**Q.1 Write a program to implement “this” keyword.**

class TriangleDemo

{

int base,height;

TriangleDemo(int base,int height)

{

System.out.println("The value of Base in constructor= "+base);

System.out.println("The value of Base from global variable base= "+this.base);

this.base=base;

this.height=height;

}

public void show()

{

System.out.println(" Base= "+base+"\n Height= "+height+"\n Area of triangle = "+(0.5\*base\*height));

}

}

class ThisDemo

{

public static void main(String args[])

{

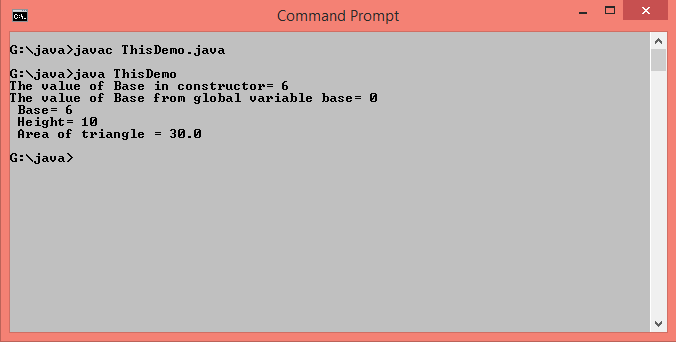
TriangleDemo t=new TriangleDemo(6,10);

t.show();

}

}

**Output :-**

****

**Q.2 Write a program to implement Accessors and Mutator methods for class “Student”.**

class Student{

private int PRN;

public int getPRN() {

return PRN;

}

public void setPRN(int PRN){

this.PRN=PRN;

}

public static void main(String args[]){

Student s=new Student();

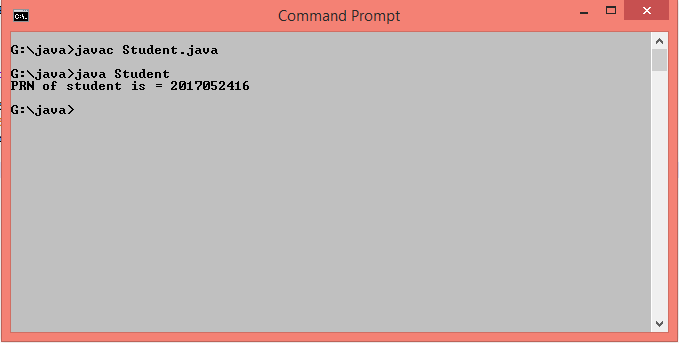
s.setPRN(2017052416);

System.out.println("PRN of student is = "+s.getPRN());

}

}

**Output :-**



**Q.3 Write a program to implement 2 methods of cloning Objects for class “Employee”.**

**//Shallow copy**

class Department

{

String dept1;

String dept2;

public Department(String d1, String d2)

{

this.dept1 = d1;

this.dept2 = d2;

}

}

class Employee2 implements Cloneable

{

int id;

String name;

Department dept;

Employee2(int id,String name,Department dept)

{

this.id=id;

this.name=name;

this.dept=dept;

}

public Object clone()throws CloneNotSupportedException

{

return super.clone();

}

public static void main(String args[])

{

Department d=new Department("Finance","HR");

Employee2 e1=new Employee2(75,"Dhiraj",d);

Employee2 e2=null;

System.out.println("\nBefore clonning");

System.out.println(e1.id+" "+e1.name+" "+e1.dept.dept1);

try

{

e2=(Employee2)e1.clone();

System.out.println(e2.id+" "+e2.name+" "+e2.dept.dept2);

}

catch (CloneNotSupportedException e)

{

e.printStackTrace();

}

e2.dept.dept2="Marketting";

System.out.println("\nAfter clonning");

System.out.println(e1.id+" "+e1.name+" "+e1.dept.dept2);

}

}

**Output :-**

****

**//Deep Copy**

class Department implements Cloneable

{

String dept1;

String dept2;

public Department(String d1, String d2)

{

this.dept1 = d1;

this.dept2 = d2;

}

public Object clone()throws CloneNotSupportedException

{

return super.clone();

}

}

class Employee3 implements Cloneable

{

int id;

String name;

Department dept;

Employee3(int id,String name,Department dept)

{

this.id=id;

this.name=name;

this.dept=dept;

}

public Object clone()throws CloneNotSupportedException

{

Employee3 e = (Employee3) super.clone();

e.dept = (Department) dept.clone();

return e ;

}

public static void main(String args[])

{

Department d=new Department("Finance","HR");

Employee3 e1=new Employee3(75,"Dhiraj",d);

Employee3 e2=null;

System.out.println("\nBefore clonning");

System.out.println(e1.id+" "+e1.name+" "+e1.dept.dept1);

try

{

e2=(Employee3)e1.clone();

System.out.println(e2.id+" "+e2.name+" "+e2.dept.dept2);

}

catch (CloneNotSupportedException e)

{

e.printStackTrace();

}

e2.dept.dept2="Marketting";

System.out.println("\nAfter clonning");

System.out.println(e1.id+" "+e1.name+" "+e1.dept.dept2);

}

}

**Output :-**

****

**Q.4 Write a program to implement Generic class.**

class People<T>

{

private T value;

public People(T value)

{

this.value=value;

}

public T getData()

{

return this.value;

}

}

class GenericClassDemo

{

public static void main(String args[])

{

People<String> p1=new People<>("Santosh Kadam");

System.out.println("Name of Person : "+p1.getData());

People<Integer> p2=new People<>(40);

System.out.println("Age of Person : "+p2.getData());

People<Double> p3=new People<>(20000.00);

System.out.println("Salary of Person : "+p3.getData());

}

}

**Output :-**

