

Assignment 7: Programming constructs in Python - Hands - on - Practice

Objective: Given a real world problem, be able to identify the variables and operators required to solve the problem and implement it using a high level programming language like Python

Problem Description: Write the assignment statement to swap 2 numbers x and y? (Without using temp variable)

Estimated time: 5 minutes

Summary of this assignment: In this assignment, you have learnt to implement the solution for a simple real world problem using variables and operators

Assignment 8: Programming constructs in Python - Guided Activity

Objective: Given a real world problem, be able to identify the variables and operators required to solve the problem and implement it using a high level programming language like Python

Problem Description: Predict the output of following code snippet

```
num = 16
num1 = num/6
num2 = num//6
num3 = num//6.0
print(num1)
print(num2);
print(num3)
```

Output:

```
2.6666666666666665
2
2.0
```

Estimated time: 5 minutes

Summary of this assignment: In this assignment, you have learnt to implement the solution for a simple real world problem using variables and operators

Assignment 9: id() and type() functions - Quiz

Objective: Revisit the concept and usage of id() and type() functions

Problem Description:

Dry run the below code snippets and predict the output. You may want to confirm the output by executing the code using Eclipse IDE.

Code – 1:

```
obj_x = 10
obj_y = obj_x

if ( id(obj_x) == id(obj_y) ):
    print("Address of obj_x and obj_y is same")
else:
    print("Address of obj_x and obj_y is not same")
```

Output:

```
Address of obj_x and obj_y is same
```

Code – 2:

```
obj_x = 10
obj_y = obj_x

if ( obj_x is obj_y ):
    print("obj_x and obj_y have same identity")
else:
    print("obj_x and obj_y do not have same identity")
```

Output:

```
obj_x and obj_y have same identity
```

Code – 3:

```
obj_x = 10
obj_y = obj_x
obj_y = obj_x + 1
if ( obj_x is not obj_y ):
    print("obj_x and obj_y do not have same identity")
else:
    print("obj_x and obj_y have same identity")
```

Output:

```
obj_x and obj_y do not have same identity
```

Code – 4:

```
int_a = 10
raw_input = input("Enter a number")
print("Type of int_a:", type(int_a))
print("Type of raw_input:", type(raw_input))
print(int_a + raw_input)
```

Output:

```
Enter a number 6
Type of int_a: <class 'int'>
Type of raw_input: <class 'str'>
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Estimated time: 15 minutes

Summary of this assignment: In this assignment, you have understood the concept and usage of id() and type() built-in functions.

Assignment 10: Coding Standards - Guided Activity

Objective: Given a set of source code and an identified set of best practices, be able to implement the techniques during coding

Problem Description: Coding standards are very important for maintenance of code and for understanding of the code. Identify the sections of the given program where the coding standards are not followed and correct them.

```
itemNo=1005
unitprice = 250
quantity = 2
amount=quantity*unitprice
print("Item No:",itemNo)
print("Bill Amount:",amount)
```

Estimated time: 5 minutes

Summary of this assignment: In this assignment, you have learnt Python coding standards

Assignment 11: Control Structures – Observations from a real world problem - Guided Activity

Objective: Given a real world problem be able to understand the need for control structures and operators to implement the logic and solve the problem

Problem Description: The scenario discussed in Programming Fundamentals Assignment 5 is revisited here. Suppose the retail store management now wants to provide 2% discount for all bill amounts above Rs.500 and for all other bill amount, a discount of 1%.

What do you think is needed to implement this scenario?

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have understood the need of operators and control structures using a real world problem

Assignment 12: Control Structures - Demo

Objective: Given a real world problem be able to understand the need for control structures and operators to implement the logic and solve the problem

Problem Description: Suppose the retail store management now wants to provide 2% discount for all bill amounts above Rs.500 and for all other bill amont, a discount of 1%. Write a Python program to implement the same?

```
bill_id = 1001
customer_id = 101
bill_amount = 200.0
discounted_bill_amount = 0.0
print("Bill Id:%d" %bill_id)
print("Customer Id:%d" %customer_id)
print("Bill Amount:Rs.%f" %bill_amount)
if bill_amount > 500:
    discounted_bill_amount = bill_amount - bill_amount * 2 / 100
else:
    discounted_bill_amount = bill_amount - bill_amount * 1 / 100
print("Discounted Bill Amount:Rs.%f" %discounted_bill_amount)
```

Note the use if else statement

Output:

```
Bill Id:1001
Customer Id:101
Bill Amount:Rs.200.000000
Discounted Bill Amount:Rs.198.000000
```

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have understood the implementation of operators and control structures using a real world problem

Assignment 13: Control Structures - Guided Activity

Objective: Given a real world problem be able to understand the need for control structures and operators to implement the logic and solve the problem

Problem Description: Suppose the retail store management now wants to provide discount for all bill amounts as mentioned below.

Assume bill amount will be always greater than 0.

Bill Amount	Discount %
>=1000	5
>=500 and <1000	2
>0 and <500	1

Write a Python program to implement the same?

```
bill_id = 1001
customer_id = 101
bill_amount = 1200.0
discounted_bill_amount = 0.0
discount = 0
print("Bill Id: %d" %bill_id)
print("Customer Id: %d" %customer_id)
print("Bill Amount:Rs. %f" %bill_amount)
if bill_amount >= 1000:
    discount = 5
elif bill_amount >= 500:
    discount = 2
else:
    discount = 1
discounted_bill_amount = bill_amount - bill_amount * discount / 100
print("Discounted Bill Amount:Rs. %f" %discounted_bill_amount)
```

Note the use of elif ladder

Output:

```
Bill Id: 1001
Customer Id: 101
Bill Amount:Rs. 1200.000000
Discounted Bill Amount:Rs. 1140.000000
```

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have understood the implementation of operators and control structures using a real world problem

Assignment 14: Control Structures - Guided Activity

Objective: Given a real world problem be able to understand the need for control structures and operators to implement the logic and solve the problem

Problem Description: Suppose the retail store management now wants to provide discount for all bill amounts as mentioned below.

Customers can be considered to be valid, if their customer id is between 101 and 1000(both inclusive).

For valid customers, discount must be provided as per the table given below:

Bill Amount	Discount %
>=500	10

Assume for all other cases, discount is 0%.

Note: Display appropriate error messages wherever applicable.

Write a Python Program to implement the same.

```
bill_id = 1001
customer_id = 101
bill_amount = 200.0
discounted_bill_amount = 0.0
print("Bill Id: %d" %bill_id)
print("Customer Id: %d" %customer_id)
print("Bill Amount:Rs. %f" %bill_amount)
if ((customer_id > 100) and (customer_id <= 1000)) is True:
    if bill_amount >= 500:
        discounted_bill_amount = bill_amount - bill_amount * 10 / 100
        print("Discounted Bill Amount:Rs. %f" %discounted_bill_amount)
    else:
        print("No Discount")
else:
    print("Invalid Customer id, customer id must between 101 and
1000")
```

Note the usage of logical AND operator

Note the use of nested if statement

Note the display of appropriate error messages

Output:

```
Bill Id: 1001
Customer Id: 101
Bill Amount:Rs. 200.000000
No Discount
```

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have understood the implementation of operators and control structures using a real world problem

Assignment 15: Control Structures – Hands – on – Practice

Objective: Given a real world problem be able to understand the need for control structures and operators to implement the logic and solve the problem

Problem Description: Objective: Given a real world problem be able to understand the need for control structures and operators to implement the logic and solve the problem

Problem Description: The finance department of a company wants to calculate the monthly net pay of one of its employee by finding the income tax to be paid (in Indian Rupees) and the net salary after the income tax deduction. The employee should pay income tax if his monthly gross salary is more than Rs. 10,000 (Indian Rupees) and the percentage of income tax should be considered as 20% of the gross salary. Display the employee id, basic salary, allowances, gross pay, income tax and net pay.

Note:
Employee Id must be considered as 1001,
Basic salary of the employee must be considered as Rs.15000.00 and
Allowances must be considered as Rs.6000.00

Write a Pseudo code and Python program in Eclipse to solve the above real world problem.

Refer below for the formulae to be used.

Monthly Gross Salary = Basic Salary + Allowances
Net Salary = Gross Salary – Income Tax

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have understood the implementation of operators and control structures using a real world problem

Assignment 16: Control Structures – Hands - on - Practice

Objective: Given a real world problem be able to understand the need for control structures and operators to implement the logic and solve the problem

Problem Description: Extend the program written for Assignment 15 to find the income tax to be paid (In Indian Rupees) and the total salary after the income tax deduction as per the details given in below table.

Gross Salary (In Indian Rupees)	Income Tax percentage
Below 5,000	Nil
5,001 to 10,000	10 %
10,001 to 20,000	20%
More than 20,000	30%

Display the employee id, basic salary, allowances, gross pay, income tax and net pay.

Note:
Employee Id must be considered as 1001,
Basic salary of the employee must be considered as Rs.15000.00 and
Allowances must be considered as Rs.6000.00

Write a Pseudo code and Python program in Eclipse to solve the above real world problem.

Estimated time: 15 minutes

Summary of this assignment: In this assignment, you have understood the implementation of operators and control structures using a real world problem

Assignment 17: Iteration Control Structures - Guided Activity

Objective: Given a real world problem, implement the logic and solve the problem using appropriate constructs (sequential, selection, iteration) using an object oriented programming language (Python)

Problem Description:

Dry run the below code snippets and predict the output. You may want to confirm the output by executing the code using Eclipse IDE.

Code – 1:

```
counter = 1

while counter <= 3:
    print(counter)
    counter += 1
print("End of Program")
```

Output:

```
1
2
3
End of Program
```

Code-2:

```
print("To find the sum of first 10 integers:")
result = 0
for value in range(1,11):
    result = result + value
print("Sum:",result);
```

Output:

```
To find the sum of first 10 integers:
Sum: 55
```

Code-3:

```
number = 1
result = 0
while number < 5:
    result = result + number
    number = number + 1
print(result)
```

Output:

```
10
```

Code-4:

```
result = 0
for index in range(40, 10, -2):
    if(index % 5 == 0):
        result = result + index
        print(result)
```

Output:

