Study the Student class and answer.

Write the heading for the constructor public Student (int n) ~~String s~~

Name 2 different types of instances variables

String name

int studentNo

int[] marks

int numMarks

List the mutator methods:

passCourse

List the acessor methods and the type they return:

average type float

Find some local variables:

grade
ave
sum

Find some parameters:

int mark
int n
~~String s~~

Write a 2 argument constructor for this class:

```
public Student( String nm, int sNum) ~~int mMarks~~
{
    name = nm;
    studentNo = sNum;
    numMarks = 0;
}
```

```
public class Bicycle {

   private int cadence;
   private int gear;
   private int speed;

   private int id;

   private static int numberOfBicycles = 0;


   public Bicycle(int startCadence,
                  int startSpeed,
                  int startGear){
      gear = startGear;
      cadence = startCadence;
      speed = startSpeed;
      numberOfBicycles++;
      id = numberOfBicycles;
   }
```

//WRITE 2 MORE CONSTRUCTORS

```
public Bicycle ( . ) {
      cadence = 3;
      gear = 2;
      speed = 25;
}

public Bicycle ( int speed1) {
      speed = speed1;
      cadence = 3;
      gear = 2;
}
```

```
   public int getID() {
      return id;
   }

   public static int getNumberOfBicycles() {
      return numberOfBicycles;
   }

   public int getCadence(){
      return cadence;
   }

   public void setCadence(int newValue){
      cadence = newValue;
   }
```

//WRITE OVERLOADED METHOD setCadence which will always set cadence to a random number between 1 and 20.

```
public void setCadence ( ) {
      cadence = (int) (20* Math.random() + 1);
}
```

```
   public int getGear(){
   return gear;
   }

   public void setGear(int newValue){
      gear = newValue;
   }

   public int getSpeed(){
      return speed;
   }

   public void applyBrake(int decrement){
      speed -= decrement;
   }
```

//WRITE OVERLOADED METHOD FOR applyBrake which takes a double and decrements by that.

```
public void applyBrake(double dec) {
      speed -= (int)dec;
}
```

or `Math.round(dec);`

```
//

   public void speedUp(int increment){
      speed += increment;
   }
}
```

1. look at following code and answer questions:

```
public class PlantRunner {

  public static void main(String[] args) {
    Plant p1 = new Plant();
    p1.setName("Adolpho");
    Plant p2 = new Plant();
    Plant p3 = new Plant();
    int waterToAdd =
      (int)(101*Math.random());
    p1.addWater(waterToAdd);
    System.out.println(p1.toString());
    String nm2 = p2.getName();
  }
}
```

```
public class Plant {

 private String name;
 private boolean living;

 public Plant() {
   living = true;
   name = "???";
 }

public Plant(String nm) {
   name = nm;
   living=true;
 }


 public void setName(String inName) {
    name = inName;
 }

 public String getName() {
    return name;
 }

 public void addWater(int h2o) {
   if (h2o < 20 || h2o > 80)
     living = false;
 }

 public String toString() {
  String x="false";
  if (living) x = "true";

  String temp = name+" "+x;
  return temp;
 }
```

W Write the header line for a constructor with 1 parameter:

public Plant(String nm)

W Write the names of all accessor methods:

gg getName
toString

W Write the name of all mutator methods:

addWater
setName

L List all instance varaibles:

name
living

Li List any parameters:

nm
inName
h2o

List all local variables:

String x
temp

W Write a line to add to the client code which will instantiate a plant object with name BillyBob.

Plant BillyBob = new Plant("BillyBob");
↑ ang

W Write a constructor that takes 2 parameters:

```
public Plant (String nm, boolean liv){
    name = nm;
    living = liv;
}
```

Write a method setDead which will kill a plant and the line in client code to call this method.

```
public void setDead ( ){
    living = false;
}
```

---

client code: p1.setDead();