LAB # 01

Statement Purpose:

Objective of this lab is to make students understand the difference between object oriented and procedural approaches to programming

Activity Outcomes:

The student will understand the advantages of using OOP

The student will understand the difference between procedural and object oriented approaches

Instructor Note:

The Students should have knowledge about structured programming.

1) Stage J (Journey)

Introduction

Procedural programming uses a list of instructions to tell the computer what to do step-by-step. Procedural programming relies on procedures, also known as routines or subroutines. A procedure contains a series of computational steps to be carried out.
Procedural programming is intuitive in the sense that it is very similar to how you would expect a program to work. If you want a computer to do something, you should provide step-by-step instructions on how to do it. It is, therefore, no surprise that most of the early programming languages are all procedural. Examples of procedural languages include Fortran, COBOL and C, which have been around since the 1960s and 70s.

Object-oriented programming, or OOP, is an approach to problem-solving where all computations are carried out using objects. An object is a component of a program that knows how to perform certain actions and how to interact with other elements of the program. Objects are the basic units of object-oriented programming. A simple example of an object would be a person. Logically, you would expect a person to have a name. This would be considered a property of the person. You would also expect a person to be able to do something, such as walking. This would be considered a method of the person.
A method in object-oriented programming is like a procedure in procedural programming. The key difference here is that the method is part of an object. In object-oriented programming, you organize your code by creating objects, and then you can give those objects properties and you can make them do certain things.

One of the most important characteristics of procedural programming is that it relies on procedures that operate on data - these are two separate concepts. In object-oriented programming, these two concepts are bundled into objects. This makes it possible to create more complicated behavior with less code. The use of objects also makes it possible to reuse code. Once you have created an object with more complex behavior, you can use it anywhere in your code.

2) Stage a1 (apply)

Lab Activities:

Activity 1:

The example demonstrates the difference in approach if we want to find the circumference of circle.

Solution:

Procedural Approach	**Object Oriented Approach**

Public class **Circle**{
int radius;
Public void setRadius(int r)
{ radius = r;}
Public void showCircumference()
{
double c = 2*3.14*radius;
System.out.println(“Circumferenceis”+ c);
}
Public static void main()
{
setRadius(5);
showCircumference();
//output would be 31.4
setRadius(10);
showCircumference();
// output would be 62.8
}
}
	Public class **Circle**{
	Private int radius;
	Public void setRadius(int r)
	{ radius = r;}
	Public void showCircumference()
	{
	double c = 2*3.14* radius;
	System.out.println(“Circumference is”+ c);
	}
	}
	Public class **runner**
	{
	Public static void main()
	{
	Circle c1= new circle();
	c1.setRadius(5);
	c1.showCircumference();
	//output would be 31.4; it belongs to c1
	Circle c2= new circle();
	c2.setRadius(10);
	c2.showCircumference();
	//output would be 62.8; it belongs to c2
	}
	}

Activity 2:

The example demonstrates the difference in approach if we want to model the concept of a Book. In object Oriented approach the concept can be defined once and then reused in form of different objects.

Procedural Approach	**Object Oriented Approach**

Public class **Book**{
string title;
double price;
int noOfPages;
 	
Public void setTitle(string t)
{ title = t;}
Public void setPrice (double p)
{ price = p;}
Public void setNoOfPages (int n)
{ noOfPages = n;}
	Public class **Book**{
	Private string title;
	Private double price;
	Private int noOfPages;
	
	Public void setTitle(string t)
	{ title = t;}
	Public void setPrice (double p)
	{ price = p;}
	Public void setNoOfPages (int n)
	{ noOfPages = n;}

Public void display()
{
System.out.println(“BookTitle”+ title + “ BookPrice “ + price + “BookPages” + noOfPages);
}
 	
Public static void main()
{
setTitle (“OOP”);
setPrice (200);
setNoOfPages (500);
display();
}
 	
}
	Public void display()
	{
	System.out.println(“BookTitle”+ title + “ BookPrice “ + price + “BookPages” + noOfPages);
	}
	}
	
	Public class **runner**
	{
	Public static void main()
	{
	Book b1= new Book();
	b1.setTitle (“OOP”);
	b1.setPrice (200);
	b1.setNoOfPages (“OOP”);
	b1.display();
	//output belongs to b1
	
	Book b2= new Book();
	b2.setTitle (“ICP”);
	b2..setPrice (150);
	b2.setNoOfPages (350);
	b2.display();
	
	//output belongs to b2
	}
	}

Activity 3:

The example demonstrates the difference in approach if we want to model the concept of an Account. Again we can see that in object Oriented approach the concept can be defined once and then reused uniquely by different objects.

Procedural Approach	**Object Oriented Approach**

Public class **Account**{
double balance;
Public void setBalance(double b)
{ balance = b;}
Public void showBalance()
{
System.out.println(“Balance is”+ balance);
}
Public static void main()
{
setBalance (5000);
showBalance (); // output would be 5000
}
}
	Public class **Account**{
	double balance;
	Public void setBalance(double b)
	{ balance = b;}
	Public void showBalance()
	{
	System.out.println(“Balance is”+ balance);
	}
	}
	Public class **runner**
	{
	Public static void main()
	{

	Account a1= new Account ();
	a1.setBalance(2500);
	a1.showBalance();
	//output would be2500; it belongs to a1
	
	Account a2= new Account ();
	a2.setBalance(5000);
	a2.showBalance();
	//output would be 5000; it belongs to a2
	
	}
	}

3) Stage v (verify)

Home Activities:

Activity 1:

Modify the last activity and include functions of withdraw and deposit.
Test these methods in main for procedural approach. For Object Oriented approach, modify the runner class and call withdraw and deposit functions for two objects.

Activity 2:

Write a program that has variables to store Car data like; CarModel, CarName, CarPrice and CarOwner. The program should include functions to assign user defined values to the above mentioned variable and a display function to show the values . Write a main that calls these functions

Now write another runner class that declares three Car objects and displays the data of all three.

.

4) Stage a2 (assess)

Assignment:

Write a program that contains variables to hold employee data like; employeeCode, employeeName and date Of Joining. Write a function that assigns the user defined values to these variables. Write another function that asks the user to enter current date and then checks if the employee tenure is more than three years or not. Call the functions in main.

Now write a runner class that declares two employee objects and check their tenure periods.