```
40
70
90
```

Code-5:

```
amount = 100.0
interest = 0.0
months = 1
while months < 6:
    interest = amount * 0.2
    amount = amount + interest
    months += 1
print(amount)
```

Output:

```
248.83200000000002
```

Estimated time: 20 minutes

Summary of this assignment: In this assignment, you have understood the implementation of iteration control structures.

Assignment 18: break statement - Demo

Objective: Given a real world problem, implement the logic and solve the problem using appropriate constructs (sequential, selection, iteration) using an object oriented programming language (Python)

Problem Description:

Dry run the below code snippet and predict the output. You may want to confirm the output by executing the code using Eclipse IDE.

Code:

```
count = 0
result = 0
for count in range (1, 10):
    result = result + count
    if result > 6:
        break
print("Result =", result)
```

Note the use of break statement

On encountering the break statement, control will move out of the loop as shown

Output:

```
Result = 10
```

Estimated time: 5 minutes

Summary of this assignment: In this assignment, you have understood the implementation of break statement.

Assignment 19: continue statement - Demo

Objective: Given a real world problem, implement the logic and solve the problem using appropriate constructs (sequential, selection, iteration) using an object oriented programming language (Python)

Problem Description:

Dry run the below code snippet and predict the output. You may want to confirm the output by executing the code using Eclipse IDE.

Code:

```
count = 0
for count in range(0,10):
    if 4 == count:
        continue
    print(count)
```

Note the use of continue statement

On encountering the continue statement, control will move to the increment portion of the for loop

Output:

```
0
1
2
3
5
6
7
8
9
```

Estimated time: 5 minutes

Summary of this assignment: In this assignment, you have understood the implementation of continue statement.

Assignment 20: Iteration Control Structure – Debugging - Guided Activity

Objective: Given a real world problem, implement the logic and solve the problem using appropriate constructs (sequential, selection, iteration) using an object oriented programming language (Python)

Problem Description: The code given below is written to display all the even numbers between 50 and 80 (both inclusive). Debug the program to get the correct output.

Step 1: Type the below program in Eclipse, save the file as for_loop.py, compile and execute.

```
for i in range (50, 80):
    if i % 2 == 0:
        print(i)
    else:
        break
```

Step 2: Correct the logical error in the code, save, compile and execute the code

Step 3: Implement the same logic using while loop

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have learnt the implementation of iteration control structure and break statement.

Strings - Assignments

Assignment 21: Strings – Observations from Retail Application - Guided Activity

Objective: Observe the need for features in the retail application scenario to correlate the application of Strings

Problem Description: In the retail application, along with the earlier details included for Customer, the retail shop wants to keep track of customer name. Customer name should be between 3 and 20 characters.

Answer the following question:

What do you think is needed to implement the solution for the above problem?

Estimated time: 5 minutes

Summary of this assignment: In this assignment, you have understood the need of strings using a retail application scenario

Assignment 22: Strings – Demo

Objective: Observe the need for features in the retail application scenario to correlate the application of Strings

Problem Description: In the retail application, along with the earlier details included for Customer, the retail shop wants to keep track of customer name. Customer name should be between 3 and 20 characters.

bill_id = 1001
customer_id = 1001
bill_amount = 2000
customer_name = Kevin

Code:

```
bill_id = input("Please enter Bill id")
customer_id = input("Please enter Customer id")
bill_amount = input("Please enter bill Amount")
customer_name = input("Please enter Customer Name")
if ((len(customer_name) >= 3) and (len(customer_name) <= 20)) is True:
    print("Bill Id: ",bill_id)
    print("Customer Id: ",customer_id)
    print("Bill Amount:Rs. ",bill_amount)
    print("Customer Name: ",customer_name)
else:
    print("Invalid customer name. Customer name must be between 3 and 
20 characters");
```

Note how len() method is used for checking the number of characters in a String object

Output:

```
Please enter Bill id101
Please enter Customer id1001
Please enter bill Amount2000
Please enter Customer NameKevin
Bill Id:  101
Customer Id:  1001
Bill Amount:Rs.  2000
Customer Name:  Kevin
```

Estimated time: 15 minutes

Summary of this assignment: In this assignment, you have understood the implementation of Strings and String methods using a retail application scenario

Assignment 23: Strings - Quiz

Objective: Revisit String concepts through a quiz

Problem Description:

Assume,
string1 = “Infosys Limited”
string2 = “Mysore”

Predict the output of following statements:

```
Q1: print(string1[:4])

Q2: print(string1[-1])

Q3: print(string1 * 2)

Q4: print(string1[:-1] + string2 + string1[:-1])

Q5: print (string2[1])

Q6: print (string2[4])

Q7: print(string1 * 2 + string1[:-1] + string2)
```

ANSWERS

```
Q1: Info
Q2: d
Q3: Infosys LimitedInfosys Limited
Q4: Infosys LimiteMysoreInfosys Limite
Q5: y
Q6: r
Q7: Infosys LimitedInfosys LimitedInfosys LimiteMysore
```

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have revisited concepts of strings through code snippets

Assignment 24: Strings – Hands - on - Practice

Objective: Given a computational problem, be able to use the right data structures (lists and strings) and implement the solution to the problem and test using a set of values in an IDE

Problem Description:

a. Write a program to count and display the number of capital letters in a given string.
b. Write a program to check if the given string is Palindrome or not?
c. Write a program to count the number of each vowel in a string
d. Write a program to remove all punctuation from the string provided by the user
   punctuations = '''!()-[]{};:'"\,<>./?@#$%^&*_~'''
   Hint: Use membership operators IN, Not IN

Estimated time: 20 minutes

Summary: In this assignment, you have understood the application and implementation of Strings concept for the given computational problem.

Assignment 25: Strings – Hands – on - Practice

Objective: Given a computational problem, be able to use the right data structures (lists and strings) and implement the solution to the problem and test using a set of values in an IDE

Problem Description: Write a Python program to accept a string and display the resultant string in reverse order. The resultant string should contain all characters at the even position of accepted string ignoring blank spaces.

```
accepted_string: An apple a day keeps the doctor away
resultant_string: Aapedyepteotrwy
expected_output:    ywrtoetpeydepaA
```

Estimated time: 20 minutes

Summary: In this assignment, you have understood the application and implementation of Strings concept for the given computational problem.

Assignment 26: Strings – Hands – on - Practice

Objective: Given a computational problem, be able to use the right data structures (lists and strings) and implement the solution to the problem and test using a set of values in an IDE

Problem Description: Given a string containing both upper and lower case letters. Write a Python program to count the number of repeated characters and display the maximum count of a character along with the character.

```
Sample Input: ABaBCbGc
Output:
        2A
        3B  (three times B is repeated)
        2C
        1G
```

Estimated time: 10 minutes

Summary: In this assignment, you have understood the application and implementation of Strings concept for the given computational problem.

Assignment 27: Strings – Hands - on - Practice

Objective: Given a computational problem, be able to use the right data structures (lists and strings) and implement the solution to the problem and test using a set of values in an IDE

Problem Description: Consider 2 strings string1 and string2 and display the merged_string as the output. The merged_string should be capital letters from both the strings in the order they appear.
Note: Each character should be checked if it is a capital letter and then it should be merged.

```
Sample Input:
string1: I Like C
string2: Mary Likes Python
```

merged_string: ILCMLP

Estimated time: 20 minutes

Summary: In this assignment, you have understood the application and implementation of Strings concept for the given computational problem.

Assignment 28: Operations on Tuples - Demo

Objective: To understand the operations that can be performed on Tuples through a quiz

Problem Description:

Assume,
head = ("CEO",)
elements = ('Air', 'Water', 'Fire', 'Light', 'Land')

Predict the output of following statements:

```
Q1: print(head)

Q2: print(elements)

Q2: print(elements[3])

Q3: elements[0]='Ice'

Q5: print(elements[-1])

Q6: print(elements[4])

Q7: print(elements[5])
```

ANSWERS

```
Q1: ('CEO',)
NOTE: In this case the symbol , is mandatory without which it becomes
just a string assignment operation

Q2: ('Air', 'Water', 'Fire', 'Light', 'Land')
```

```
Q3: Light

Q4: Type Error
Note: Tuples are immutable. Cannot change the contents of tuple

Q5: Land

Q6: Land

Q7: Index Error: tuple index out of range
```

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have revisited concepts of tuples through code snippets

Assignment 29: Tuples – Hands – on - Practice

Objective: Given a List of elements representing a computational problem, be able to provide solution by performing required operations using data structures like tuple from an object oriented language (Python) using Eclipse IDE

Problem Description: Consider the list of courses opted by a Student "John" and available electives as a part of Student Management System.

courses = ("Python Programming", "RDBMS", "Web Technology", "Software Engg").
electives = ("Business Intelligence", "Big Data Analytics")

Write a Python Program to satisy following business requirements:

a. List the number of courses opted by Student by "John"
b. List all the courses opted by Student "John".
c. Students "John" is also interested in elective course mentioned above. Print the updated tuple including electives
d. Check whether Student "John" is allowed to change his course from "Software Engg" to "Computer Networks". If yes, print the updated course list else mention the reason for the same.

Estimated time: 15 minutes

Summary of this assignment: In this assignment, you have learnt the implementation of Tuple operations for given business scenario.

Lists - Assignments

Assignment 30: Accessing Elements from Lists - Demo

Objective: Given a List of elements representing a computational problem, be able to access elements in different ways using an object oriented language (Python) using an IDE

Problem Description:

Assume,
language = ['Python']
languages = ['Python', 'C', 'C++', 'Java', 'Perl']

Predict the output of following statements:

```
Q1: print(language)

Q2: print(languages)

Q3: print(languages[0:3])

Q4: print(languages[1:4])

Q5: print(languages[2])

Q6: print(languages[5])

Q7: print (languages[0] + " and " + languages[1] + " are quite
different!")

Q8: print ("Accessing the last element of the list: " + languages[-1])
```

ANSWERS

```
Q1: ['Python']

Q2: ['Python', 'C', 'C++', 'Java', 'Perl']
```

```
Q3: ['Python', 'C', 'C++']

Q4: ['C', 'C++', 'Java']

Q5: C++

Q6: Index Error: list index out of range

Q7: Python and C are quite different!

