CSC 160 WEEK 8 TEST

SOLUTIONS

Note: The last page of this test lists some classes and methods that may be useful in answering some of the questions.

1. Suppose that A is a class. Explain what it means to say that a second class B is a *subclass* of A.

> *B starts out with all the methods and fields of the class A, and can add its own methods and fields.*

2. Explain precisely what is meant by the term *method overloading.*

> Method overloading *occurs when a class has more than one method of the same name, but different parameter type lists.*

3. Explain precisely what is meant by the term *method overriding.*

> Method overriding *occurs when a subclass defines a method that has the same name and parameter type list as a method in the super-class.*

4. Define the concept of a *class constructor.*

> *A* class constructor *is an instance method of a class that is used to create objects of that class. Java constructors have the same name as the class they belong to.*

5. Explain the difference between the *formal parameters* and *actual parameters* of a method.

> *Formal parameters occur in the method definition as place holders for data that will be passed into the method when the method is called.*
>
> *Actual parameters occurs in the method call and represent the actual data to be processed by the method call.*

6. Explain the difference between a *static method* of a class and an *instance method* of the class.

> *A* static method *belongs to the class. It exists independently of any objects of the class.*
>
> *An* instance method *exists in an object of the class: that is, every object of the class has its own copy of that method.*

7. Suppose that there is an array of integers

```
int [] arr;
```

Write some code that prints the minimum value in the array, and the index (position) at which the minimum occurs. If the array has more than one minimum value, print the position of any one of them.

For example if the array is

index:	0	1	2	3	4	5	6
value:	13	78	92	12	8	23	82

Then the code you write should print 4 because the minimum value `arr[4] = 8` occurs at position 4.

```
The minimum value is 8 and occurs at index 4.
```

Note that this is just an example. You should assume that the array can be any length and have any values in it.

Solution:

```
int minPos = 0;  // index of minimum value found so far
for (int k = 1; k < arr.length; k++)
{
   if (arr[k] < arr[minPos])
   {
      minPos = k;
   }
}
System.out.printf("The minimum value is %d and occurs at %d",
                  arr[minPos], miniPos);
```

8. Suppose that there is an array of integers

```
int [] arr;
```

Write code that prints out the array entries in reverse order.

For example if the array is

index:	0	1	2	3	4	5	6
value:	13	78	92	12	8	23	82

Then your code will print the sequence of values

82 23 8 12 92 78 13

Solution: *Note that you only need to print the array values in reverse order. You do not have to create a new array to store the reversed values.*

```
for (int k = arr.length-1; k >= 0; k--)
{
   System.out.print(arr[k] + "   ");
}
```

9. Suppose that you have an array of text fields, and another text field

```
JTextField [] tFs = new JTextField[10];
JTextField sumTf = new JTextField(4);
```

Suppose the user has entered an integer value in each text field. Write code that uses a loop to fetch the integer values from each text field in the array of text fields, adds up those values, and displays the sum of those values in the `sumTf` text field.

You do not have to write a complete program.

Solution:

```
int sum = 0;
for (int k = 0; k < arr.length; k++)
{
   // Get the text from the text field
   String tfStringValue = tFs[k].getText();

   // Convert the text to an integer value
   int value = Integer.parseInt(tfStringValue);

   // Add the value to the accumulator
   sum = sum + valu;
}
// Set the sum text field
sumTf.setText(String.valueOf(sum));
```

10. A Java program has a GUI with one text field and two buttons:

```
JTextField tf = new JTextField(10);
JButton b1 = new JButton("English");
JButton b2 = new JButton("Spanish");
```

When the "English" button is clicked, the program is supposed to display the English word `"Friend"` in the text field. When the "Spanish" button is pressed, the program is supposed to display the Spanish word `"Amigo"` in the text field. The program is required to do this using a single non-nested class that implements the `ActionListener` interface. This class should have a constructor that is passed the appropriate user interface component that it needs.

(a) Pick appropriate strings to use as action command strings, and write code that will enable a single listener object added to the two buttons to determine which button has been clicked.
   Solution:
   ```
   b1.setActionCommand("English");
   b2.setActionCommand("Spanish");
   ```

(b) Write the listener class needed by this program.

SOLUTION:

```
class MyListener implements ActionListener
{
   private JTextField tf;
   public MyListener(JTextField tf1)
   {
       tf = tf1;
   }

   public void actionPerformed(ActionEvent e)
   {
     String action = e.getActionCommand();
     switch(action)
     {
       case "English":
            tf.setText("Friend");
            break;

       case "Spanish":
            tf.setText("Amigo");
            break;
     }
   }
}
```

(c) Write code that creates a listener object and adds that listener object to the two buttons.

SOLUTION:

```
ActionListener listener = new MyListener(tf);
b1.addActionListener(listener);
b2.addActionListener(listener);
```

11. Suppose that your have three arrays of integers that have the same length:

```
int [] a;
int [] b;
int [] s;
```

The two arrays `a` and `b` represent two integer numbers, with each array element holding decimal digit in the range 0, 1, . . . , 9. The highest order digit of the arrays `a` and `b` will always be 0. Write code that put a similar representation of the sum of the two numbers into the third array `s`.

For example, if the arrays `a` and `b` have the values shown below, then your code will fill in the entries of the array `s` as shown.

Note that in displaying the entries of the array, entries of lower index are shown to the right of entries of higher index.

a:	0	5	9	3
b:	0	7	4	1
s:	1	3	3	4

Hint: At each position, you want to add the two digits at that position and a carry from the previous position. Take the carry at index position 0 to be zero.

Solution:

```
int carry = 0;
for (int k = 0; k < a.length; k++)
{
   // sum
   s[k] = (a[k] + b[k] + carry) % 10;
   // carry
   carry = (a[k] + b[k] + carry) / 10;
}
```

Notes and Documentation

(1) The `ActionListener` interface contains a single method

```
public void actionPerformed(ActionEvent evt)
```

(2) The `ActionEvent` class has a method

```
String getActionCommand()
```

(3) The `JTextField` class has methods

```
String getText()
void setText(String text)
```

(4) The `JButton` class has methods

```
void setActionCommand(String ac)
void addActionListener(ActionListener listener)
```