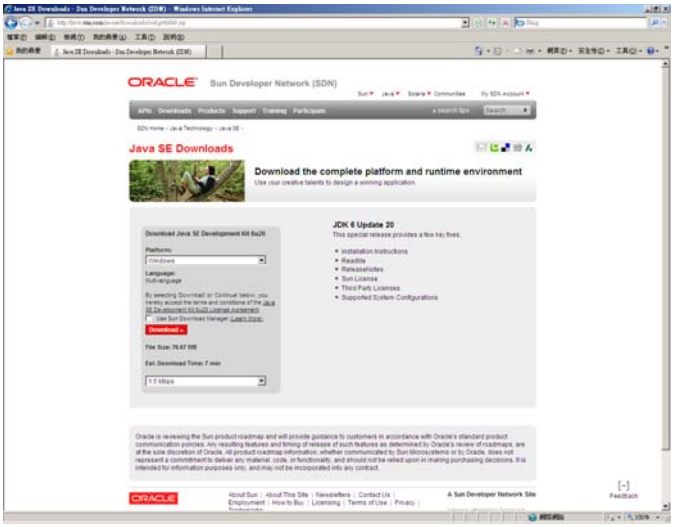
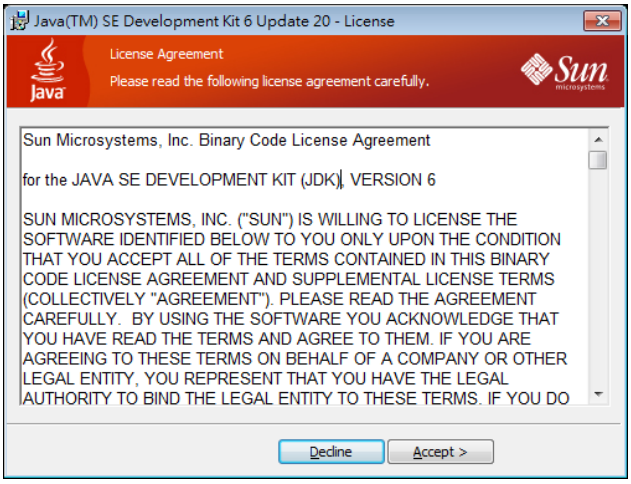
1. Install the JDK

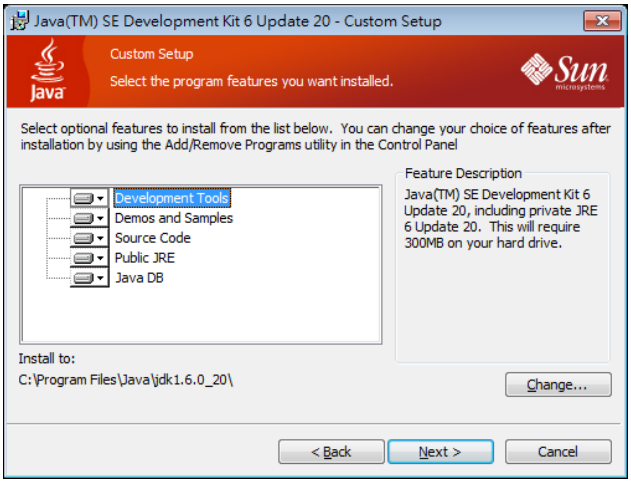
1. Download the JDK 6, from http://java.sun.com/javase/downloads/widget/jdk6.jsp.



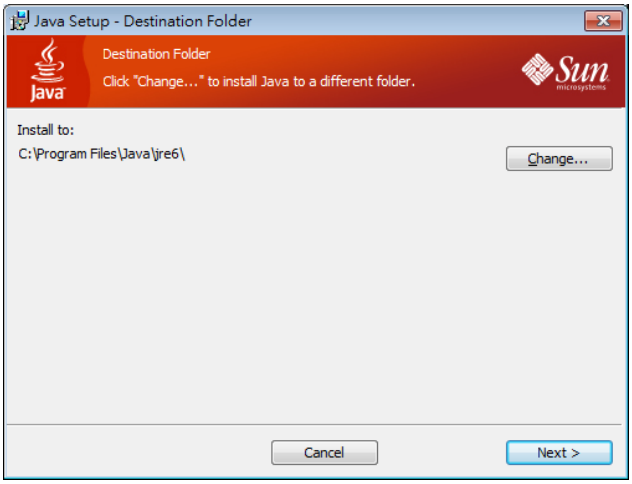
2. Once the file is completed downloaded, execute it and accept the license agreement.