Q8: Accessing the last element of the list: Perl
```

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have revisited concepts of lists through code snippets

Assignment 31: Lists – Hands - on - Practice

Objective: Given a computational problem, be able to use the right data structures (lists and strings) and implement the solution to the problem and test using a set of values in an IDE

Problem Description: Write a Python program to generate first 'n' Fibonacci numbers. Store the generated Fibonacci numbers in a list and display it.
Sample input: Enter n 5
Sample Output: List: [0, 1, 1, 2, 3]

Estimated time: 10 minutes

Summary: In this assignment, you have understood the application and implementation of Lists concept for the given computational problem.

Assignment 32: Lists – Hands - on - Practice

Objective: Given a List of elements representing a computational problem, be able to sort the elements in ascending/descending order using an object oriented language (Python) using an IDE

Problem Description: You have a bundle of currency of varied denominations. You want to arrange them in descending order.
Given below is one approach to perform the above operation.

Algorithm: Bubble Sort (List of N element)
Input: A List of N elements to be sorted
Output: A List of sorted (increasing order) of N elements

Step 1: Repeat steps 2 and 3 for n-1 times
Step 2: Repeat step 3 for 1 time less than n
Step 3: Compare first element with next element and swap if second element is greater

Pseudo-code is represented below:

```
1.    for i =1 to N -1
2.        for j = 1 to N - i
3.            if ( list[ j ]  > list[ j + 1]) then
4.                interchange (list[ j ] , list[ j +1])
5.            end-if
6.        end-for
7.    end-for
```

Write a Python program to sort the elements using bubble sort technique in the list and display the elements in descending order.
Sample Input: [4, 1, 6, 2, 8, 5]
Sorted: [1, 2, 4, 5, 6, 8]
Output: [8, 6, 5, 4, 2, 1]

Estimated time: 20 minutes

Summary: In this assignment, you have understood the application and implementation of Lists concept for the given computational problem.

Assignment 33: Lists - Hands - on - Practice

Objective: Given a computational problem, Implement the solution for the problem using suitable data structures from an object oriented language (Python) using an IDE

Problem Description: ABC Retail Store sells different varieties of Furniture to the customers. The list of furnitures available and its cost list are given below:

Furniture	Sofa set	Dining table	T.V. Stand	Cupboard
Cost in Rs.	20,000	8,500	4,599	13,920

The furniture's and its corresponding Cost should be stored as a list. If the required furniture is available in list of furniture's listed above and Quantity purchased is greater than zero, only then bill amount should be calculated. In case of invalid values for furniture required by the customer and quantity purchased, consider bill amount to be 0.
Initialize required furniture and required 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.

Estimated time: 15 minutes

Summary: In this assignment, you have understood the application and implementation of Lists concept for the given computational problem.

Assignment 34: Sets - Demo

Objective: Given a Set of elements representing a computational problem, be able to perform different operations using an object oriented language (Python) using an IDE

Problem Description:

Consider the sets,

fruits = {"apple", "orange", "banana", "apple", "pear", "papaya", "papaya"}

fruit_basket = {"apple", "banana", "grapes", "mango", "kiwi"}

Predict the output of following statements:

```
Q1: print(fruits)

Q2: print(fruits & fruit_basket)

Q3: print(fruits | fruit_basket)

Q4: print(fruits - fruit_basket)

Q5: print(fruits ^ fruit_basket)
```

```
Q6: print(len(fruit_basket))

Q7: print("pear" in fruits)

Q8: print("pear" not in fruit_basket)

Q9: print(fruits.issubset(fruit_basket))

Q10: print(fruits.issuperset(fruit_basket))

Q11: print(fruit_basket.copy())
```

ANSWERS

```
Q1: {'banana', 'apple', 'orange', 'papaya', 'pear'}

Q2: {'apple', 'banana'}

Q3: {'banana', 'mango', 'orange', 'apple', 'grapes', 'papaya', 'kiwi',
'pear'}

Q4: {'orange', 'papaya', 'pear'}

Q5: {'mango', 'orange', 'grapes', 'kiwi', 'papaya', 'pear'}

Q6: 5

Q7: True

Q8: True

Q9: False

Q10: False

Q11: {'mango', 'apple', 'grapes', 'kiwi', 'banana'}
```

Estimated time: 15 minutes

Summary of this assignment: In this assignment, you have revisited concepts of Sets through code snippets

Assignment 35: Sets – Hands - on - Practice

Objective: Given a Set of elements representing a computational problem, be able to provide solution by performing required operations on sets from an object oriented language (Python) using an IDE

Problem Description: Consider the scenario from course in student management system. Given below are 2 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 satisy below mentioned business requirements:

a. List the number of Students enrolled for Python course
b. List the names of Students enrolled for Java course only
c. List the names of Students enrolled for Python course only
d. List the number and names of Students enrolled for both Java and Python courses
e. List the number and names of Students enrolled for either Java or Python courses but not both
f. List names and number of Students enrolled for either Java or Python courses

Estimated time: 15 minutes

Summary of this assignment: In this assignment, you have learnt the implementation of Set operations for given business scenario.

Assignment 36: Dictionary - Demo

Objective: Given a Dictionary of elements representing a computational problem, be able to perform different operations using an object oriented language (Python) on Eclipse IDE

Problem Description:

Given below is a Dictionary customer_details representing customer Details from Retail Application - Customer Id is key and Customer Name is value.

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

Write Python code to perform below mentioned operations:

a. Print details of Customers

b. Print number of Customers
c. Print Customer names in ascending order
d. Delete the details of customer with customer id = 1005 and print updated dictionary
e. Update the name of customer with customer id = 1003 to “Mary” and print updated dictionary
f. Check whether details of customer with customer id 1002 exists in the dictionary.

ANSWERS

```
a: print(customer_details)

b: print(len(customer_details))

c: print(sorted(customer_details.values()))

d: del(customer_details[1005])
   print(customer_details)

e: customer_details[1003] = "Mary"
   print(customer_details)

f: print(1002 in customer_details)
```

Estimated time: 15 minutes

Summary of this assignment: In this assignment, you have learnt the implementation of Dictionary operations for given business scenario.

Assignment 37: Dictionary – Hands - on - Practice

Objective: Given a computational problem, select the right set of data structures (lists and dictionary) and implement the solution to problem and test using a set of values in an IDE

Problem Description: Consider the scenario of processing marks of students for a course in student management system. Given below is the list of marks scored by students. Find top three scorers for the course and also display average marks.

Implement the solution for given business scenario.

Student Name	Marks Scored
John	86.5
Jack	91.2
Jill	84.5
Harry	72.1
Joe	80.5

Estimated Time: 20 minutes

Summary: In this assignment, you have understood the application and implementation of Dictionary concept for the given computational problem.

Assignment 38: Functions – Pass by Reference - Demo

Objective: Given a computational problem, implement functions and solve it using parameter passing technique followed in python

Problem Description: Consider the distance in miles between two locations:

- distance between Phoenix, Arizona and Salt Lake City, Utah in USA
- distance between Phoenix, Arizona and Tampa, Florida in the US.

We want to compare the distances and check which one is far off from Phoenix, Arizona.

Provided below are codes written to solve the above problem using pass by reference technique – using Required arguments and Keyword Arguments.

Code:

Method 1: Pass by Reference – Required Arguments

Phoenix_to_tampa is formal argument

```
def compare(phoenix_to_slc, phoenix_to_tampa):
    if phoenix_to_slc > phoenix_to_tampa:
        print("SLC is far from Phoenix compared to Tampa, Florida")

    elif(phoenix_to_slc < phoenix_to_tampa):
        print("Tampa, Florida is far from Phoenix compared to SLC")

    else:
        print("Both locations are equidistance from Phoneix")
```

```
compare (1790, 506)
```

506 is the actual argument which is passed and assigned to the formal argument, phoenix_to_tampa, when the method compare() is invoked. It is mandatory to pass both the arguments

Method 2: Pass by Reference – Keyword Arguments

```
def compare(phoenix_to_slc, phoenix_to_tampa):
    if phoenix_to_slc > phoenix_to_tampa:
        print("SLC is far from Phoenix compared to Tampa, Florida")

    elif(phoenix_to_slc < phoenix_to_tampa):
        print("Tampa, Florida is far from Phoenix compared to SLC")

    else:
        print("Both locations are equidistance from Phoneix")

compare (phoenix_to_tampa = 506, phoenix_to_slc = 1790)
```

Here parameters are passed by keyword arguments. Hence, it is not mandatory to pass both the arguments and need not follow positional order

Output:

