CS 112-WEEK 4 (March 4-6, 2008)
Yıldız KOCA
Kumsal ÖZGÜN

Clock(revisited)

```
public class Clock {

   static int clockCounter = 0;
   private int hours,  minutes,  seconds;

  //default constructor. The consructor name should match to its class name. You can invoke
  // it in the main  by such an  object in:
  //Clock myClock= new Clock();     that is it does not take any parameter as input .

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

   //other constructor. This consructor name should also match to its class name. You can
   // invoke it in the main  by such an  object in:
   //Clock yourClock= new Clock(13,50,0);     that is it does  take parameters as input .

   //Since the variable names are same (seconds) we use "this" key. Actually these variables
   //are not same, just, they have same name.

   public Clock(int inh, int inm, int seconds) {
      hours = inh;
      minutes inm;
      this.seconds = seconds;// new(this) seconds = old seconds
      clockCounter++;
   }

   public static void main(String[] args) {
      Clock myClock = new Clock();
      Clock yourClock = new Clock(13, 50, 0);
   }
}
```

Question: Do we create primitive type data?
Answer: No. The programmer defined data types are referance data types like String.

Question: Why do we some times need final instead of static while declaring a variable?
Answer: If we write final there will be compile error. By using final we get one copy of the value and we can not change it. But, by using static, we get one copy of the value and we can change it when we need.

Main : Memory:

//primitive type information:
//(|| char||short||double||boolean||long||real||int)

int i=10; ==========➔ i

10

Clock myClock= new Clock; ======➔ myClock 200

200

hours
0
minutes
0
seconds
0

//To reach the members of the block we should equalize it to a referance variable, otherwise
//it will collect by garbage collection, since there is no path to reach it
// for every int variable, 4-byte-memory is created, that is 32 bits, interms of binary numbers
// binary written:
//Eg: 123=1*10^2+2*10^1+3*10^0

Note: “new” invokes a consructor according to the parameters.

Question: When we use static while declaring a variable in class, does it keep counting even it is in some other methods?
Answer: Since it is not a part of main method, yes, it keeps counting.

```
//TO STRING METHOD
public String toString(){
   String hzero,mzero,szero;
   String result;
   hzero=(hours<10)”0”:””;
   mzero=(minutes<10)”0”:””;
   szero=(seconds<10)”0”:””;
   result=”Time” + “ ” +hzero + hours +”:” +mzero+ minutes+ “:” szero+seconds;
   return result;
}
System.out.println(myClock.toString());
}
```

Note: “toString” is a special name like “equals” ,”clone”.

Arrays of Clocks

For primitive data types

int [] a=new int[p];

```
Clock [ ] myClocks=new Clock[2];
myClocks[0]=new Clock[ ];
myClocks[1]=new Clock[2,53,0];

for(int i=0;i<myClocks.length;i++)
System.out.println(myClocks[i]);

public Class BetterClock
                extends clock
{
  private String brandName;

BetterClock myNewClock=new BetterClock(15,3,10,"casio")


public BetterClock(int h, int s, int m, String bName)
{
   Super(h,m,s);
   brandName=bName;
}
```

String & Text I/0

The String Class: String is not a primitive datatype.
int,char

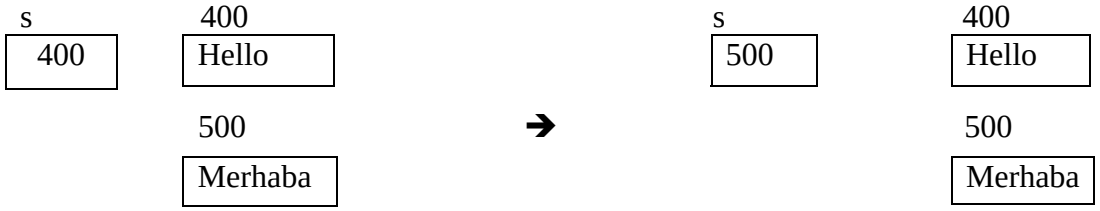
String newStr=new String (String Literal) //Literal is constant, newStr is reference variable.

