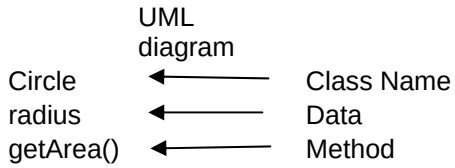
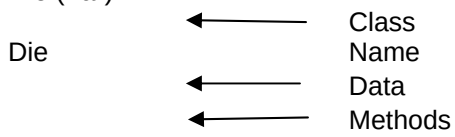
CS122/Sec. 2
Week No. 3
Dilce Gozuyasli
Sevi Durdu
Due date: March 7, 08

Notes of February 26, 08

Review of class Circle

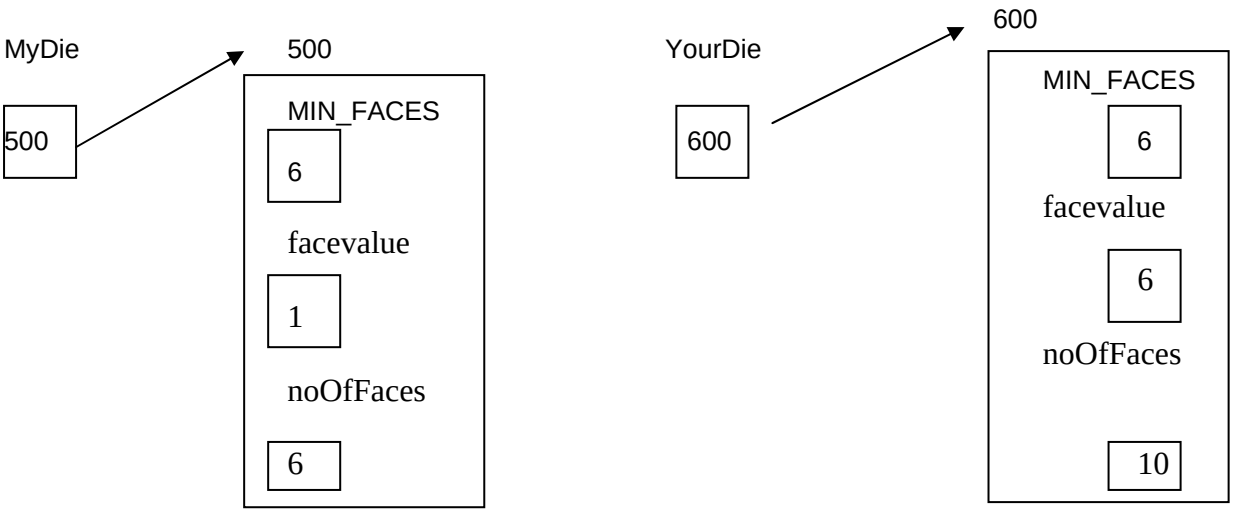


Class Die




```
public class Die{
        private final int MIN_FACES;
        private int facevalue;
        private int noOfFaces;

        public Die(){// default constructor
        noOfFaces=6;
        facevalue=1;
}
        public Die(int faces){
                if (faces<MIN_FACES)
                noOfFaces=MIN_FACES;
                else
                noOfFaces=faces;
                facevalue=(int)(Math.random()*numOfFAces+1);// or it could be like
facevalue=1;
        }
        public int roll(){
                facevalue=(int)(Math.random()*numOfFAces+1);
                return facevalue;
        }
        public int getFaceValue(){
                return facevalue;
        }
}// end of the class
```

Flash-forward:
In the main method of another class:
Die myDie=new Die(); ← Calls default constructor
Die yourDie=new Die(10); ← Calls other constructor



Note : Constructors are used to create objects.

```
public class Tester{
        public static void main(String[] args){
                int count1=0; count2=0; count3=0; count4=0; count5=0; count6=0;
                int noOfRolls;
                int dieFaceCount;
                System.out.println("how many faces do you want?");
                dieFaceCount=Keyboard.readInt();
                System.out.println("enter the number of rolls");
                noOfRolls=Keyboard.readInt();
                Die myDie=new Die(dieFaceCount);// object is created
                for (int i=1; i<=noOfROlls; i++){
                        int fVal=myDie.roll();
        }
```

