

# Exercise

CSC140 Foundations of Computer Science

19 January 2015

I would like you to “learn through your fingers” by transcribing these programs and then defining a new method for each class as directed. Work with a partner. Type one of the programs yourself while your teammate types the other. When you both finished typing, each of you can copy the other’s program into your own folders:

Let us suppose that your partner Steve has a home folder named *sjobs*. He transcribed the *search* program while you transcribed the *weight* program. You can get a copy of Steve’s work by opening a terminal window and typing the following:

```
cd
cp -R ../sjobs/NetBeansProjects/search NetBeansProjects
```

## Search.java

Create a project named *search* that contains a single class named *Search*. Copy the code that follows into the class.

Add another method named *minimum*. This method will differ from the given method named *minimum* by the fact that it will have a second parameter. This second parameter will be the index where the search for the smallest value will begin. The method will return to its caller the value of the smallest element in just a part of the array specified by the first parameter.

```
package search;

public class Search {

    private static double [] makeArray( int size ) {
        double [] data = new double[ size ] ;
        for( int i = 0; i < data.length; i++ ) {
            data[i] = Math.random();
        } // for
        return data;
    } // makeArray()
```

```

private static void printArray( double [] data ) {
    for( int i = 0; i < data.length; i++ ) {
        System.out.printf( "%2d:_%8.2f\n", i, data[i] );
    } // for
    System.out.println();
} // printArray()

private static double minimum( double [] data ) {
    double bestGuessSoFar = data[0];
    for( int i = 1; i < data.length; i++ ) {
        if( data[i] < bestGuessSoFar ) {
            bestGuessSoFar = data[i];
        } // if
    } // for
    return bestGuessSoFar;
} // minimum( double [] )

private static int positionOfMinimum( double [] data ) {
    int bestGuessSoFar = 0;
    for( int i = 1; i < data.length; i++ ) {
        if( data[i] < data[bestGuessSoFar] ) {
            bestGuessSoFar = i;
        } // if
    } // for
    return bestGuessSoFar;
} // positionOfMinimum( double [] )

public static void main( String[] args ) {
    double [] numbers = makeArray( 12 );
    printArray( numbers );
    System.out.printf( "Smallest_value=_%8.2f\n",
                      minimum(numbers) );
    System.out.println( "Position_of_smallest_value:_ " +
                       positionOfMinimum(numbers) );
} // main( String[] )
} // Search

```

## Weight.java

Create a project named *weight* that contains a single class named *Weight*. Copy the code that follows into the class.

Add a method named *toKilograms* that returns to its caller the value of a weight expressed in kilograms. (The value is a single number that may have a fractional part.)

```

package weight;

public class Weight {
    private static final int OUNCES_IN_A_POUND = 16;

    private int pounds;
    private int ounces;

    public Weight( int pounds, int ounces ) {
        this.pounds = pounds;
        this.ounces = ounces;
    } // Weight( int, int )

    public int getPounds() {
        return this.pounds;
    } // getPounds()

    public int getOunces() {
        return this.ounces;
    } // getOunces()

    public void setPounds( int pounds ) {
        this.pounds = pounds;
    } // setPounds( int )

    public void setOunces( int ounces ) {
        this.ounces = ounces;
    } // setOunces( int )

    public Weight add( Weight otherWeight ) {
        int lbs = this.pounds + otherWeight.pounds;
        int oz = this.ounces + otherWeight.ounces;
        Weight sum = new Weight( lbs + oz/OUNCES_IN_A_POUND,
                                   oz % OUNCES_IN_A_POUND );

        return sum;
    } // add( Weight )

    @Override
    public String toString() {
        return this.pounds + "_lbs.,_" + this.ounces + "_oz.";
    } // toString()

    public static void main( String[] args ) {
        Weight cherries = new Weight( 1, 10 );
        Weight grapes = new Weight( 2, 8 );
        Weight fruit = cherries.add( grapes );
    }
}

```

```
        System.out.println( cherries + " + " + grapes +  
                             " = " + fruit );  
    } // main( String[] )  
  
} // Weight
```