

**Due:**

Activity (in-lab): Monday, February 27, 2012 by the end of lab

**Goals:**

By the end of this activity you should be able to do the following:

- Understand the basics of the ternary *conditional operator*
- Understand the basics of the *for loop* and the *do-while loop*

**Description:**

In this activity you will create two classes. `Temperatures` will hold a set of integer values representing daily temperatures. `TemperatureInfo` will allow users to interact with the `Temperatures` class.

**Directions:****Part 1: Temperatures: Method Stubs**

- Create a class called `Temperatures`, which will hold a set of integer values representing daily temperatures.
- Add method stubs for the following methods.

- The constructor takes an `ArrayList` of integer values

```
public Temperatures(ArrayList temperaturesIn) {  
}
```

- `getLowTemp`: takes no parameters; returns an integer value
- `getHighTemp`: takes no parameters; returns an integer value
- `lowerMinimum`: takes an `int` parameter; returns an integer value
- `higherMaximum`: takes an `int` parameter; returns an integer value
- `toString`: no parameters; returns a `String`

**Part 2: Temperatures: instance variable, Constructor, getLowValue**

- Add an instance variable with the name `temperatures` to your class that is of type `ArrayList` with generic type `Integer`.
- In your constructor, set `temperatures` equal to `temperaturesIn`.
- In `getLowValue`, first return 0 if the `ArrayList` is empty:

```
if(temperatures.isEmpty()) {  
    return 0;  
}
```

- Now iterate through the entire list and find the lowest temperature:

```
for (int i = ____; i < ____; i++) {  
    if (temperatures.get(i) < low) {  
        low = temperatures.get(i);  
    }  
}
```

- Finally, return the lowest temperature:

```
return low;
```

### Part 3: getHighValue

- In getHighValue, again return 0 if there are no temperatures in the ArrayList:

```
if(temperatures.isEmpty()) {
    return 0;
}
```

- This time, use a for-each loop to iterate through the list of temperatures to find the highest value.

```
int high = temperatures.get(0);
for (Integer currentTemp : temperatures) {
    if (_____ > high) {
        high = _____;
    }
}

return high;
```

- Add code to the toString method to return a string containing the low and high temperatures (hint: make a method call to getLowValue and getHighValue):

```
public String toString() {
    return "Low: " + _____ ()
        + "\r\nHigh: " + _____ ();
}
```

### Part 4: lowerMinimum & higherMaximum

- The lowerMinimum method takes an int value and returns the parameter if it is lower than the value returned by getLowerValue. Otherwise, it returns the return of getLowerValue.

```
public int lowerMinimum(int lowIn) {
    return lowIn < getLowerValue() ? lowIn : getLowerValue();
}
```

- The higherMaximum method takes an int value and returns the parameter if it is greater than than the value returned by getHigherValue. Otherwise, it returns the return of getHigherValue.

```
public int higherMaximum(int highIn) {
    return _____ ? _____;
}
```

- Test your methods in the interactions pane:

```
import java.util.ArrayList;
ArrayList tempList = new ArrayList();
tempList.add(34);
tempList.add(52);
tempList.add(36);
tempList.add(65);
```

```

Temperatures temps = new Temperatures(tempList\);
temps.getLowValue()
34
temps.getHighValue()
65
temps.lowerMinimum(33)
33
temps.lowerMinimum(35)
34
temps.higherMaximum(64)
65
temps.higherMaximum(67)
67

```

### Part 5: TemperatureInfo

- Create a class called TemperatureInfo with a main method. Declare and instantiate an ArrayList with generic type Integer called tempList. Declare and instantiate a Scanner object called userInput that reads from System.in.
- Create the following do-while loop that will add numbers to the list until the user enters a -1 input. Also create a Temperatures object with the ArrayList as input and print the temperatures object:

```

int tempInput = -1;
do {
    System.out.print("Enter a postive temperature (-1 to stop): ");
    tempInput = userInput.nextInt();
    if (tempInput > -1) {
        _____.add(tempInput);
    }
} while (tempInput > -1);

Temperatures temps = new Temperatures(_____);
System.out.println(_____);

```

Run your program as below to test its output:

```

----jGRASP exec: java Temperatures

Enter a postive temperature (-1 to stop): 45
Enter a postive temperature (-1 to stop): 44
Enter a postive temperature (-1 to stop): 12
Enter a postive temperature (-1 to stop): 33
Enter a postive temperature (-1 to stop): 24
Enter a postive temperature (-1 to stop): -1
Low: 12
High: 45

----jGRASP: operation complete.

```