LAB # 1

INTRODUCTION TO JAVA

OBJECTIVE

To understand Java Environment and Java Data Types.

THEORY

JAVA

Java is a set of several computer software products and specifications from **Sun Microsystems** (which has since merged with **Oracle Corporation**), that together provide a system for developing application software and deploying it in a cross-platform computing environment. Java is used in a wide variety of computing platforms from embedded devices and mobile phones on the low end, to enterprise servers and supercomputers on the high end. While less common, Java applets are sometimes used to provide improved and secure functions while browsing the World Wide Web on desktop computers. Java is object oriented programming language. OOP is a programming methodology that helps organize complex programs through the use of inheritance, encapsulation, and Polymorphism. Current version of java is **Java SE 7 Update 25**, Code named Dolphin and released on July 28, 2011.

Platform

An edition of the *Java platform* is the name for a bundle of related programs from Sun that allow for developing and running programs written in the Java programming language. The platform is not specific to any one processor or operating system, but rather an execution engine (called a virtual machine) and a compiler with a set of libraries that are implemented for various hardware and operating systems so that Java programs can run identically on all of them.

- **Java Card**: A technology that allows small Java-based applications (applets) to be run securely on smart cards and similar small-memory devices.
- **Java ME** (Micro Edition): Specifies several different sets of libraries (known as profiles) for devices with limited storage, display, and power capacities. Often used to develop applications for mobile devices, PDAs, TV set-top boxes, and printers.
- **Java SE** (Standard Edition): For general-purpose use on desktop PCs, servers and similar devices.
- **Java EE** (Enterprise Edition): Java SE plus various APIs useful for multi-tier client–server enterprise applications.

A First Simple Program

Let's start by compiling and running the short sample program shown here. As you will see, this involves a little more work than you might imagine.

```
/*
This is a simple Java program.
Call this file "Example.java".
*/

public class Example
{
     public static void main(String args[]) //Your program begins with a call to main().
    {
        System.out.println("This is a simple Java program.");
    }
 }
```

The first thing that you must learn about Java is that the name you give to a source file is very important. For this example, the name of the source file should be same as you give to class. Here, file name of this source code is **Example.java**.

Compiling the Program

```
javac Example.java
```

Running the Program

```
java Example
```

Output

```
This is a simple Java program
```

Explanation

- Open notepad -> Type Code -> Save your source file with extension .java to the folder "C:\jdk1.6\bin"

- ***multiline comment***. This type of comment must begin with **/*** and end with ***/**.
- ***single-line comment*** // Your program begins with a call to main().

- ***class Example {***

This line uses the keyword **class** to declare that a new class is being defined. **Example** is an *identifier* that is the name of the class.

- ***public static void main(String args[]) {***

This line begins the main( ) method. As the comment preceding it suggests, this is the line at which the program will begin executing. All Java applications begin execution by calling main( ).

- The keyword ***void*** simply tells the compiler that **main( )** does not return a value.

- Keep in mind that Java is case-sensitive

Format Specifiers

The general syntax of a format specifier is

%[*flags*][*width*][*.precision*][*argsize*]*typechar*

JAVA Data Types

Java defines eight simple (or elemental) types of data: **byte**, **short**, **int**, **long**, **char**, **float**, **double**, and **boolean**. These can be put in four groups:

- **Integers** This group includes **byte**, **short**, **int**, and **long**, which are for whole valued signed numbers.

- **Floating-point numbers** This group includes **float** and **double**, which represent numbers with fractional precision.

- **Characters** This group includes **char**, which represents symbols in a character set, like letters and numbers.

- **Boolean** This group includes **boolean**, which is a special type for representing true/false values.

Integers

Name	Width	Range
long	64	–9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
int	32	–2,147,483,648 to 2,147,483,647
short	16	–32,768 to 32,767
byte	8	–128 to 127

long

long is a signed 64-bit type and is useful for those occasions where an **int** type is not large enough to hold the desired value. Here is a program that computes the number of miles that light will travel in a specified number of days

```
// Compute distance light travels using long variables.
public class Light
{
   public static void main(String args[])
  {
       int lightspeed;
       long days;
       long seconds;
       long distance;
// approximate speed of light in miles per second
       lightspeed = 186000;
      days = 1000; // specify number of days here
      seconds = days * 24 * 60 * 60; // convert to seconds
      distance = lightspeed * seconds; // compute distance

     System.out.print("In " + days);
     System.out.print(" days light will travel about ");
     System.out.println(distance + " miles.");
  }
}
```

Output

```
In 1000 days light will travel about 16070400000000 miles.
```

Floating-Point Types

Name	Width in Bits	Approximate Range
double	64	4.9e–324 to 1.8e+308
float	32	1.4e–045 to 3.4e+038

double
Here is a short program that uses **double** variables to compute the area of a circle:

```
// Compute the area of a circle.
public class Area
{
    public static void main(String args[])
   {
       double pi, r, a;
       r = 10.8; // radius of circle
     pi = 3.1416; // pi, approximately
     a = pi * r * r; // compute area
     System.out.println("Area of circle is " + a);
   }
}
```

Characters

In Java **char** is a 16-bit type. The range of a **char** is 0 to 65,536. There are no negative **char**s.

Booleans

Java has a simple type, called **boolean**, for logical values. It can have only one of two possible values, **true** or **false**.
Here is a program that demonstrates the **boolean** type:

```
// Demonstrate boolean values.
public class BoolTest
{
    public static void main(String args[])
   {
       boolean b;
       b = false;
       System.out.println("b is " + b);
       b = true;
       System.out.println("b is " + b);
// a boolean value can control the if statement
       if(b)
           System.out.println("This is executed.");
       b = false;
```

```
      if(b)
         System.out.println("This is not executed.");
// outcome of a relational operator is a boolean value
      System.out.println("10 > 9 is " + (10 > 9));
   }
}
```

Output

```
b is false
b is true
This is executed.
10 > 9 is true
```

LAB TASK

1. Execute all programs as define above.

2. Write a program that can take 2 integer values as input between1 to 50000 (or assign them). Add the values and then write the sum of the 2 values in vertical order. Example: If the Inputs are 23459 + 36937; the sum 60396, will be displayed vertically (Each Digit per Line)