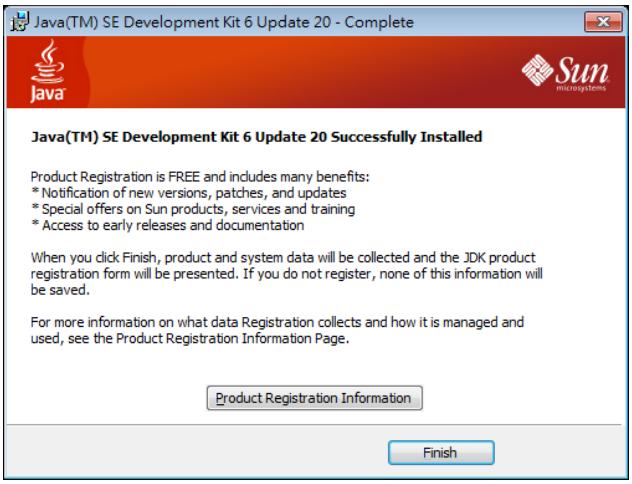
3. Select the component that you want to install (Recommend select all), and then press **[Next]** to continue.



4. Select the path for installing Java Virtual Machine, click **[Next]** to start the setup procedure



5. After the install process finish, click **[Finish]** to complete.



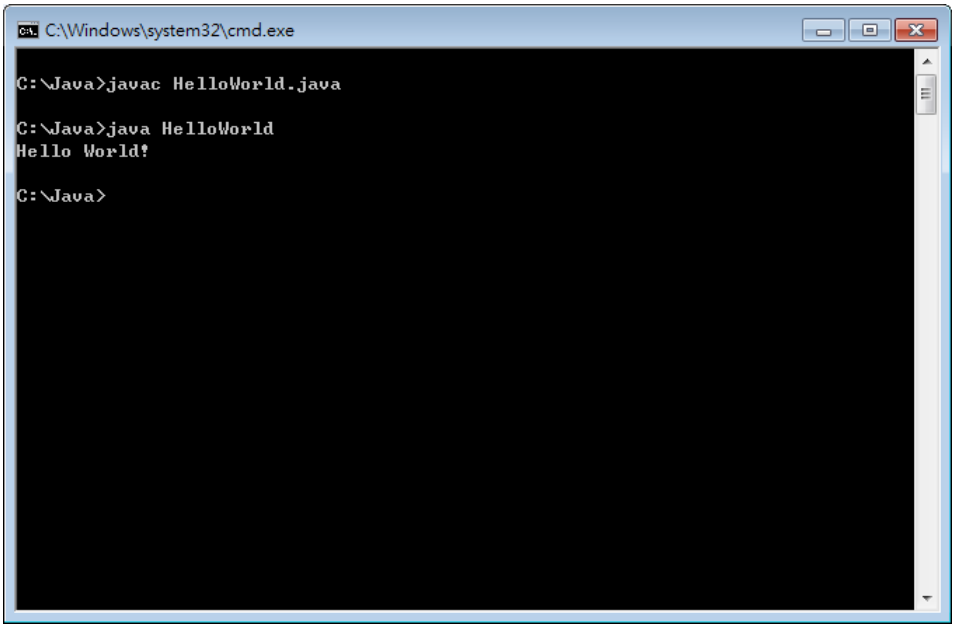
2. Hello World

1. Create a file called "**HelloWorld.java**" in Notepad with the following contents.

```
/** Comment
 * Displays "Hello World!" to the standard output.
 */

class HelloWorld {
  public static void main (String args[]) {
    //Displays the enclosed String on the Screen Console
    System.out.println("Hello World!");
  }
}
```

2. To compile Java code, we need to use the "**javac**" tool. From a command line, the command to compile this program is. ***javac HelloWorld.java***
If the compilation is successful, javac will quietly end and return you to a command prompt. If you look in the directory, there will now be a HelloWorld.class file. This file is the compiled version of your program. Once your program is in this form, its ready to run. Check to see that a class file has been created. If not, or you receive an error message, check for typographical errors in your source code.

