

Basic of OOPS Assignment

Submitted by:

Rishika Kumari

1852

1. Write a Java Program to Copy the values from one object to another Object.

Source Code:

```
package com.knolduscopyingoneobjecttoanother;
public class CopyingOneObjectToAnother {

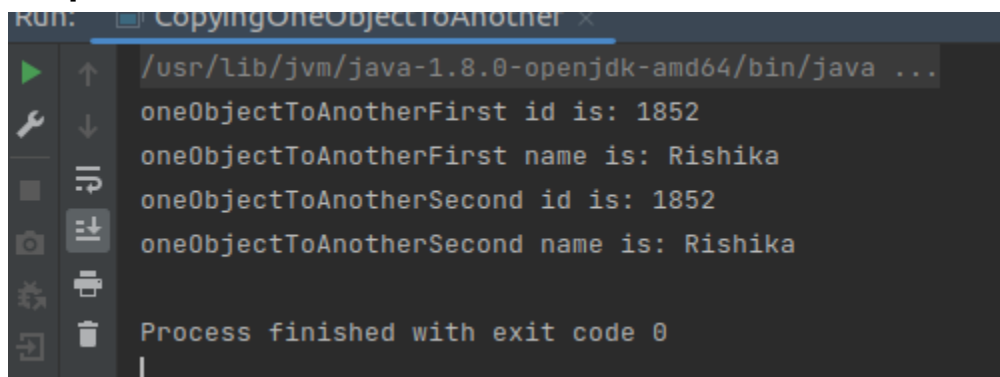
    int id;
    String name;
    CopyingOneObjectToAnother(int id,String name){
        this.id = id;
        this.name = name;
    }
    CopyingOneObjectToAnother(CopyingOneObjectToAnother oneObjectToAnother) {
        this.id = oneObjectToAnother.id;
        this.name = oneObjectToAnother.name;
    }
    public static void main(String[] args) {

        CopyingOneObjectToAnother oneObjectToAnotherFirst = new
        CopyingOneObjectToAnother(1852,"Rishika");
        System.out.println("oneObjectToAnotherFirst id is:
        "+oneObjectToAnotherFirst.id);
        System.out.println("oneObjectToAnotherFirst name is:
        "+oneObjectToAnotherFirst.name);

        CopyingOneObjectToAnother oneObjectToAnotherSecond = new
        CopyingOneObjectToAnother(oneObjectToAnotherFirst);
        System.out.println("oneObjectToAnotherSecond id is:
        "+oneObjectToAnotherSecond.id);
        System.out.println("oneObjectToAnotherSecond name is:
        "+oneObjectToAnotherSecond.name);

    }
}
```

Output:



```
Run: CopyingOneObjectToAnother x
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...
oneObjectToAnotherFirst id is: 1852
oneObjectToAnotherFirst name is: Rishika
oneObjectToAnotherSecond id is: 1852
oneObjectToAnotherSecond name is: Rishika
Process finished with exit code 0
```

2. Create a class named 'Member' having the following members:

Data members

- 1 - Name
- 2 - Age
- 3 - Phone number
- 4 - Address
- 5 - Salary

It also has a method named 'printSalary' which prints the salary of the members.

Two classes 'Employee' and 'Manager' inherits the 'Member' class. The 'Employee' and 'Manager' classes have data members 'specialization' and 'department' respectively. Now, assign a name, age, phone number, address, and salary to an employee and a manager by making an object of both of these classes and print the same.

Source Code:

```
package com.knoldusmemberemployeemanager;

public class Member {

    private String name;
    private int age;
    private String phoneNumber;
    private String address;
    private double salary;
    public Member(String name,int age,String phoneNumber,String address, double
salary)
    {

        this.name = name;
        this.age = age;
        this.phoneNumber = phoneNumber;
        this.address = address;
        this.salary = salary;
    }
    public void printsalary() {
        System.out.println("Salary is:Rs. "+salary);
    }
    public void printDetails() {
        System.out.println("Name is: "+name);
        System.out.println("Age is: "+age);
        System.out.println("Phone Number is: "+phoneNumber);
        System.out.println("Address is: "+address);
    }
}

class Employee extends Member {

    private String specialization;
```

```

    Employee(String name, int age, String phoneNumber, String address, double
salary, String specialization) {
        super(name,age,phoneNumber,address,salary);
        this.specialization = specialization;
    }
    public void printDetails() {
        System.out.println("\nEmployee details: ");
        super.printDetails();
        System.out.println("Specialization is : "+specialization);
    }
}
class Manager extends Member {
    private String department;
    Manager(String name, int age, String phoneNumber, String address, double
salary, String department) {
        super(name,age,phoneNumber,address,salary);
        this.department = department;
    }
    public void printDetails() {
        System.out.println("\nManager details: ");
        super.printDetails();
        System.out.println("Department is : "+department);
    }
}
}