```
SLC is far from Phoenix compared to Tampa, Florida
```

Estimated Time: 20 minutes

Summary of this assignment: In this assignment, you have learnt the implementation of functions and parameter passing technique - pass by reference through keyword arguments and reference arguments

Assignment 39: Functions – Pass by Reference – Hands - on - practice

Objective: Given a computational problem, Implement functions and solve it using appropriate parameter passing techniques (pass by reference)

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

i. **Baggage Check**
ii. **Immigration Check**
iii. **Security Check**

The logic for the check methods are given below:

Implementation details

check_baggage (baggage_amount)

- Check if baggage_amount is greater than or equal to 0 and less than or equal to 40.

- If *baggage_amount* is VALID
    - return TRUE

  else
    - return FALSE

check_immigration (expiry_year)

- Check if expiry_year is greater than or equal to 2001 and less than or equal to 2025.

- If *expiry_year* is VALID
    - return TRUE

  else
    - return FALSE

check_security(noc_status):boolean

- If *noc_status* is TRUE
    - return TRUE

  else
    - return FALSE

traveler()

In *traveler()* function, 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.

Variable	Value
traveler_id	1001
traveler_name	Jim
baggageAmount	35
expiryDate	2019
nocStatus	true

If all values of *check_baggage(), check_immigration()* and *check_security()* are **true,**
diplay *traveler_id* and *traveler_name*
display “Allow Traveller to fly!”
else,

display *traveler_id* and *traveler_name*
display “Detain Traveller for Re-checking!”

Estimated time: 25 minutes

Summary of this assignment: In this assignment, you have learnt pass by reference technique

Assignment 40: Strings built - in functions - Guided Activity

Objective: Revisit String built-in functions through match the following.

Problem Description:

There are many built-in String methods available in Python. Match each method to what it does.

Method	**What the method does**
text.endswith(“.jpg”)	Return a copy of the string with all occurences of one substring replaced by another
text.upper()	Return a copy of the string converted to lowercase
text.lower()	Return the value **True** if the string has the given substring at the beginning
text.replace(“tomorrow”, “Saturday”)	Return the value **True** if the string has the given substring at the end
text.strip()	Returns the first index value when the given substring is found
text.find(“python”)	Return a copy of the string with the leading and trailing whitespace removed
Text.startswith(“<HTML>”)	Return a copy of the string converted to uppercase

ANSWERS:

Method	**What the method does**
text.endswith(“.jpg”)	Return the value True if the string has the given substring at the end
text.upper()	Return a copy of the string converted to uppercase
text.lower()	Return a copy of the string converted to lowercase

text.replace(“tomorrow”, “Saturday”)	Return a copy of the string with all occurences of one substring replaced by another
text.strip()	Return a copy of the string with the leading and trailing whitespace removed
text.find(“python”)	Returns the first index value when the given substring is found
Text.startswith(“<HTML>”)	Return the value True if the string has the given substring at the beginning

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have revisited String built-in functions through match the following.

Assignment 41: Lists built-in functions - Guided Activity

Objective: Given a computational problem, select the right set of data structures (lists and dictionary) and implement the solution to problem and test using a set of values in an IDE

Problem Description: Consider the price list of various items in the Retail Store. Customer John wants to know the

i) Price of Costliest item sold in Retail store
ii) Average price of items in the Retail store
iii) Display of item price list in increasing order

Implement the solution using user-defined and built-in functions to accomplish above mentioned business requirement.

Code:

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

def price_max(item_price):
    print("Price of Costliest item in the Retail Store:
",max(item_price))

def average_price(item_price):
    total_price = 0
    for i in item_price:
        total_price += i
    average_price = total_price/len(item_price)
    print("Average Price of items in the Retail Store:
",average_price)
```

```
def display_sorted_price(item_price):
    item_price.sort()
    print("Item Price List in increasing order: ",item_price)

price_max(item_price)
average_price(item_price)
display_sorted_price(item_price)
```

Output:

```
Price of Costliest item in the Retail Store:  8575
Average Price of items in the Retail Store:  1870.4285714285713
Item Price List in increasing order:  [150, 234, 399, 485, 1050, 2200,
8575]
```

Estimated Time: 20 minutes

Summary: In this assignment, you have understood the application and implementation of Lists concept for the given computational problem.

Assignment 42: Exception Handling – Handson

Objective: Given a computational problem, implement the solution to the given problem using exception handling.

Problem Description: Consider the customer ids of a customer in the Retail Store. Customer id can be vary between 1001 to 1005. Store the customer id in a list and handle appropriate exceptions for the following:

i) Store the customer id "1002" as string
ii) Print customerid[5]

Object Oriented Fundamentals- Assignments

Assignment 43: Identify class and objects

Objective: Given a business scenario, able to identify the classes and objects

Problem Description:
A supermarket wants to automate the system of purchase of items by customers and the billing process. The automation involves the maintenance of items, employees, customers, purchase of items by customer and billing of items. Customers can be regular visitors to the store in which case they are eligible for discounts based on the bill amount. The customers can also be privileged ones, wherein they are given membership cards (Platinum, Gold and Silver). Such customers are eligible for gifts based on the type of membership card. The billing staff does the billing and delivery of items to the customer. The bill calculation involves the logic of computation of the bill depending on customer type. The customer can pay the bill through credit card or cash. In the former case, two percent processing charge is applicable. Sales tax is also applicable on the final bill amount. Employees in that supermarket can be permanent and temporary. Permanent employees will get additional benefits in salary.

Questions

- Identify the classes.
- Identify the attributes / behaviors associated with each class.

Estimated time: 15 minutes

Summary of this assignment: In this assignment, you have revisited object oriented concepts using scenarios

Assignment 44: Object Oriented Concepts - Quiz

Objective: Revisit object oriented concepts using a quiz

Answer the following questions:

1. In the ATM machine, the customer chooses the operations using a touch screen. The customer need not know the internal working of the ATM machine. Which OO concept(s) can be used in this scenario?

2. Consider the following statement: “Vehicles can be of two types viz. Water vehicles and Land vehicles “. Which OO concept may be used to represent this scenario?

3. As part of our family trip plan we went to Zoo. My son asked me lot of questions I tried my level best to answer all of his questions. I showed him different types of monkeys which were locked in different rooms. He asked me, “Dad, you said all are monkeys then why they are kept in different rooms?” Now, which OO concept may be used to represent this scenario?

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have revisited object oriented concepts using scenarios

Assignment 45: Object Oriented Concepts – Rapid Fire

Objective: Revisit object oriented concepts using a simple True/False questions

Answer the following questions:

1. Wrapping up of data of different types into a single unit is known as encapsulation.
2. Inheritance means the ability to reuse the data values of one object by other objects.
3. Polymorphism is extensively used in implementing inheritance.
4. Object-oriented systems can scale up better from small to large
5. Object-based languages do not support inheritance and dynamic binding.

Estimated time: 5 minutes

Summary of this assignment: In this assignment, you have revisited object oriented concepts using simple true/false questions

Assignment 46: Introduction to UML

Objective: Given a business scenario, be able to identify the components of a use case diagram

Problem Description: Refer to the course registration system case study and answer the following questions.

Situation: A Course Registration System needs to be developed for an engineering college. The college wants an automated system to replace its manual system for the purpose of registration of students to branches and calculation of fees for each year. The engineering college provides graduation courses in various branches of engineering.
The system will be used by the admin staff to register students admitted to the college to the branches at the time of joining the college and also to calculate the yearly fees for the students. The student has to register every year for the next academic year. The Admin takes care of the yearly registration of the students and the calculation of yearly fees. The system needs to be authenticated with a login id and password.
Registration of a student to a branch is based on the qualifying exam marks and the entrance counseling. For every branch, a yearly branch fee is applicable. Discounts are given to the branch fees of the first year based on the qualifying exam marks. There is a registration fees also applicable to the first year students. Students can opt to be a day scholar or hostelite. Yearly bus fees are applicable for all the day scholars based on the

distance of travel. Yearly hostel fees are applicable for all the hostelites. Yearly infrastructure fees and library fees are also applicable to all the students. Admin calculates the yearly college fees for each student and the college fees include all the fees specified earlier based on the type of student. Admin will provide a printed receipt of the fees to the students once the annual college fees have been paid.
At the time of registration, student has to provide the permanent address and in case the student is opting to be a day scholar, he/she has to provide the residential address also.
Assumption:

1. Decision of the branch of study a student is allocated, is not within the scope of this case study

Questions:

1. Identify all the classes of the course registration system
2. Identify the attributes and behaviors of those classes as well.
3. Draw the complete class diagram using the information collected.

Estimated time: 20 minutes

Summary of this assignment: In this assignment, you have learnt how to actors and activities of a use case diagram using a course registration system scenario

Assignment 47: Classes & Objects – Hands-on

Objective: Given a class diagram for a use case representing a computational problem, be able to recognize the three compartments of the class diagram and implement it using an object oriented language (Python) using an IDE

Problem Description: In the retail application, there are many customers who visit the retail outlet to purchase various items. The manager of the retail outlet now wants to keep track of all its customers' data. Let us assume that customer details include Customer Id and Telephone Number.

For the class diagram identified in OO Fundamentals, implement the class using Eclipse IDE and execute it by writing a starter class.

Code: Execute the code using Eclipse IDE with the given inputs and understand the following:

- Access Specifiers
- Variables – Local and Instance variables
- Methods
- Starter class
- Creation of objects
- Reference variables
- Compilation and Execution of a python program

Note the use of access specifiers in the class, Customer. __ indicates that the variable is **private**

Note how the class, Customer is written

Member method should contain 'self' as first parameter

Note the use of member methods in the class, Customer

Note the how local variables are used in the member methods of class, Customer

Note the how method header, method definition, return statement is written in member method of class, Customer

Note how object of Customer class is created using reference variable

Note how member methods of Customer class are invoked using reference variable and dot operator

Note the use of print() statement for display of console output

```
class Customer:
     def setcustomerid(self, id):
          self.__customerid = id

     def settelephoneno(self, teleno):
          self.__telephoneno = teleno

     def getcustomerid(self):
          return self.__customerid

     def gettelephoneno(self):
          return self.__telephoneno


custobj = Customer()
custobj.setcustomerid(1001)
custobj.settelephoneno(9201861311)
print("Customer Id : ", custobj.getcustomerid())
print("Telephone No : ", custobj.gettelephoneno())
```

Estimated time: 15 minutes

Summary of this assignment: In this assignment, you have understood object oriented fundamentals - Access Specifiers, Variables – Local and Instance variables.

Assignment 48: self reference - Demo

Objective: Given a class diagram for an use case representing a computational problem, use the 'self' reference to create and initialize instance variables of a class and test using a set of values in an IDE

Problem Description: Let us revisit the class diagram drawn and implemented for Customer class as part of Object Oriented Fundamentals.

Class diagram:

Customer	
-customerid	: int
-telephoneno	: long
+setcustomerid(int)	: void
+getcustomerid()	: int
+settelephoneno(long)	: void
+gettelephoneno()	: long

Code:

Execute the code using Eclipse IDE with the given inputs and observe the results.

```
class Customer:
     def setcustomerid(self, customerid):
          customerid = customerid

     def settelephoneno(self, teleno):
          telephoneno = teleno

     def getcustomerid(self):
          return customerid

     def gettelephoneno(self):
          return telephoneno

custobj = Customer()
custobj.setcustomerid(1001)
custobj.settelephoneno(9201861311)
print("Customer Id : ", custobj.getcustomerid())
print("Telephone No : ", custobj.gettelephoneno())
```

Output:

```
Traceback (most recent call last):
  File "D:/****/Assignment6.py", line 17, in <module>
    print("Customer Id : ", custobj.getcustomerid())
  File "D:/****/Assignment6.py", line 9, in getcustomerid
    return customerid
NameError: name 'customerid' is not defined
```

Why do you get this error message?