3. You're ready to run your first Java application. To run the program, you just run it with the java command: ***java HelloWorld***

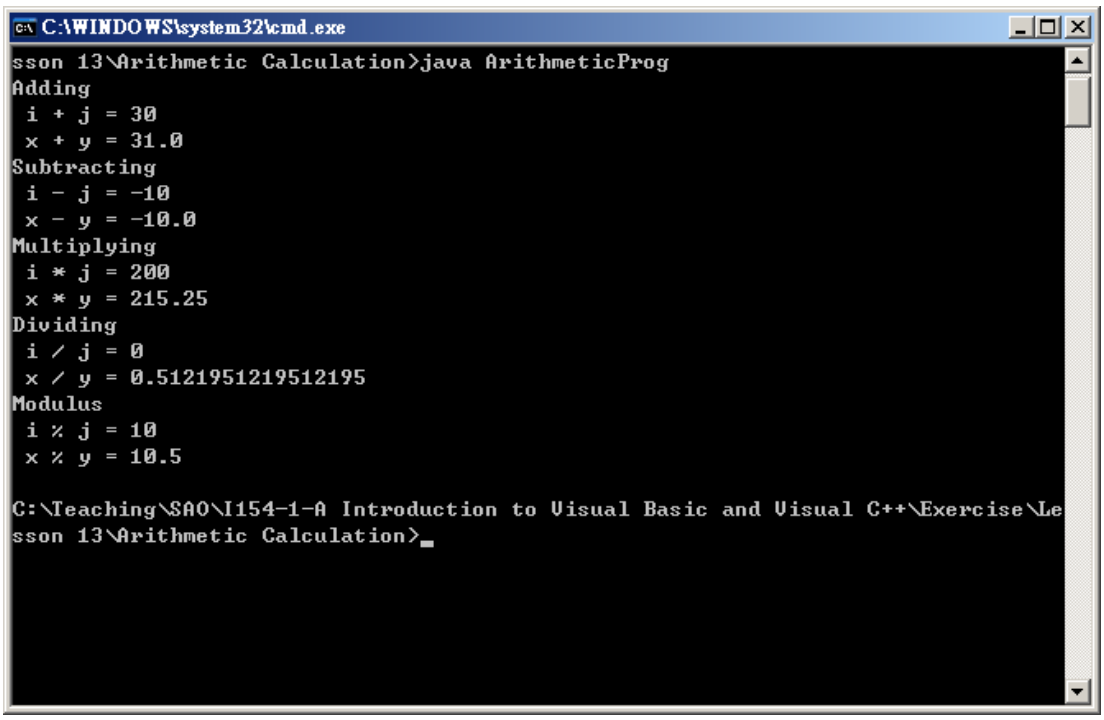


3. Arithmetic Calculation

1. The following java program, **ArithmeticProg.java**, defines two integers and two double-precision floating-point numbers and uses the five arithmetic operators to perform different arithmetic operations. This program also uses + to concatenate strings.

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

    // Declarate variables
    int i = 10;
    int j = 20;
    double x = 10.5;
    double y = 20.5;

    //adding numbers
    System.out.println("Adding");
    System.out.println(" i + j = " + (i + j));
    System.out.println(" x + y = " + (x + y));

    //subtracting numbers
    System.out.println("Subtracting");
    System.out.println(" i - j = " + (i - j));
    System.out.println(" x - y = " + (x - y));

    //multiplying numbers
    System.out.println("Multiplying");
    System.out.println(" i * j = " + (i * j));
    System.out.println(" x * y = " + (x * y));

    //dividing numbers
    System.out.println("Dividing");
    System.out.println(" i / j = " + (i / j));
    System.out.println(" x / y = " + (x / y));