```

Main.java

```

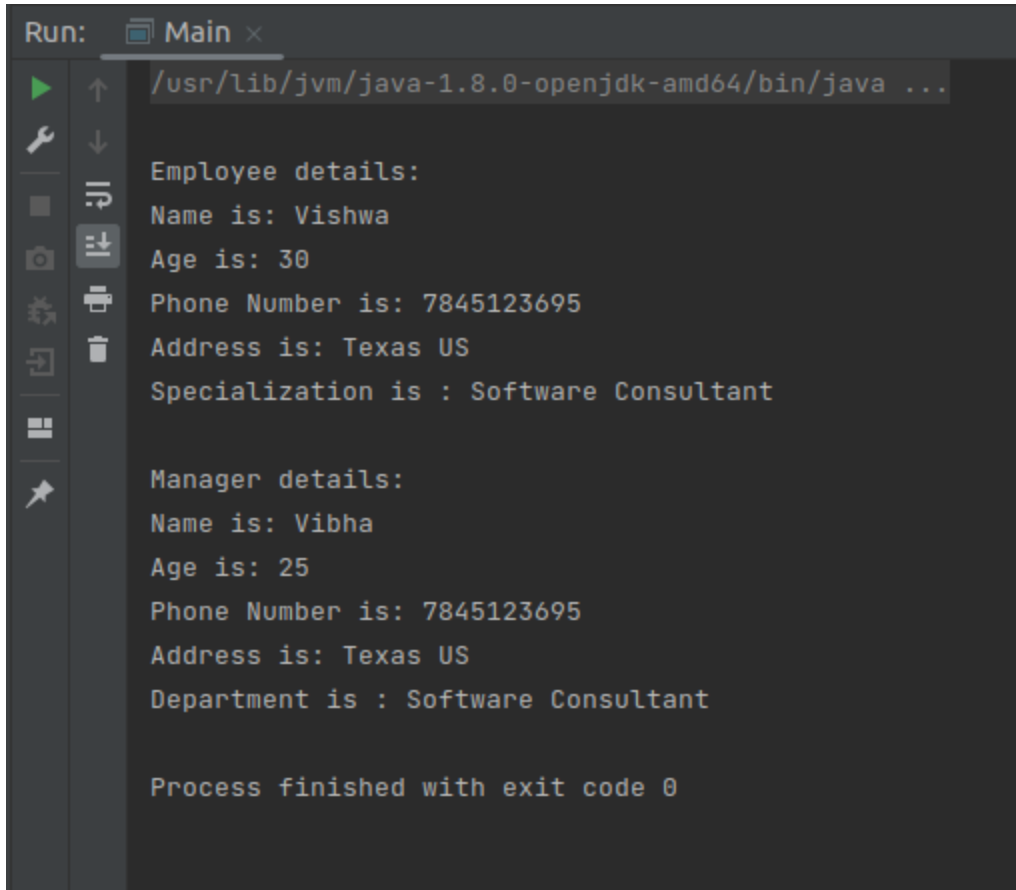
package com.knoldusmemberemployeemanager;

public class Main {
    public static void main(String[] args) {
        Employee firstEmployee = new Employee("Vishwa",30,"7845123695","Texas
US",70000,"Software Consultant");
        firstEmployee.printDetails();

        Manager firstManager = new Manager("Vibha",25,"7845123695","Texas
US",70000,"Software Consultant");
        firstManager.printDetails();
    }
}

```

Output:



```
Run: Main x
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...

Employee details:
Name is: Vishwa
Age is: 30
Phone Number is: 7845123695
Address is: Texas US
Specialization is : Software Consultant

Manager details:
Name is: Vibha
Age is: 25
Phone Number is: 7845123695
Address is: Texas US
Department is : Software Consultant

Process finished with exit code 0
```

3. Create a class 'Degree' having a method 'getDegree' that prints "I got a degree". It has two subclasses namely 'Undergraduate' and 'Postgraduate' each having a method with the same name that prints "I am an Undergraduate" and "I am a Postgraduate" respectively. Call the method by creating an object of each of the three classes.

Source Code:

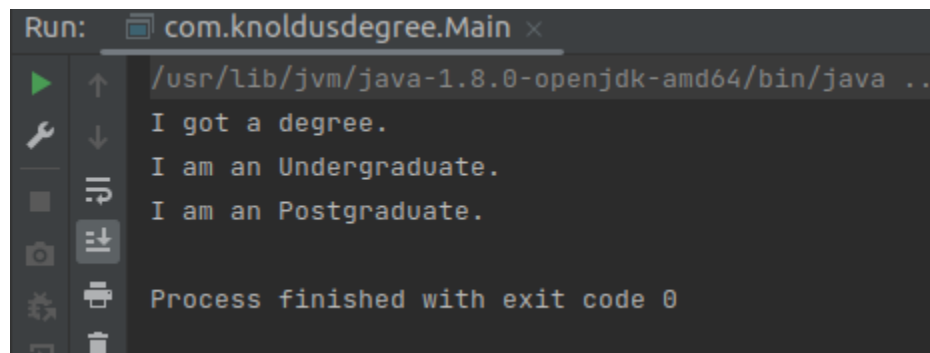
```
package com.knoldusdegree;

public class Degree {
    public void getDegree(){
        System.out.println("I got a degree.");
    }
}

class Undergraduate extends Degree{
    public void getDegree(){
        System.out.println("I am an Undergraduate.");
    }
}
```

```
}  
class Postgraduate extends Degree{  
    public void getDegree(){  
        System.out.println("I am an Postgraduate.");  
    }  
}  
class Main{  
    public static void main(String[] args) {  
  
        Degree degree = new Degree();  
        degree.getDegree();  
  
        Undergraduate undergraduateDegree = new Undergraduate();  
        undergraduateDegree.getDegree();  
  
        Postgraduate postgraduateDegree = new Postgraduate();  
        postgraduateDegree.getDegree();  
    }  
}
```