Note: `int fVal=myDie.roll();` could be written as

```
int fVal;
myDie.roll();
fVal=myDie.getFaceValue();
```

Both of them give same result.

```
                for (int j=1;j<=noOfRolls;j++){
                        switch(myDie.roll()){
                                case 1: count1++;
                                        break;
                                case 2: count2++;
                                        break;
                                case 3: count3++;
                                        break;
                                case 4: count4++;
                                        break;
                                case 5: count5++;
                                        break;
                                case 6: count6++;
```

break;

} //the end of the switch
} // the end of the for loop
In switch, if breaks are not written, then all cases are tested and program will not be efficient. Thus, break is unnecessary for the last case (case 6 in this example).
Note: switch can be used only for int and char types. If it is used with char data type it should be written as case 'a': to test whether it is 'a' or not.

Notes of February 28, 08

Diagram for "Class Die" example

Die
numberOfFaces:int faceValue:int
Die() Die(faces:int) roll():void getFaceValue():int setFaceValue(fvalue:int):void

To set face value, we can write:

1st way

```
public void setFaceValue(int fValue){
faceValue=fValue;  }
```

2nd way

```
public void setFaceValue(int faceValue){
this.faceValue=fValue;  }
```

To copy face value, we can write:

1st way

```
public void copyFaceValue(Die otherDie){
this.faceValue=otherDie.getFaceValue();  }
```

2nd way

```
public void copyFaceValue(Die otherDie){
this.faceValue=otherDie.faceValue;  }
```

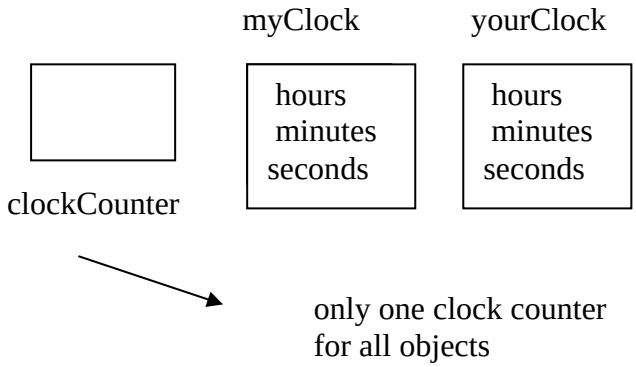
3th way

```
public void copyFaceValue(Die otherDie){
```

faceValue=otherDie.faceValue; }

Let's work on another Class called "Clock"

```
public class Clock{
private int hours, minutes, seconds;
private static int clockCounter=0;

public Clock(){ //default constructor
hours=0; minutes=0; seconds=0; clockCounter++; }

public Clock(int inHours; int inMinutes; int inSeconds){
hours=inHours; minutes=inMinutes; seconds=inSeconds; clockCounter++; }

public Clock(Clock otherClock){ //user defined constructor
hours=otherClock.hours;
minutes=otherClock.minutes;
seconds=otherClock.seconds;
clockCounter++; }

public void setHours(int inHours){ // setter method
hours=inHours; }

public void setMinutes(int inMinutes){ // setter method
minutes=inMinutes; }

public void setSeconds(int inSeconds){ // setter method
seconds=inSeconds; }

public int getHours(){  //getter method
return hours; }

public int getMinutes(){ //getter method
return minutes; }

public int getSeconds(){ //getter method
return seconds; }

public void tick(){ //corrects time, a member method
int totalSeconds;
totalSeconds=3600*hours+60*minutes+1*seconds;
hours=totalSeconds/3600%24;
minutes=totalSeconds%3600/60;
seconds=totalSeconds%3600%60; }

}
```



Questions

1. Why do we use “this” ? What is its function?

“this” helps us to differ to variables with same name. The term that is called with this means that the variable is class variable.
For example, in setValue method we said “this.faceValue=faceValue”. In this expression, we mean that let faceValue of the class have the value that is set in the method.
(if faceValue=5 and we say setFaceValue(4) in the method, then faceValue will be equal to 4.)