Note: Without 'self', it will be considered as normal local variable and that will be removed at the end of the scope of method. Hence, it is not possible to access it from outside of the method where it is defined.

Revised Code:

Note the use of 'self' reference to create member variables for a class

```
class Customer:
     def setcustomerid(self, customerid):
          self.customerid = customerid

     def settelephoneno(self, telephoneno):
          self.telephoneno = telephoneno

     def getcustomerid(self):
          return self.customerid

     def gettelephoneno(self):
          return self.telephoneno

custobj = Customer()
custobj.setcustomerid(1001)
custobj.settelephoneno(9201861311)
print("Customer Id : ", custobj.getcustomerid())
print("Telephone No : ", custobj.gettelephoneno())
```

To make this private, need to insert __ as prefix to the data member i.e. Self.__telephoneNo=telephoneNo

Output:

```
Customer Id: 1001
Telephone No: 9201861311
```

Note: "self" is not a keyword and has no special meaning in Python. We can use any name in that place. However, it is recommended not to use any name other than "self" (merely a convention and for readability)

Estimated time: 15 minutes

Summary of this assignment: In this assignment, you have learnt the usage of self reference for accessing the instance variables using a retail application scenario

Assignment 49: __init__() method & Static members – Observations from Retail Application

Objective: Observe the need for features in the retail application scenario to correlate the application of initializing the members and 'static' members

Problem Description: In the retail application, each time a customer is registered, customer id must be automatically generated starting from 1001 and the details of the customer must be initialized. Also, retail shop management wants to know how many customers have registered at a point of time.

Answer the following question:
What do you think is needed to implement the solution for the above problem?

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have learnt
- The need of __init__() method and static keyword in the retail application scenario

Assignment 50: __init__() method (Constructors Equivalent) – Demo

Objective: Given a class diagram for a use case representing a computational problem, initialize the data members using __init__() method and test using a set of values in an IDE

Problem Description: In the retail application, as a customer is registered to the retail shop, the details of the customer must also be initialized.

Class diagram:

Customer	
-customerid	: int
-telephoneno	: long[]
-customername	: String
+ setcustomerid(int)	: void
+getcustomerid()	: int
+settelephoneno(long[])	: void
+gettelephoneno()	: long[]
+setcustomername(String)	: void
+getcustomername()	: String
+validatecustomername()	: boolean

Code:

Execute the code using Eclipse IDE with the given inputs and observe the results.

```
class Customer:
    def setcustomerid(self, id):
```

Note that since we have not explicitly coded the __init()__ method, system will not initialize the instance variables to their default values

```
          self.__customerid = id

     def setcustomername(self, customername):
          self.__customername = customername

     def settelephoneno(self, teleno):
          self.__telephoneno = teleno

     def getcustomerid(self):
          return self.__customerid

     def gettelephoneno(self):
          return self.__telephoneno

     def getcustomername(self):
          return self.__customername

     def validatecustomername(self):
          if(len(self.__customername)>=3 and len(self.__customername)
<=20):
               return True
          else:
               return False

custobj = Customer()
print("Customer Id : ", custobj.getcustomerid())
print("Telephone Nos : ", custobj.gettelephoneno())
print("Customer Name : ", custobj.getcustomername())
```

Output:

```
Traceback (most recent call last):
  File "D:/****/Assignment9.py", line 27, in <module>
    print("Customer Id : ", custobj.getcustomerid())
  File "D:/****/Assignment9.py", line 12, in getcustomerid
    return self.__customerid
AttributeError: 'Customer' object has no attribute '_Customer__customerid'
```

Note: There is no default __init()__ method in Python as like we have default constructor in Java. Hence, we have to initialize some value to the variable if we want to use it in the program later.

Revised Code: The code above has been revised by adding __init__() method

```
class Customer:
     def __init__(self):
          self.__customerid=0
          self.__customername=None
          self.__telephoneno=[]

     def setcustomerid(self, id):
          self.__customerid = id

     def setcustomername(self, customername):
          self.__customername = customername

     def settelephoneno(self, teleno):
          self.__telephoneno = teleno

     def getcustomerid(self):
          return self.__customerid

     def gettelephoneno(self):
          return self.__telephoneno

     def getcustomername(self):
          return self.__customername

     def validatecustomername(self):
          if(len(self.__customername)>=3 and len(self.__customername)
<=20):
               return True
          else:
               return False

custobj = Customer()
print("Customer Id : ", custobj.getcustomerid())
print("Telephone Nos : ", custobj.gettelephoneno())
print("Customer Name : ", custobj.getcustomername())
```

Do you think this will satisfy the requirement provided as part of problem description?

Output:

```
Customer Id :  0
Telephone Nos :  []
Customer Name :  None
```

No…Values initialized for each customer must be different. It depends on each customer. This can be achieved using parameterized __init__()

Revised Code 2: The code above has been revised by adding parameterized __init__() method

```
class Customer:
     def __init__(self, customerid, telephoneno, customername):
          self.__customerid=customerid
          self.__customername=customername
          self.__telephoneno=telephoneno

     def setcustomerid(self, id):
          self.__customerid = id

     def setcustomername(self, customername):
          self.__customername = customername

     def settelephoneno(self, teleno):
          self.__telephoneno = teleno

     def getcustomerid(self):
          return self.__customerid

     def gettelephoneno(self):
          return self.__telephoneno

     def getcustomername(self):
          return self.__customername

     def validatecustomername(self):
          if(len(self.__customername)>=3 and len(self.__customername)
<=20):
               return True
          else:
               return False

telephoneno=[9201861311, 9201861321, 9201661311]
custobj = Customer(1001, telephoneno, "Kevin")
if(custobj.validatecustomername()):
     print("Customer Id : ", custobj.getcustomerid())
     temp = custobj.gettelephoneno()
     print("Telephone Nos : ", temp[0], ",", temp[1], ",", temp[2])
     print("Customer Name : ", custobj.getcustomername())
else:
     print("Invalid customer name. Customer name must be between 3 and
20 characters")
```

Output:

```
Customer Id :  1001
Telephone Nos :  9201861311 , 9201861321 , 9201661311
Customer Name :  Kevin
```

Estimated time: 20 minutes

Summary of this assignment: In this assignment, you have understood the implementation of __init__() method and the need of parameterized __init__() using a retail application scenario.

Assignment 51: Parameterized __init__() - Demo

Objective: Given a class diagram for a use case representing a computational problem, initialize the data members using default constructors, parameterized __init__() and test using a set of values in an IDE

Problem Description: In the retail application, as a customer is registered to the retail shop, the details of the customer must also be initialized.

The class diagram discussed in Object Oriented Fundamentals Assignment 51 has been modified as shown below.

Class diagram:

Customer	
-customerid	: int
-telephoneno	: long[]
-customername	: String
+ Customer(int, long[], String)	
+ setcustomerid(int)	: void
+getcustomerid()	: int
+settelephoneno(long[])	: void
+gettelephoneno()	: long[]
+setcustomername(String)	: void
+getcustomername()	: String
+validatecustomername()	: boolean

Code:

Execute the code using Eclipse IDE with the given inputs in the starter class, Retail and observe the results.

```
class Customer:
     def __init__(self, customerid, telephoneno, customername):
          self.__customerid=customerid
          self.__customername=customername
          self.__telephoneno=telephoneno

     def setcustomerid(self, id):
          self.__customerid = id

     def setcustomername(self, customername):
          self.__customername = customername

     def settelephoneno(self, teleno):
          self.__telephoneno = teleno

     def getcustomerid(self):
          return self.__customerid

     def gettelephoneno(self):
          return self.__telephoneno

     def getcustomername(self):
          return self.__customername
```

Note how parameterized __init__() method is written

```
    def validatecustomername(self):
        if(len(self.__customername)>=3 and len(self.__customername)
<=20):
            return True
        else:
            return False

telephoneno=[9201861311, 9201861321, 9201661311]
custobj = Customer(1001, telephoneno, "Kevin")
if(custobj.validatecustomername()):
    print("Customer Id : ", custobj.getcustomerid())
    temp = custobj.gettelephoneno()
    print("Telephone Nos : ", temp[0], ",", temp[1], ",", temp[2])
    print("Customer Name : ", custobj.getcustomername())
else:
    print("Invalid customer name. Customer name must be between 3 and
20 characters")
```

Note how parameterized __init__() method is invoked

Output:

```
Customer Id:1001
Telephone Nos:9201861311, 9201861321, 9201661311
Customer Name:Kevin
```

Summary of this assignment: In this assignment, you have understood the implementation of parameterized __init__() method and the need of parameterized __init__() using a retail application scenario.

Assignment 52: Instance and Static Methods - Demo

Objective: Given a class diagram for a use case representing a computational problem, use static/class variable and test using a set of values in an IDE

Problem Description: In the retail application, each time a customer is registered, customer id must be automatically generated starting from 1001. Also, retail shop management wants to know how many customers have registered at a point of time.

Note: Few instance variables are not shown in the class diagram so as to keep the code simple to understand.

Class diagram:

Customer	
-customerid	: int
+Customer()	
+setcustomerid(int)	: void
+getcustomerid()	: int
+totalnoofcustomers()	: int

Code:

Execute the code using Eclipse IDE with the given inputs in the starter class, Retail and observe the results.

```
class Customer:
     def __init__(self, cid=1000):
          self.__customerid=cid+1

     def setcustomerid(self, cid):
          self.__customerid=cid

     def getcustomerid(self):
          return self.__customerid

     def totalcustomers(self):
          return 0

objcust = Customer()
print("Customer Id: ", objcust.getcustomerid())

objcust2 = Customer()
print("Customer Id: ", objcust2.getcustomerid())
```

Output:

```
Customer Id:1001

Customer Id: 1001
```

Why do you think customer id is not auto incremented for each customer?

customerId is an instance variable of Customer class and it will be created separately for each and every object of Customer class. Hence each time an object of Customer class is created, constructor is called and customerId will be initialized to 1000 and will be incremented by 1 in the constructor as a result for all the objects customerId will remain as 1001. Hence we need to have a variable common to all the objects of Customer class

Modified Code: A modified version of the above code is provided below

Execute the code using Eclipse IDE with the given inputs in the starter class, Retail and observe the results.

