Assignment-3

1. Write a example program to show why Java does not support multiple inheritance and how to resolve that issue (with seperate program).
2. write a program to print

```
   *
  * *
 * * *
* * * *
```

3. Program to get use of abstraction and interface.
4. program for parameterised constructor. How it helps in the executions of any program (little bit of conceptual)
5. Java code to implement
   a. static variable  b. instance variable  c. local variable.

① 
```
class A1 {
  int height;
  String name;
  void Work() {
     System.out.println("Working");
  }
}
class A2 {
  int marks;
  void study() {
     System.out.println("studying");
  }
}
class A3 extends A1, A2  //not possible
```

As A3 class does not extend A1, A2 classes multiple inheritance is not supported so we use interface to overcome this.

```
* Interface A {
     void wakeup() {
        System.out.println("wakeup");
     }
     void getReady() {
        System.out.println("getting Ready");
     }
  }
```

```
class B implements A {
    public void walk() {
        System.out.println("walking");
    }
    void run() {
        System.out.println("running");
    }
}
public class Multiple {
    public static void main(String args[]) {
    B b = new B();
    b.walk();
    b.run();
    b.wakeup();
    b.getready();
    A a = new a();
    a.wakeup();
    a.getready();
    a.walk();
    }
}
```

(8)

```
    *
  * * *
* * * * *
```

```
for (int i=0; i<=5; i++)
{
for (int j=1; j<=i; j++)
{
```

③

```
public class AbstractAInterface {
    public static void main (String args[]) {
        B b = new B();
        b.A1();
        b.A2();
        b.A3();
        A a = new A();
        a.A1();
        a.A3();
        a.A2(); // method doesn't exist
    }
}
interface A {
    void A1();
    abstract void A3();
}
class B implements A {
    public void A1() {
        System.out.println("A1 method");
    }
    public void A3() {
        System.out.println("A3 method");
    }
    void A2() {
        System.out.println("A2 method");
    }
}
```

④

```
class Student {
    int rollno;
    String name;
// parameterized Constructor
    Student(int roll, String n)
    {
        rollno = roll;
        name = n;
    }
    void print {
        System.out.println(rollno+" "+name);
    }
```

```
public static void main (String args[]){
    Student s = new Student ("1", "Rafi");
    Student s1 = new student (2, "Ravi");
    s.print();
    s1.print();
}
}
```

(4) • It allows us to pass the values during object creation.

(5)
```
public class Static {
    int a = 90;
    static int a1 = 100;
    void calculate() {
        int b = a+100;
        int c = a1+20;
    }
    static void derive()
    {
        int a3 = a1+100;
    }
    public static void main (String args) {
        Static s = new Static();
        System.out.println(s.a);
        System.out.println(s.a1);
    }
}
```

Instance variable:

```
Public class Instance {
    int m = 15;
    int m1 = 20;
    void a()
    {
        System.out.println(m);
        System.out.println(m1);
```

```
Public static void main (String args[]){
    Instance i = new Instance();
      i.a();
  }
}
```

local variable:

```
Public class Local {
      int a=50; //instance variable
      int b=60;
  void cal()
  {
  int m=20; //local variable
   int n=a+n;
System.out.println(m);
System.out.println(n);
}
Public static void main (String args[])
     Local l=new Local();
      l.cal();
  }
}
```