OOP Labsheet-4

Prof.R.Gururaj

Java Command-line arguments:

What is a Java command-line argument?

How is it passed on to Java program.

What is type of argument, and how do we pass arguments.

How do we pass integers as arguments.

Use of *Integer.parseInt()*

Practice Problem-1

Write a program that takes two integers as command-line arguments and print the sum.

Practice Problem-2

Write a program that takes some command-line arguments and – (i) print the number of args and (ii) print them in the same order.

Java Constructors

Purpose: To initialize java objects when they are created. They have no return type.

- Constructor is automatically called after object is created and new operator completes.
- A default constructor is created by java, instance variables will be initialized to zero or null.
- If we explicitly define a constructor the default constructor is no more available.

Practice Problem-3

Write a class without user defined constructor (having only default constructor)

```
class Box
{
   int length, width, height;
   void printBoxDetails()
   {          System.out.println(“Box length is : "+length);
         System.out.println(“Box width is : "+width);
         System.out.println(“Box height is : "+height);
    }
}
class Demo
{
         public static void main(String args[])
          {
                  Box b=new Box();  // using default constructor
                        b.printBoxDetails();
         }
   }
```

Note: when no user defined constructor exists, a default constructor will do the job. Observe what are the default values if default constructor does initialization.

Practice problem-4

Define a class Box with three attributes length, breadth, and height.

Define a no-argument constructor that initializes length, breadth, and height with 10, 5, and 2 respectively.

Also define a three-argument constructor that initializes length, breadth, and height with the arguments passed on.

(Constructor Overloading)

Write a class with *public static void main()*, that will use these constructors

```
class Box
{
  int length, width, height;
  void printBoxDetails()
  {         System.out.println(“Box length is : "+length);
        System.out.println(“Box width is : "+width);
        System.out.println(“Box height is : "+height);
   }
// constructor overloading
   Box(int x, int y, int z){length=x;, width=y, height=z;}
   Box(){length=10;, width=5, height=2;}
}
class Test
{
        public static void main(String args[])
         {       Box b1=new Box();  b1.printBoxDetails();
                 Box b2=new Box(4,7,9);  b2.printBoxDetails();
        }
  }
```

Adding methods to a class.

Practice problem-5

Now for the class Box with three attributes length, breadth, and height, with three args constructor to initialize these attributes (as already defined earlier) and add methods- as follows.

1.A method *volume()* -to compute the volume and return it.

2. A method *setDimensions()* - that takes three arguments and sets the length, breadth, and height with the new values passed on.

Write a class with public static void main(), that will use these methods.

Method Overloading-

What is method overloading?

Practice Problem-6

Now define the class Shape with a method *computeArea()* – having two versions (overloaded) as follows.

1. First one will have one argument and compute the area and return it as square of the value.
2. The second one will have two arguments and compute the area and return it as product of two values.

Write a class Test with public static void main(), that will use these methods.

Static members of a class.

What is a static variable?

What is a static method?

Practice problem-7

Write a class *Test* with public static void main().

The class will also have a method *show()* that will print the message "Hello World".

Now call the show() in *p s v main() . // what will happen?*

Declare an *int* variable *item* in class Test, assign value 20, and print the same in *p s v main(). // what will happen?*

Why it is so?

We understand how this static keyword works with members of a class.

Inheritance

What is inheritance.

What is the use.

What is an abstract class and abstract method.

What is method overriding.

Exercise:

Write a class Shape with a method *computeArea*() that prints just the string "Area of shape".
Now declare class Triangle with int attributes-length, height, and class Square with attribute-int side) as sub classes of Shape.

Override the method *computeArea* () in subclasses to compute and print the area accordingly.

Write a class with *public static void main(),* that will validate the above.

Do the following.

Declare an object reference of Shape and assign a shape object and call compute area.

Declare an object reference of Shape and assign an object of type Triangle and call computeArea.

Declare an object reference of Shape and assign an object of type Square and call computeArea.

Declare an object reference of Square and assign an object of type Shape and call computeArea.

Carefully observe the outcome of above experiments.

****