2. We used an expression like “Die.faceValue” which will give the faceValue of the class. In what conditions we can get a varible in this manner?

This expression is acceptable only if it is used in member methods.

3. Why did we use static while creating clockCounter varible?

If you want to create a variable which will give same result for all objects, you should use static, so it means you create only one variable that can be used for all objects.

4. Why did we use “final” while creating MIN_FACES variable in Die class?

If you want to create a variable with a constant non-changable value, you should use final. That means, there is no way to change its value in other steps.

5. Question 7.9 from text book. Please try to solve the exercise and crosscheck the output. (This exercise was used in section 1’s lab (not used in section 2’s lab), and it is a very good exercise to understand creating classes.)

```
public class MyInteger {
   private int value;

   public MyInteger(int value) {
      this.value=value;}

   public int getValue(){
```

```
    return value; }

public boolean isEven(){
    boolean ans=false;
    if (value%2==0)
        ans=true;
    return ans; }

public boolean isOdd(){
    boolean ans=false;
    if (value%2==1)
        ans=true;
    return ans; }

public boolean isPrime(){
   boolean ans=true;
   int count=0;
   double test=(Math.sqrt(value))+1;
   for (int i=2;i<test;i++){
      if (value % i ==0)
         count++;
   }
   if (count>0)
      ans=false;
  return ans; }

 public static boolean isEven(int a){
    boolean ans=false;
    if (a%2==0)
        ans=true;
    return ans; }

public static boolean isOdd(int a){
    boolean ans=false;
    if (a%2==1)
        ans=true;
    return ans; }

public static boolean isPrime(int a){
   boolean ans=true;
   int count=0;
   double test=(Math.sqrt(a))+1;
   for (int i=2;i<test;i++){
      if (a % i ==0)
         count++;
   }
   if (count>0)
      ans=false;
   return ans; }

    public static boolean isEven(MyInteger a){
```

```
        boolean ans=false;
        int val=a.getValue();
        if (val%2==0)
            ans=true;
        return ans; }

    public static boolean isOdd(MyInteger a){
        boolean ans=false;
        int val=a.getValue();
        if (val%2==1)
            ans=true;
        return ans; }

    public static boolean isPrime(MyInteger a){
       boolean ans=true;
       int val=a.getValue();
       double test=(Math.sqrt(val))+1;
       for (int i=2;i<test;i++){
           if (val % i ==0)
               ans=false;
       }
      return ans; }

    public boolean equals(int a){
       boolean ans=false;
        if (a==value)
            ans=true;
       return ans; }

    public boolean equals(MyInteger a){
        boolean ans=false;
        int comVal=a.getValue();
        if(comVal==value)
            ans=true;
        return ans; }

    public static int parseInt(char[] a){
     int numval=0;
     for(int i=0;i<a.length;i++){
         numval=(int)(numval+a[i]*(Math.pow(10,a.length-i-1)));
     }
     return numval; }

}
```

```
public class TestMyInteger {

    public static void main(String [] args) {
        MyInteger testvalobj=new MyInteger(5);
```

```
        int tv=testvalobj.getValue();
        boolean isev1=testvalobj.isEven();
        boolean isod1=testvalobj.isOdd();
        boolean ispri1=testvalobj.isPrime();
        boolean isev2=MyInteger.isEven(tv);
        boolean isod2=MyInteger.isOdd(tv);
        boolean ispri2=MyInteger.isPrime(tv);
        boolean isev3=MyInteger.isEven(testvalobj);
        boolean isod3=MyInteger.isOdd(testvalobj);
        boolean ispri3=MyInteger.isPrime(testvalobj);
        boolean iseq1=testvalobj.equals(tv);
        boolean iseq2=testvalobj.equals(testvalobj);
        char[] charray={'a','b','d','g','h'};
        int ch2num=MyInteger.parseInt(charray);
        System.out.println(tv+" "+isev1+" "+isod1+" "+ispri1+" "+isev2+" "+isod2+" "+ispri2+"
            "+isev3+" "+isod3+" "+ispri3+" "+iseq1+" "+iseq2);
    }

}
```

Output will be:

5 false true true false true true false true true true true