String message=new String("Hello"); // => String message "Hello"

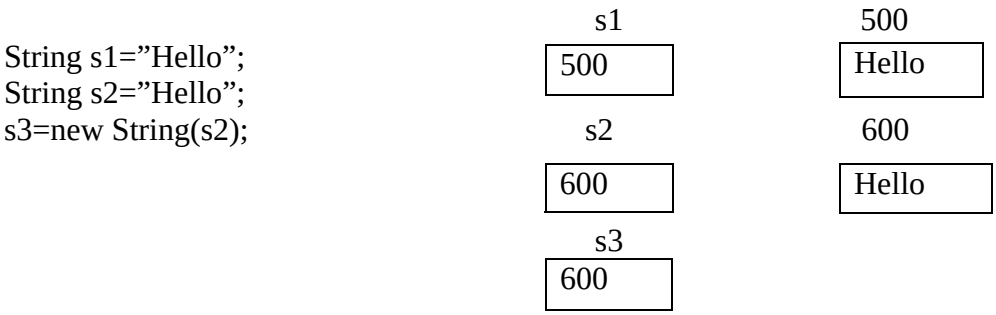

```
char[ ]  charArray={'H','e','l','l','o'};
int [ ]   a={1,5,10};
//works with same principle
```

String message=new String(charArray);

```
String s="Hello";
        s="Merhaba";
```



(“Hello” is collected by the garbage collector.)



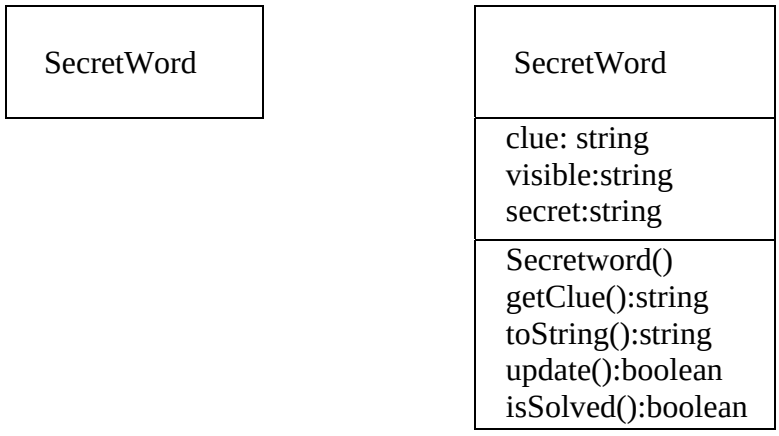

```
if(s1==s2)
System.out.println(“They are equal”);
else
System.out.println(“They are not equal.”);
```

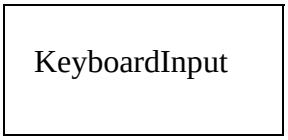
compareTo: usage: if (s2.compareTo(s3)==0)

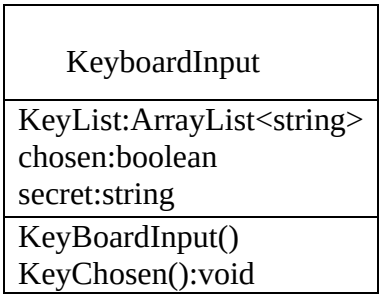
conditions ➔ 0; if s2=s3;
>0; if s2>s3;
<0; if s2<s3;

s2	s3		
anne	baba	<0	according to
baba	anne	>0	dictionary order

Example Program:







HANGMAN PROGRAM EXAMPLE

public class hangManProject

```
import java.util.Scanner;

// provided by
// Nur Timurlenk, Okan Dukkanci, Neslihan Kahyaoglu, Efe Subasi, Gizem Ozbaygin
// (Section # 07)

public class hangManProject
{
   public static void main( String[] args)
   {
      Scanner scan = new Scanner( System.in);



      // CONSTANTS

      // VARIABLES
      int totalMistake = 0;
      boolean b = false;
      // PROGRAM CODE
```

```
        showWelcome();
        SecretWord word = new SecretWord();
        System.out.println("Your clue: " + word.getClue());
        KeyboardInput input = new KeyboardInput();
        while (totalMistake < 4 && !word.isSolved())
        {
           System.out.println(word);
           char c = input.getNewLetter();
           if (!word.update(c))
              totalMistake = totalMistake + 1 ;
        }
        if (word.isSolved())
           System.out.println("Great, you done it.");
        else
           System.out.println("Sorry, too many errors.");

    }

        public static void showWelcome()
        {
           System.out.println( "Welcome to Hangman Game:)!!" );
        }

}
```