Revised Code using static method:

```
class Customer:
    counter = 1000
    def __init__(self):
        Customer.counter = Customer.counter+1
        self.__customerid=Customer.counter

    def setcustomerid(self, cid):
        self.__customerid=cid

    def getcustomerid(self):
        return self.__customerid

    @staticmethod
    def totalcustomers():
        return Customer.counter-1000

objcust = Customer()
print("Customer Id: ", objcust.getcustomerid())

objcust2 = Customer()
print("Customer Id: ", objcust2.getcustomerid())

print("Total Customers: ", objcust.totalcustomers())
print("Total Customers: ", objcust2.totalcustomers())
print("Total Customers: ", Customer.totalcustomers())
```

Note how the class/static variable is created for the class

Note how the class/static variable is accessed

Note the use of @staticmethod to convert a method into static one

“self” argument is not needed for static method

Note the different ways of invoking the static methods in the class.

Output:

```
Customer Id:  1001
Customer Id:  1002
Total Customers:  2
Total Customers:  2
Total Customers:  2
```

Summary of this assignment: In this assignment, you have understood the implementation of instatnce methods and static methods.

Assignment 53: Using default parameters - Observations from Retail application

Objective: Observe the need for features in the retail application scenario to correlate the application of method overloading

Problem Description: The Retail Store has the requirement for printing many reports including the bill. All reports contain header. The header may be

- a line containing a character printed 70 times or
- a title of a report or
- a line containing a character specified number of times

Answer the following questions:

1. Which object oriented concept do you think is needed to implement this scenario?

Estimated time: 10 minutes

Summary of this assignment: In this assignment, you have understood the need of method overloading using a retail application scenario

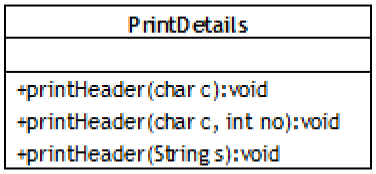
Assignment 54: Using default parameters in a method – Demo

Objective: Given a class diagram for a use case representing a computational problem, use method and constructor overloading techniques to solve the problem and test using a set of values in an IDE

Problem Description: The scenario discussed in OO Concepts Part I Assignment 1 is revisited here. The Retail Store has the requirement for printing many reports including the bill. All reports contain header. The header may be

- a line containing a character printed 70 times or
- a title of a report or
- a line containing a character specified number of times

A new class called ***PrintDetails*** with overloaded methods have to be created as shown below:


PrintDetails
+printHeader(char c):void
+printHeader(char c, int no):void
+printHeader(String s):void

Code:

Execute the code using Eclipse IDE with the given inputs in the starter class, Retail and observe the results.

Note how default arguments are used to bring method overloading equivalent in printHeader()

```
class PrintDetails:
     def printheader(self, c='*', no=1):
          print(c * no)


obj = PrintDetails()
obj.printheader('#', 10)
obj.printheader("Report")
obj.printheader('#' ,10)
obj.printheader()
```

Output:

```
##########
Report
##########
*
```

Summary of this assignment: In this assignment, you have learnt the need and implementation of method with default parameters using a retail application scenario.

Assignment 55: Relationships – Demo

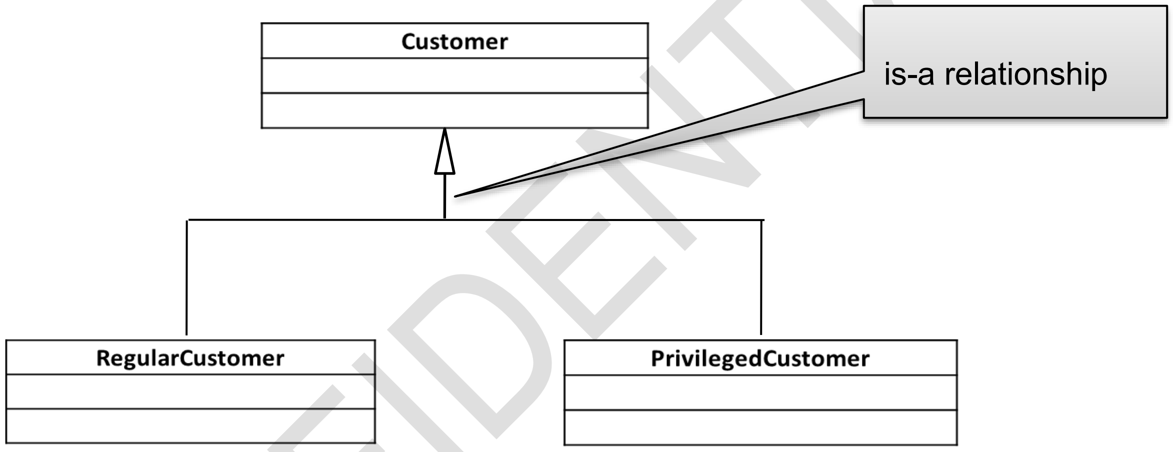
Objective: Observe the need for features in the retail application scenario to correlate the application of relationships concept

Problem Description:

1. The retail shop has two types of customers, Regular and Privileged.

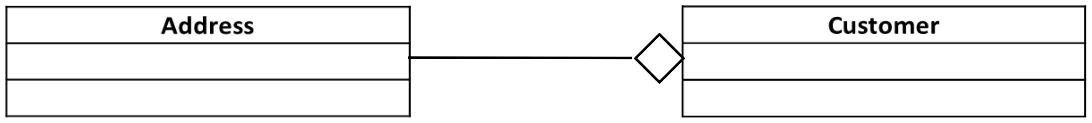
a. What are the properties common to all Customers?
b. What are the properties which are different for the two kinds of customers?
c. What is the relationship between
   i. Customer and Regular customer
   ii. Customer and Privileged customer?
d. How can this relationship be represented using class diagram?

Regular customers are entitled to get some discount on each purchase apart from having all the properties of a customer. On the other hand, privileged customers hold membership card – Platinum, Gold or Silver based on which they receive gifts on each purchase. This is in addition to having all the properties of a customer. Thus in this case, Customer is the generalized case and Regular and Privileged customer are the specialized cases of Customer. This relationship is known as "is-a" relationship and can be represented using class diagram as shown below:



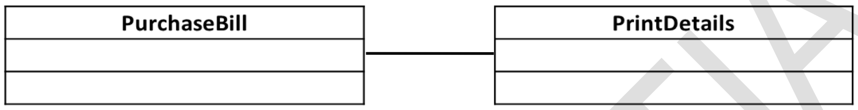
2. The retail store management wants to keep track of the address of every customer so as to allow for home delivery at a later point of time.
a. How many fields represent the address?
b. Do you think address qualifies to be a class?
c. What do you think is the relationship between Customer and Address?
d. How can this relationship be represented using class diagram?

Address can be considered as a separate class since the same class if required can be reused at a later point of time. For example, if retail store needs to keep track of its employees address, the same address class can be reused. The relationship between Customer class and Address class will be "Has –A" relationship since every customer has an address. This relationship can be represented using class diagram as shown below:



3. Consider PurchaseBill and PrintDetails classes, the bill needs to be created in a particular format. PrintDetails class contains methods which can be used to display the bill in the required format.
    a. What do you think is the relationship between PurchaseBill and PrintDetails classes?
    b. How can this relationship be represented using class diagram?

As stated, PrintDetails class contains methods which can be used to display the bill in the required format. Hence the relationship between PurchaseBill and PrintDetails class is "uses-a" relationship. This relationship can be represented using class diagram as shown below:



Estimated time: 20 minutes

Summary of this assignment: In this assignment, you have understood the need of relationships and its representation using a retail application scenario

Assignment 56: Inheritance – Demo

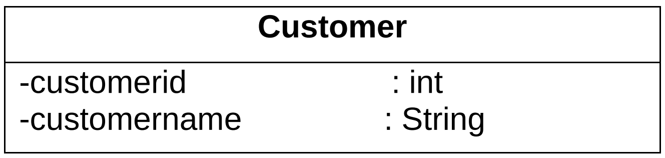
Objective: Given a class diagram for a use case representing a computational problem, implement inheritance (single level, multilevel and hierarchical) to solve the problem, trace the flow of the program between the base and derived classes and test using a set of values in an IDE and recognize the benefits of inheritance

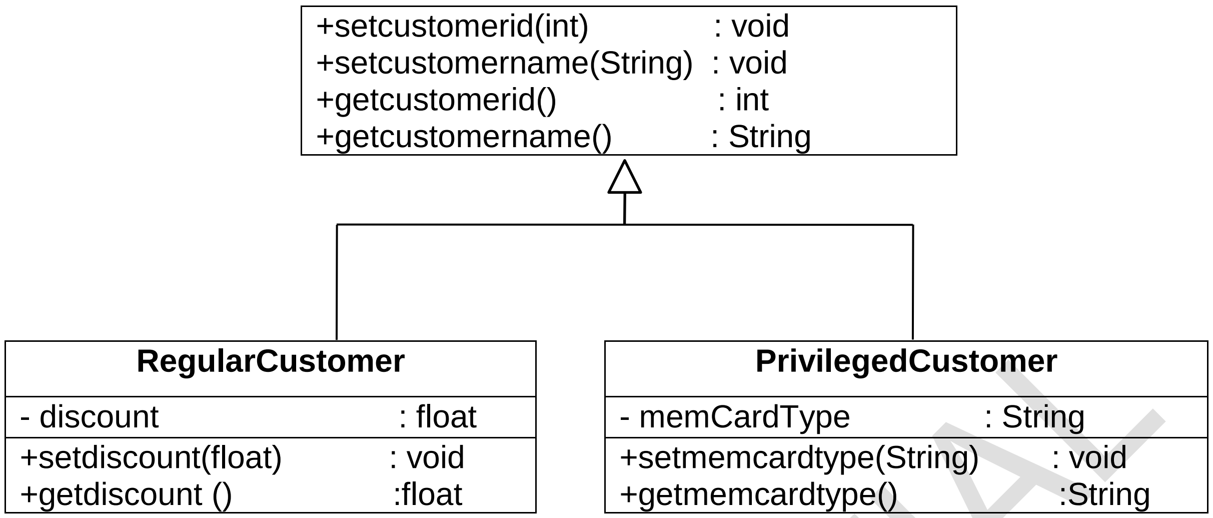
Problem Description: In the retail application, we have already seen that customers are of two types – Regular and Privileged. Regular customers are entitled for discount on each purchase and Privileged customers hold membership cards – Platinum, Gold or Silver based on which they are entitled to get gifts on each purchase.