    //computing the remainder resulting from dividing numbers
    System.out.println("Modulus");
    System.out.println(" i % j = " + (i % j));
    System.out.println(" x % y = " + (x % y));
    }
}
```

2. Compile the program and execute it.

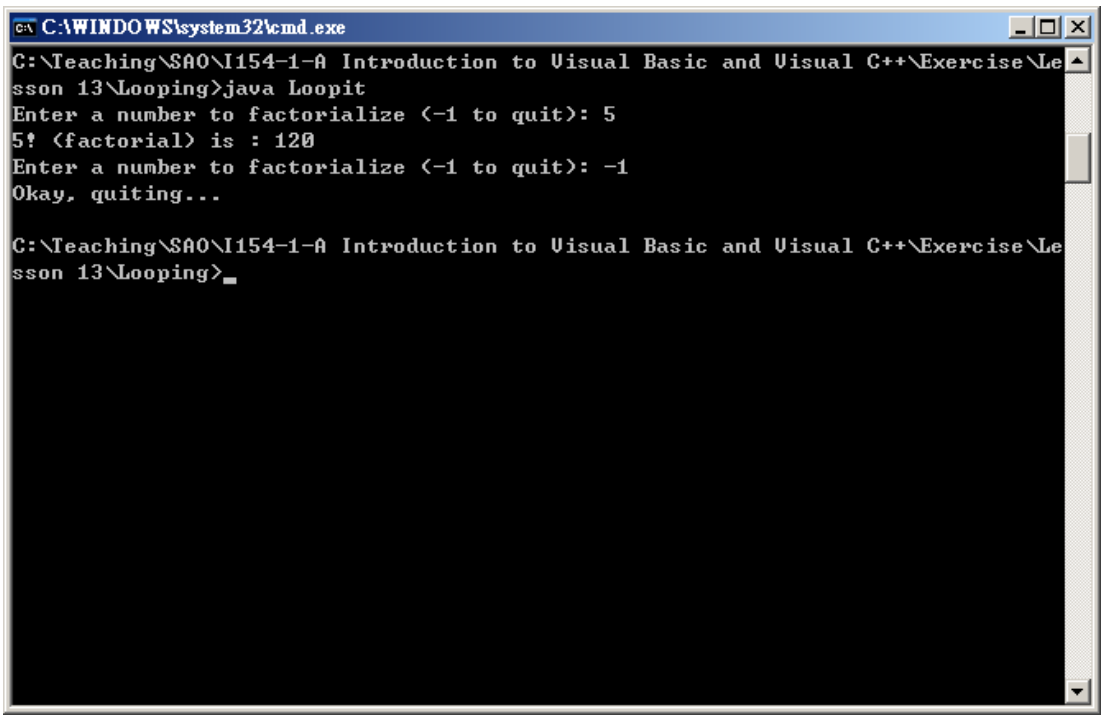


4. Looping

1. Java has several types of loops, the most useful being the for & while loops, which are both demonstrated in this next example. The standard if-then-else, will also be a very handy programming tool. Consider the following Java program "**Loopit.java**".

```
import java.io.*;

class Loopit {
  public static void main (String[] args) throws IOException {
    BufferedReader stdin = new BufferedReader
      (new InputStreamReader (System.in));
    int count, max, num;
    num = 0; // Assign initial value of count

    while (num != -1) {
      System.out.print ("Enter a number to factorialize (-1 to quit): ");
      System.out.flush();

      num = Integer.parseInt (stdin.readLine());
      max = 1; // Assign to 1, so factorial isn't zero every time

      if (num == -1) {
        System.out.println("Okay, quiting...");
      }
      else { // Since they're not quitting we better factorialize
        for  (count = 1; count<=num; count++) {
          max = count * max;
        }
        System.out.println (num+"! (factorial) is : "+ max);
      }
    }
  } // method main
}
```

2. Compile the program and execute it.



5. Class and Object

3. In Java almost everything of interest is either a class itself or belongs to a class. Methods are defined inside the classes they belong to. Even basic data primitives like integers often need to be incorporated into classes before you can do many useful things with them. The class is the fundamental unit of Java programs. Consider the following Java program "**SimpleClass.java**".

```
import java.io.*; //include Java's standard Input and Output routines

class Simple {
// Constructor
    Simple() {
     p = 1;
     q = 2;
     r = 3;
    }

    // Attrib
    int p,q,r;

    // Add Number
    public int addNumbers(int var1, int var2, int var3)
    {
        return var1 + var2 + var3;
    }

    // Method: Display Message
    public void displayMessage()
    {
        System.out.println("Display Message");
    }
}

class SimpleClass {
    public static void main(String args[])
    {
        // To create a new instance class
        Simple sim = new Simple();
```

```
        // To show the result of the addNumbers
        System.out.println("Result =" +
            Integer.toString(sim.addNumbers(5,1,2)));

        // To display message
        sim.displayMessage();
    }
}
```

4. Compile the program and execute it.