public class SecretWord

```
public class SecretWord
{
   String clue, visible, secret;

public SecretWord()
{
   String [] wordList = { "Umbrella", "Richard Feynman", "American Gangster", "The
Silmarillion" };

   String [] clueList = { "Rihanna's song", "Nobel Prize in Physics", "Denzel Washington",
"Tolkien" };
   int rand;
   rand = (int) (Math.random()*4);
   secret = wordList[rand];
   clue = clueList[rand];
   int wordLength = secret.length();

   visible = "";

   for ( int i = 0; i < secret.length(); i++)
   {
```

```
        char c = secret.charAt(i);

        if ( c == ' ' )
            visible += c;
        else
            visible += '_';
    }
}

public String getClue()
{
    return clue;
}

public String toString()
{
    // String s = "";
    // for(int i = 0; i < wordLength; i++)
    //    s = s + "_ ";
    // return s;

    return visible;
}

    // provided by Group 4 and 8
    public boolean update(char c)
    {
        char[] secretArray = new char [secret.length()];
        char[] visibleArray = new char [secret.length()];
        String s = "";
        boolean contains = false ;

        //We make the String secret an array
        for(int i = 0; i < secret.length(); i++)
        {
            secretArray[i] = secret.charAt(i);
            visibleArray[i] = visible.charAt(i);
        }

        //We are uptading our empty visible array
        for(int k = 0; k < secret.length();k++)
        {

            if(secretArray[k] == c)
            {
                visibleArray[k] = c;
                contains = true;
            }
        }
        //We are changing our visible array into the visible string
```

```
        for(int m=0; m < secret.length(); m++ )
        {
           s = s + visibleArray[m];
        }
        visible = s;

        return contains;
    }
    //This is a method to check if the secret word is solved
    public boolean isSolved()
    {
        return secret.equalsIgnoreCase(visible);
    }
}
```

public class KeyboardInput

```
import java.util.ArrayList;
import java.util.Scanner;

public class KeyboardInput
{
    // Properties
    ArrayList<String> keyList ;
    boolean chosen;

    //Constructor
    public KeyboardInput()
    {
        keyList = new ArrayList<String>();
    }

    // Adds and stores the entered key in the Arraylist
    public void keyChosen( String key )
    {
        keyList.add( key );
    }

    // Checks if the entered key has been chosed before
    public boolean hasBeenChosen( String key)
    {
        chosen = keyList.contains( key );
        return chosen;
    }


    public char getNewLetter()
    {
        Scanner scan=new Scanner( System.in);
```

```
        char a;
        boolean chosen;
        String aTmp;    // used to convert a into a String for compatibility with Group 6

        do
        {
           System.out.println("Enter a Letter");
           a = scan.next().charAt( 0);
           aTmp = "" + a;
           chosen = hasBeenChosen( aTmp);
           if(!chosen)
              keyChosen( aTmp);
           else
           {
              System.out.println("You've already used that one. Please enter a new Letter");

           }


        }
        // we repeat the process until we're out of trials.
        while(chosen);

        return a;
     }
}
```

QUESTİONS:

1) How can *compareTo* method distinguish the alphabetical order of the letters?
Answer: Each letter has an ascii code, and these codes are formed parallelly to their order in alphabet. In this way, this method can compare two strings.

2) What is the meaning of *static* in the class?
Answer: When we use *static* while defining a parameter, it means that in whole class there will be just one parameter which is used for the same target.

3) How can *super()* works by accurately choosing the parameters from the class?
Answer: Because, there is no multiinheritance in java, *super()* does not confuse about the other parameters.

4) Create a class Cylinder which has a default constructor, a user constructor and getter methods. Then test your class in a main method by creating an array.

Solution:

```
public class TesterCylinder {

   public static void main(String[] args) {
```

```
        Cylinder[] Cylinder1 = new Cylinder[4];
        Cylinder1[0] = new Cylinder(3, 6);
        Cylinder1[1] = new Cylinder(10, 20);
        Cylinder1[2] = new Cylinder(5, 10);
        Cylinder1[3] = new Cylinder(2, 4);
        for (int i = 0; i < 4; i++) {
            System.out.println(i + 1 + ": ");
            System.out.println("length: " + Cylinder1[i].getLength());
            System.out.println("radius: " + Cylinder1[i].getRadius());
            System.out.println("volume: " + Cylinder1[i].getVolume());
            System.out.println("surface area: " + Cylinder1[i].getSurfaceArea());
        }
    }
}

class Cylinder {

    double radius, length;

    public Cylinder() {
        radius = 1;
        length = 1;
    }

    public Cylinder(double r, double l) {
        radius = r;
        length = l;
    }

    public double getRadius() {
        return radius;
    }

    public double getLength() {
        return length;
    }

    public double getVolume() {
        double volume = 3.14 * radius * radius * length;
        return volume;
    }

    public double getSurfaceArea() {
        double surfaceArea = (2 * 3.14 * radius * length) + 2 * 3.14 * radius * radius;
        return surfaceArea;
    }
}
```

5) Do we create primitive typed data?
Answer: No. The programmer defined data types are reference data types like String