|  |  |
| --- | --- |
| Course- BTech | Type- Core |
| Course Code- **CSET** | Course Name- **Object Oriented Programming Using Java** |
| Year- First | Semester- Even Batch- BTech 2nd Semester |

# Tutorial-8

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tutorial No.** | **Name** | **CO1** | **CO2** | **CO3** |
| **8** | **Basics** |  |  | **--** |

**Objective:** The main objective of this tutorial is to learn about the basics of Java language.

* 1. Complete the below code for storing two Student record.

abstract class AbstractClass {  
 protected abstract int abstractMethod();  
}  
class test extends AbstractClass {  
 protected int abstractMethod(){  
 return 1;  
 }  
 public static void main(String[] args) {  
 test t = new test();  
 System.*out*.println(t.abstractMethod());  
 }  
}

# Output: 1

* 1. Complete the code for given output

\_ \_ \_ \_class Test {

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_;

}

public class MainClass extends Test{

public static void main(String[] args) {

\_ \_ \_ \_ \_ t=new ();

int[] ans = t.getSumAndSub(25, 55);

System.out.println("Sum = " + ans[0]);

System.out.println("Sub = " + ans[1]);

}

@Override

int[] getSumAndSub(int a, int b) {

int[] ans = new int[2];

ans[0] = a + b;

ans[1] = a - b; return ;

}

}

# Output:

# Sum= 80

**Sum= -30**

**class Test {**

**// Declare an abstract method**

**abstract int[] getSumAndSub(int a, int b);**

**}**

**public class MainClass extends Test {**

**public static void main(String[] args) {**

**MainClass t = new MainClass();**

**int[] ans = t.getSumAndSub(25, 55);**

**System.out.println("Sum = " + ans[0]);**

**System.out.println("Sub = " + ans[1]);**

**}**

**@Override**

**int[] getSumAndSub(int a, int b) {**

**int[] ans = new int[2];**

**ans[0] = a + b;**

**ans[1] = a - b;**

**return ans;**

**}**

**}**

* 1. What will be the output of the following program?

public class test extends Product {  
 test(int pid, String n) {  
 super(pid, n);  
 }  
 public void display() {  
 System.*out*.print("Product Id = "+pro\_Id + " " + " Product Name = "+pro\_name);  
 }  
 public static void main(String args[])  
 {  
 test[] obj = new test[5] ;  
 obj[0] = new test(23907,"Dell Laptop");  
 System.*out*.println("Product Object:");  
 obj[0].display();  
 }  
}  
abstract class Product {  
 int pro\_Id;  
 String pro\_name;  
 Product(int pid, String n) {  
 pro\_Id = pid; pro\_name = n;  
 }  
 abstract void display();  
}

**Output: Product Object:**

**Product Id = 23907 Product Name = Dell Laptop**

* 1. What is the output of following code?

abstract class A {  
 abstract void myMethod(Number N);  
}  
interface B {  
 abstract void myMethod(Object O);  
}  
class C extends A implements B {  
 void myMethod(Number N) {  
 System.*out*.println("Number");  
 }  
 public void myMethod(Object O) {  
 System.*out*.println("Object");  
 }  
}  
public class test {  
 public static void main(String[] args) {  
 A a = new C();  
 a.myMethod(new Integer(121));  
 B b = new C();  
 b.myMethod(new Integer(121));  
 C c = new C();  
 c.myMethod(new Integer(121));  
 }  
}

**Output:**

**Number**

**Object**

**Number**

* 1. What would be the result of the following code?

abstract class A {  
 int i = 111, j = 222;  
 abstract void methodOne();  
 abstract void methodTwo();  
}  
abstract class B extends A {  
 @Override  
 void methodOne() {  
 System.*out*.println(i);  
 System.*out*.println(j); i = ++i;  
 j = --j;  
 }  
}  
class C extends B {  
 @Override  
 void methodTwo() {  
 System.*out*.println(i); System.*out*.println(j); i = i++;  
 j = j--;  
 }  
}  
public class test {  
 public static void main(String[] args) {  
 C c = new C();  
 c.methodOne();  
 c.methodTwo();  
 System.*out*.println(c.i);  
 System.*out*.println(c.j);  
 }  
}

**Output:**

**111**

**222**

**112**

**221**

**112**

**221**

* 1. Will the code run successfully?

interface A {  
 int *i* = 111;  
}  
class B implements A {  
 void methodB() {  
 *i* = 222;  
 }  
}

**OUTPUT:**

**error: cannot assign a value to final variable i**

**i = 222;**

* 1. What will be the Output of the below code:

abstract class X {  
 int i = 111;   
 int methodX() {  
 return methodX(i);  
 }  
 abstract int methodX(int i);  
}  
class Y extends X {  
 @Override  
 int methodX(int i) {  
 return ++i + i++;  
 }  
}  
public class test {  
 public static void main(String[] args) {  
 Y y = new Y();   
 System.*out*.println(y.methodX());  
 }  
}

**OUTPUT:**

**224**

* 1. Is the below code correct?

class X {  
 abstract class Y {  
 class Z {  
 }  
 }  
}  
public class test{  
 public static void main(String[] args) {  
 X x=new X();  
 }  
}

**Output:**

**Important link -   
https://www.geeksforgeeks.org/overriding-in-java/**[**https://www.geeksforgeeks.org/method-overloading-in-java/**](https://www.geeksforgeeks.org/method-overloading-in-java/)

[**https://www.geeksforgeeks.org/difference-between-method-overloading-and-method-overriding-in-java/**](https://www.geeksforgeeks.org/difference-between-method-overloading-and-method-overriding-in-java/)