Unit Three: Decision Statements

Relational Operators in Java

```
==      is equal to
!=      is not equal to
>       is greater than
>=      is greater than or equal to
<       is less than
<=      is less than or equal to
```

Logical Operators in Java

```
&&      and
||      or
!       not
```

Order of Relational and Logical Operators

```
1) !
2) > >= < <=
3) == !=
4) &&
5) ||
```

Note: Whenever in doubt as to the correct order, use parentheses to convey your intent.

Control Statements

1) if (condition) ...

   Examples:

```
   A)  if (num > 0)  System.out.println("Positive");

   B)  if (num > 0)
            System.out.println("Positive");

   C)  if (num == 5)
       {
            System.out.println("Positive");
            System.out.println("Odd");
       }
```

2) if ((condition1) &&/|| (condition2)) ...

Examples:

A) if ((score >= 80) && (score < 90)) System.out.println("B");

B) if ((score < 0) || (score > 100))
```
{
     System.out.println("There is a mistake in your input!");
     System.out.println("Please try again.");
}
```

Def. of Short-Circuit Evaluation: Evaluation of the left operand of || or && but not the right operand when the result of the operation can be determined from the value of the left operand alone.

Examples: A || B - B will not be evaluated if A is true
A && B - B will not be evaluated if A is false

3)
```
if (condition)
{
     statement1;
     statement2;
}
else
{
     statement3;
     statement4;
}
```

Example:

```
if (num % 2 == 0)
     System.out.println("Number is Even");
else
     System.out.println("Number is Odd");
```

4)
```
if (condition1)  statement1;
else if (condition2)  statement2;
else if (condition3)  statement3;
else statement4;
```

Example:

```
if (grade == 65)  points = 4;      // 65 is the ASCII value of "A"
else if (grade == 66)  points = 3;
else if (grade == 67)  points = 2;
else points = 1;
```

5) Nested if's ...

```
if (condition1)
{
     statement1;
     if (condition2)
     {
          statement2;
          statement3;
     }
}
else if (condition3)  statement4;
else statement5;
```

Example:

```
boolean flag;
if (year % 100 == 0)
{
     if (year % 400 == 0)
          flag = true;
     else
          flag = false;
}
else
{
     if (year % 4 == 0)
          flag = true;
     else
          flag = false;
}
```

Comparing Strings

To test if two strings are equal to each other, you must use the method called *equals*.

Example:

```
String A = "Bob";
String B = "BOB";

if (A.equals(B)) System.out.println("Same");
else System.out.println("Different");
```

In this example Different would be printed to the console window.

You can also ask the compiler to test if one string comes before another string in the dictionary.

Example:

```
String X = "Sam";
String Y = "Tom";
String Z = "Sid";                    // ASCII values

if (X.compareTo(Y) < 0)              // 83 - 84 < 0    true
     System.out.println(X);
else
     System.out.println(Y);
if (Y.compareTo(Z) > 0)              // 84 - 83 > 0    true
     System.out.println(Z);
else
     System.out.println(Y);
if (Z.compareTo(X) == 0)             // 105 - 97 == 0  false
     System.out.println("Same");
else
     System.out.println("Different");
```

OUTPUT:

```
Sam
Sid
Different
```

Note: In Java *if (string1 == string2)* is testing if the two string variables refer to the identical string object, meaning that they occupy the same place in memory.

The parseInt & parseDouble methods of the Integer & Double classes

Suppose you have the String "715" and the String "3.14" and you would like to use the numerical values of these Strings. The following code will show you how to do this...

```
public static void main(String[] args)
{
      String x = "715";
      String y = "3.14";

      int a = Integer.parseInt(x);
      double b = Double.parseDouble(y);
}
```

Introduction to use of GUI's in Java (Graphical User Interface)

- not tested on AP Exam -

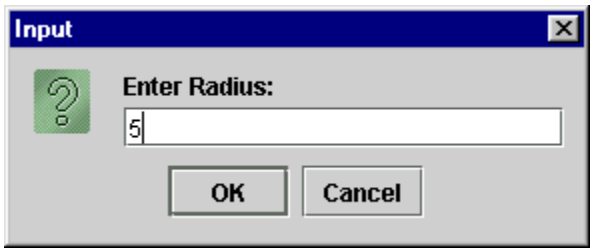
The easiest GUI's to use in Java are the pop-up windows. The methods for these GUI's can be found in the JOptionPane class of the swing package. Your import statement should look like this...

```
import javax.swing.JOptionPane;
```

To input data into a program you would use an Input Dialog box. This dialog box will always return a string. The code would look like this...

```
String input;
input = JOptionPane.showInputDialog("Enter Radius: ");
```

The actual Dialog box would look like this...



To handle the output of a program, you would use a Message Dialog box. To use a Message Dialog, the output must be a string. The code would look like this...

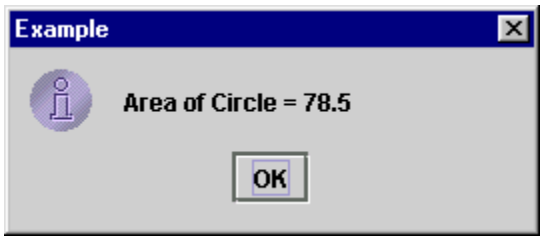
```
int radius;
double area;
String result;
radius = Integer.parseInt(input);
area = 3.14 * radius * radius;
result = "Area of Circle = " + area;
result += "\n\n";

JOptionPane.showMessageDialog(null, result, "Example",
                                    JOptionPane.INFORMATION_MESSAGE);
```

The 1st argument must always be null.
The 2nd argument is your output string.
The 3rd argument is the caption.
The 4th argument is the type of message box you want.

```
     Options:  JOptionPane.INFORMATION_MESSAGE  (i icon)
                    "     .PLAIN_MESSAGE        (no icon)
                    "     .ERROR_MESSAGE        (stop sign icon)
```

The actual Message box would look like this...



Note: Input and output GUI's can be used with both Java applications and applets.

Accessor and Mutator Methods

An accessor method returns to the calling block, the value of an instance field. It does not attempt to change this value. The name of an accessor method usually starts with the word **get**.

In contrast, the purpose of a mutator method is to change the value of an instance field. Mutator methods return nothing, so their return type is void. The name of a mutator method usually starts with the word **set**. A mutator method can be called from a constructor. In this case its modifier would be private and there would be no parameters.

```
private void setNewGrade()
{
     ...
}
```

A mutator method can be called from the driver class. In this case its modifier would be public and there would be a parameter.

```
public void setNewGrade(String grade)
{
     ...
}
```

Assignment U3A1: Letter & Numeric grades