Output:



The screenshot shows a Java IDE's Run console window. The title bar reads "Run: com.knoldusdegree.Main x". The console output is as follows:

```
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ..  
I got a degree.  
I am an Undergraduate.  
I am an Postgraduate.  
Process finished with exit code 0
```

4. What will be the output of the following program?

```
class A {  
  
}  
  
class B extends A {  
  
}  
  
class C extends B {  
  
}  
  
public class MainClass {  
  
    static void overloadedMethod(A a)  
    {  
        System.out.println("ONE");  
    }  
  
    static void overloadedMethod(B b)  
    {  
        System.out.println("TWO");  
    }  
  
    static void overloadedMethod(Object obj)  
    {  
        System.out.println("THREE");  
    }  
  
    public static void main(String[] args)  
    {  
        C c = new C();  
  
        overloadedMethod(c);  
    }  
}
```

Output:
TWO

5. In the below class, is 'method' overloaded or duplicated?

```
public class MainClass {  
  
    void method(int ... a)  
    {  
        System.out.println(1);  
    }  
  
    void method(int[] a)  
    {  
        System.out.println(2);  
    }  
}
```

Output:

Duplicated. Because var args (int ... a) are nothing but the arrays. So here, (int ... a) and (int[] a) are the same.

6. What will be the outcome of the below program?

```
public class MainClass {  
  
    double overloadedMethod(double d)  
    {  
        return d *= d;  
    }  
  
    int overloadedMethod(int i)  
    {  
        return overloadedMethod(i *= i);  
    }  
  
    float overloadedMethod(float f)  
    {  
        return overloadedMethod(f *= f);  
    }  
  
    public static void main(String[] args)  
    {  
        MainClass main = new MainClass();  
  
        System.out.println(main.overloadedMethod(100));  
    }  
}
```

Output:

It will throw java.lang.StackOverflowError at run time. Because, overloadedMethod(int) keeps calling itself..

Error:-- thread "main" java.lang.StackOverflowError

7. What is the output of the following program?

```
class Test {  
    void myMethod()  
    {  
        System.out.println("Gaurav");  
    }  
}  
  
public class Derived extends Test {  
  
    void myMethod()  
    {  
        System.out.println("GFG");  
    }  
  
    public static void main(String[] args)  
    {  
        Derived object = new Test();  
        object.myMethod();  
    }  
}
```

Output:

Compilation error(In Java, we cannot assign a parent class reference object to the child class, but if we perform downcasting, we will not get any compile-time error.)

8. What is the output of the following program?

```
interface GFG {  
  
    void myMethod();  
    void getInfo();  
}  
  
abstract class Gaurav implements GFG {  
  
    void getData()  
    {  
        System.out.println("GFG");  
    }  
}  
  
public class Test extends Gaurav {  
  
    public void myMethod()  
    {  
        System.out.println("Gaurav");  
    }  
}  
  
    public void getInfo()  
    {  
        System.out.println("Geek");  
    }  
  
    public static void main(String[] args)  
    {  
        Gaurav obj = new Test();  
        obj.getInfo();  
    }  
}
```

Output:

Geek

9. What is the output of the following program?

```
// Override of static method
class Parent {

    // static method
    static void show()
    {
        System.out.println("Parent");
    }
}

// Parent inherit in Child class
class Child extends Parent {

    // override show() of Parent
    void show()
    {
        System.out.println("Child");
    }
}

class GFG {
    public static void main(String[] args)
    {
        Parent p = new Parent();
        // calling Parent's show()
        p.show();
        // cannot override Parent's show()
    }
}
```

Output:

The instance method 'void show ' in 'child' cannot override a static parent class method. It is beyond the concept of oops.

```
public class MyClass {  
    private int x = 10;  
    static int m1() {  
        int y = x;  
        return y;  
    }  
    public static void main(String[] args) {  
        m1();  
    }  
}
```

Non static member “X” cannot be referenced from static context.

```
public class StaticDemo {

    static String n1= examName("O");{
        n1=examName("A");
    }

    static{
        n1=examName("C");
    }

    public static void main(String[] args) {
        StaticDemo sd = new StaticDemo();

    }

    public static String examName(String s){
        System.out.println(s);
        return s;
    }

}
```

The outcome will be: O
C
A