The class diagram presented to you as part of OO Concepts Part I Assignment 5 has been modified to represent the inheritance relationship.

Note: Few instance variables and methods are not shown in the class diagram so as to keep the code simple to understand.

Class diagram:




+setcustomerid(int) : void
+setcustomername(String) : void
+getcustomerid() : int
+getcustomername() : String
RegularCustomer
- discount : float
+setdiscount(float) : void
+getdiscount () :float
PrivilegedCustomer
- memCardType : String
+setmemcardtype(String) : void
+getmemcardtype() :String

Code:

Execute the code using Eclipse IDE with the given inputs in the starter class, Retail and observe the results.

```
class Customer:
     def setcustomerid(self, id):
          self.__customerid = id

     def setcustomername(self, name):
          self.__customername=name

     def getcustomerid(self):
          return self.__customerid

     def getcustomername(self):
          return self.__customername

class RegularCustomer(Customer):
     def setdiscount(self, dis):
          self.__discount=dis

     def getdiscount(self):
          return self.__discount

class PrivilegedCustomer(Customer):
```

Note that both RegularCustomer and PrivilegedCustomer are having Customer as the parent class, hence this is an example of hierarchical inheritance

```
    def setmemcardtype(self, card):
        self.__memcardtype=card

    def getmemcardtype(self):
        return self.__memcardtype

objr = RegularCustomer()
objr.setcustomerid(1001)
objr.setcustomername('Ram')
objr.setdiscount(10.0)
print("Regular Customer Details")
print("Customer Id: ", objr.getcustomerid())
print("Customer Name: ", objr.getcustomername())
print("Discount Eligible: ", objr.getdiscount())
print("*********")
objp = PrivilegedCustomer()
objp.setcustomerid(1002)
objp.setcustomername("Seetha")
objp.setmemcardtype("Gold")
print("Regular Customer Details")
print("Customer Id: ", objp.getcustomerid())
print("Customer Name: ", objp.getcustomername())
print("Discount Eligible: ", objp.getmemcardtype())
```

Output:

```
Regular Customer Details
Customer Id:  1001
Customer Name:  Ram
Discount Eligible:  10.0
*********
Regular Customer Details
Customer Id:  1002
Customer Name:  Seetha
Discount Eligible:  Gold
```

How can the instance variables be initialized along with object creation??

Instance variables can be initialized at object creation using __init__()

Revised Code:

```
class Customer:
     def __init__(self, id=0, name=None):
          self.__customerid=id
          self.__customername=name

     def setcustomerid(self, id):
          self.__customerid = id

     def setcustomername(self, name):
          self.__customername=name

     def getcustomerid(self):
          return self.__customerid

     def getcustomername(self):
          return self.__customername

class RegularCustomer(Customer):
     def __init__(self, id=0, name=None, dis=0):
          super().__init__(id,name)
          self.__discount=dis

     def setdiscount(self, dis):
          self.__discount=dis

     def getdiscount(self):
          return self.__discount

class PrivilegedCustomer(Customer):
     def __init__(self, id=0, name=None, card=None):
          super().__init__(id,name)
          self.__memcardtype=card

     def setmemcardtype(self, card):
          self.__memcardtype=card

     def getmemcardtype(self):
          return self.__memcardtype

objr = RegularCustomer()
print("Regular Customer Details")
print("Customer Id: ", objr.getcustomerid())
print("Customer Name: ", objr.getcustomername())
print("Discount Eligible: ", objr.getdiscount())
print("*********")

objp = PrivilegedCustomer()
print("Regular Customer Details")
print("Customer Id: ", objp.getcustomerid())
print("Customer Name: ", objp.getcustomername())
print("Discount Eligible: ", objp.getmemcardtype())
print("*********")

objr = RegularCustomer()
objr.setcustomerid(1001)
objr.setcustomername('Ram')
objr.setdiscount(10.0)
print("Regular Customer Details")
print("Customer Id: ", objr.getcustomerid())
print("Customer Name: ", objr.getcustomername())
print("Discount Eligible: ", objr.getdiscount())
print("*********")

objp = PrivilegedCustomer()
objp.setcustomerid(1002)
objp.setcustomername("Seetha")
objp.setmemcardtype("Gold")
print("Regular Customer Details")
print("Customer Id: ", objp.getcustomerid())
print("Customer Name: ", objp.getcustomername())
print("Discount Eligible: ", objp.getmemcardtype())
```

Output:

```
Regular Customer Details
Customer Id:  0
Customer Name:  None
Discount Eligible:  0
*********
Regular Customer Details
Customer Id:  0
Customer Name:  None
Discount Eligible:  None
*********
```

Estimated time: 30 minutes

Summary of this assignment: In this assignment, you have understood implementation of inheritance relationship and use of super() invocation to invoke parent class __init__() using a retail application scenario.

Assignment 57: Inheritance - Quiz

Objective: Revisit inheritance concepts through a quiz

Q1. What is the output of the following code snippet?

```
class Base:
     def __init__(self):
          print("Parent init invoked")

class Derived(Base):
     def __init__(self):
          print("Derived init invoked")

obj = Derived()
```

Q2. What is the output of the following code snippet?

```
class Base:
     def __init__(self, v):
          self.baseVar = v
          self.var=0
          print("Base Class init invoked")

class Der(Base):
     def __init__(self, v):
          super().__init__(v)
          self.derVar=v
          self.var=0
          print("Derived class init invoked")

     def display(self):
          print("Base variable value: ", self.baseVar)
          print("Derived variable value: ", self.derVar)

     def useOfSuper(self):
          self.var=15
          print("Base Variable Value -> ", Base.var)
          print("Derived Variable Value -> ", self.var)

d = Der(10)
d.display()
d.useOfSuper()
```

Q3. What is the output of the following code snippet?

```
class Base:
     def __init__(self, v):
          self.baseVar = v
          print("Base Class init invoked")

class Der(Base):
     def __init__(self, v):
          self.derVar=v
          print("Derived class init invoked")

     def display(self):
          print("Base variable value: ", self.baseVar)
          print("Derived variable value: ", self.derVar)
```

```
d = Der(10)
d.display()
```

Q4. What is the output of the following code snippet?

```
class Base:
    def __init__(self, v):
        self.baseVar = v
        print("Base Class init invoked")

class Der(Base):
    def __init__(self, v):
        self.derVar=v
        print("Derived class init invoked")

    def display(self):
        print("Base variable value: ", self.baseVar)
        print("Derived variable value: ", self.derVar)

d = Der(10)
d.display()
```

Q5. Say true or false

a) Multilevel inheritance is not supported in Python
b) Inheritance can be done with the help of passing the base class to derived class in Python
c) In derived class __init__() method, 'super' should be the first statement
d) All the private data members of the base class are directly accessible in the derived class

ANSWERS

Q1.
Derived init invoked

Q2.

Runtime Error: This is because the data members of the base class can’t be overridden by the derived class as the members are added to object during initialization and not to the class. Hence, the error message “type object 'Base' has no attribute 'var'”

Q3.
Derived class init invoked
AttributeError: 'Der' object has no attribute 'baseVar'

Q4.
Runtime Error: This is because, when the object of derived class is made, there is no explicit invocation of the base class __init__() and hence the baseVar is not initialized, means that base class doesn’t have any data member. Hence, when the display is invoked triggers an error while trying to print the Base class variable, which is not available.

Q5.
a)False
b)True
c)False
d)False

Estimated time: 15 minutes

Summary of this assignment: In this assignment, you have revisited concepts of inheritance through code snippets

Assignment 58: Aggregation – Demo

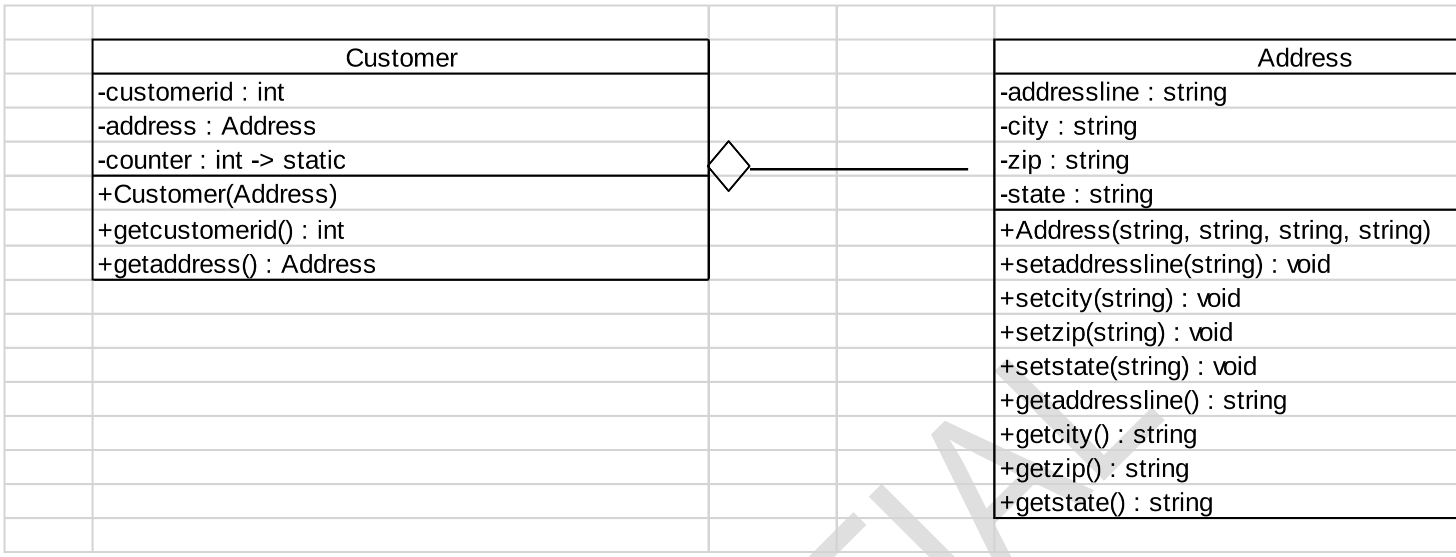
Objective: Given a class diagram for a use case representing a computational problem, implement has-a and uses-a relationships to solve the problem and test using a set of values in an IDE and recognize the benefits of the mentioned relationships

Problem Description: In the retail application, we have already seen that every customer has an address. Address consists of address line, city, state and zipcode.

The class diagram presented to you as part of aggregation (has-a) example in OO Concepts Part I Assignment 7 has been modified.

Note: Few instance variables and methods are not shown in the class diagram so as to keep the code simple to understand. Similarly, regular and privileged customer classes are also not shown here.

Class diagram:



Code:

Execute the code using Eclipse IDE with the given inputs in the starter class, Retail and observe the results.

```
class Address:
     def __init__(self, addressline, city, state, zip):
            self.__addressline = addressline
            self.__city=city
            self.__state=state
            self.__zip=zip

     def setaddressline(self, address):
          self.__addressLine = address

     def getaddressline(self):
          return self.__addressline

     def setcity(self, city):
          self.__city = city

     def getcity(self):
          return self.__city
```

```
    def setstate(self, state):
        self.__state = state

    def getstate(self):
        return self.__state

    def setzip(self, zip):
        self.__zip = zip

    def getzip(self):
        return self.__zip

class Customer:
    def __init__(self, cid, address):
        self.__customerid = cid
        self.__address = address

    def getcustomerid(self):
        return self.__customerid

    def getaddress(self):
        return self.__address

objaddress = Address("No.333,Oak street", "Strathfield", "New South
Wales", "570018")
objcust = Customer(1001, objaddress)
print("Customer Id:", objcust.getcustomerid())
address = objcust.getaddress()
print("Customer Address: ", address.getaddressline(), ",",
address.getcity(), ",", address.getstate(), ",", address.getzip())
```

Output:

```
Customer Id: 1001
Customer Address:  No.333,Oak street , Strathfield , New South Wales , 570018
```

In this example, Address reference is passed to the __init() of Customer class. Since it is a reference of an object that is passed, the parameter passing technique used is pass by referene.

Estimated time: 30 minutes

Summary of this assignment: In this assignment, you have understood the implementation of aggregation relationship using a retail application scenario.

Assignment 59: Association – Demo

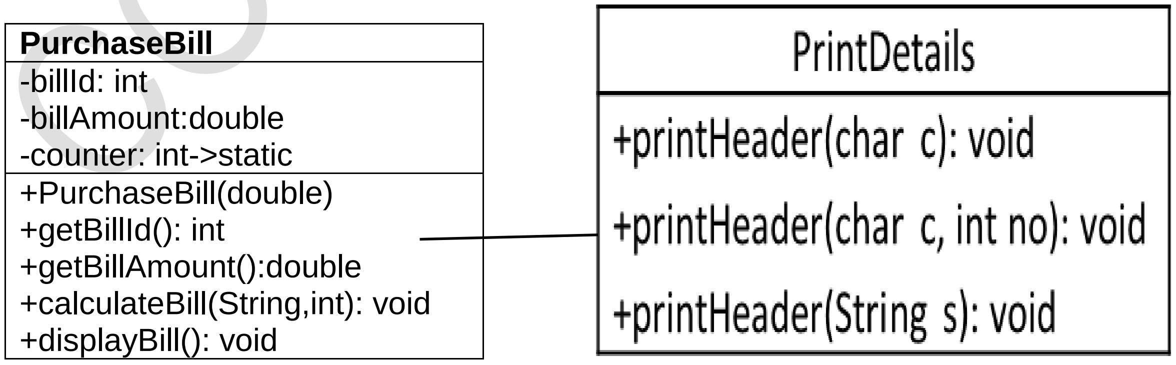
Objective: Given a class diagram for a use case representing a computational problem, implement has-a and uses-a relationships to solve the problem and test using a set of values in an IDE and recognize the benefits of the mentioned relationships

Problem Description: In the retail application, we have already seen that purchase bill has to be created in a particular format.

The class diagram presented to you as part of association (uses-a) example in OO Concepts Part I Assignment 7 has been modified to include instance variables and methods

Note: Few instance variables and methods are not shown in the class diagram so as to keep the code simple to understand.

Class diagram:



Code:

Execute the code using Eclipse IDE with the given inputs in the starter class, Retail and observe the results. Few changes are made to PurchaseBill class, the changes are highlighted in the code below:

```
class PrintDetails:
     def printheader(self, c, no=1):
          print(c*no)

class PurchaseBill:
     def __init__(self, bid, billamount):
          self.__billid = bid
          self.__billamount = billamount

     def getbillid(self):
          return self.__billid

     def getbillamount(self):
          return self.__billamount

     def calculatebill(self, mode, processcharge):
          if(mode=="Credit"):
               self.__billamount = self.__billamount +
(self.__billamount * processcharge/100)

     def displaybill(self):
          objprint = PrintDetails()
          objprint.printheader("-", 80)
          objprint.printheader("                Easy Shop Retail Store
Bill             ")
          objprint.printheader("-", 80)
          print("Bill Id: ", self.__billid)
          print("Final amount to be paid: Rs.", self.__billamount)
          objprint.printheader("-", 80)
          objprint.printheader("                     Thank You!!!
")
          objprint.printheader("-", 80)

objpur = PurchaseBill(101, 1055.0)
objpur.calculatebill("Credit", 10.5)
objpur.displaybill()
```

Execute the above code using the following values for command line arguments

Case – 1:
Arguments to be passed – “Cash” and 0

```
Case –1  Output:
--------------------------------------------------------------------------------
            Easy Shop Retail Store Bill
--------------------------------------------------------------------------------
Bill Id:  101
Final amount to be paid: Rs. 1055.0
--------------------------------------------------------------------------------
                Thank You!!!
--------------------------------------------------------------------------------
```

Case – 2:
Arguments to be passed – “Credit” and 5

```
Case – 2  Output:
--------------------------------------------------------------------------------
            Easy Shop Retail Store Bill
--------------------------------------------------------------------------------
Bill Id:  101
Final amount to be paid: Rs. 1107.75
--------------------------------------------------------------------------------
                Thank You!!!
--------------------------------------------------------------------------------
```

Estimated time: 20 minutes

Summary of this assignment: In this assignment, you have understood the implementation of association relationship using a retail application scenario.

Reference 60: Sample Code using OO Concepts

```
class customer:
    counter = 1000  #class variable
    def __init__(self, telephoneno, customername, add):
        customer.counter += 1
        self.__customerid=customer.counter
        self.__customername=customername
        self.__telephoneno=telephoneno
        self.__address = add

    def setcustomerid(self, cid):
        self.__customerid = cid

    def settelephoneno(self, teleno):
        self.__telephoneno = teleno

    def getcustomerid(self):
        return self.__customerid

    def gettelephoneno(self):
        return self.__telephoneno

    def getcustomername(self):
        return self.__customername

    def getaddress(self):
        return self.__address

    @staticmethod
    def gettotalcustomer():
        return customer.counter-1000

class regularcustomer(customer):
    def __init__(self, telephoneno, customername, discount, add):
        # super to invoke baseclass init
        super().__init__(telephoneno, customername, add)
        self.__discount = discount

    def setdiscount(self, dis):
        self.__discount = dis

    def getdiscount(self):
        return self.__discount

class address:
    def __init__(self, add):
        self.__addressline = add

    def setaddress(self, add):
        self.__addressline = add

    def getaddress(self):
        return self.__addressline

regcustadd1 = address("No.22,Vijay Nagar Mysore Karnataka  570018")

teleno=[9201861311, 9201861321, 9201661311]
regcustobj1 = regularcustomer(teleno, "John", 12.5, regcustadd1)
#custobj1.setcustomerid(1001)
#custobj1.settelephoneno(1234567890)
print("Customer id:", regcustobj1.getcustomerid())
print("Telephone no:", regcustobj1.gettelephoneno())
print("Customer name:", regcustobj1.getcustomername())
print("Discount:", regcustobj1.getdiscount())
#temp1 = regcustobj1.getaddress().getaddress()
print("Customer's address:", regcustobj1.getaddress().getaddress())

print("\n")
regcustadd2 = address("No.33,J.P. Nagar Bangalore Karnataka  570011")
teleno1 = [1122334455, 1199887766, 2244668897]
regcustobj2 = regularcustomer(teleno1, "Mary", 15.5,regcustadd2)
print("Customer id:", regcustobj2.getcustomerid())
print("Telephone no:", regcustobj2.gettelephoneno())
print("Customer name:", regcustobj2.getcustomername())
print("Discount:", regcustobj2.getdiscount())
print("Customer's address:", regcustobj2.getaddress().getaddress())
print("\n")
print("Total customers registered:", customer.gettotalcustomer())
```

Output:

```
Customer id: 1001
Telephone no: [9201861311, 9201861321, 9201661311]
Customer name: John
Discount: 12.5
Customer's address: No.22,Vijay Nagar Mysore Karnataka  570018


Customer id: 1002
Telephone no: [1122334455, 1199887766, 2244668897]
Customer name: Mary
Discount: 15.5
Customer's address: No.33,J.P. Nagar Bangalore Karnataka  570011

Total customers registered: 2
```