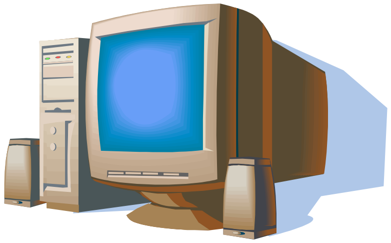
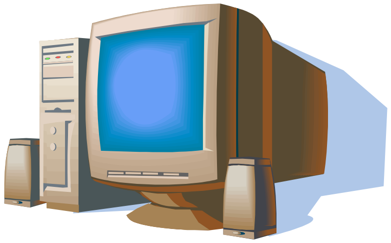


CS360 Client/Server Programming Using Java

TTh 9.25 AM - 10.40 AM

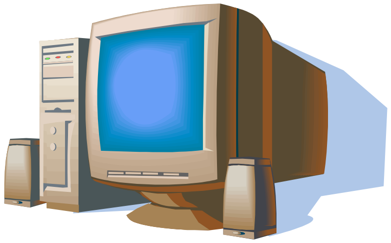
Strain 322

Questions



- What is your definition of client/server programming? Be specific.
- What would you like to learn in this course?

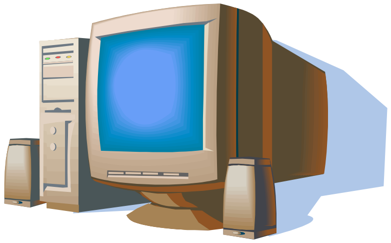
Aims and Objectives



- Or, what will you learn in this course?

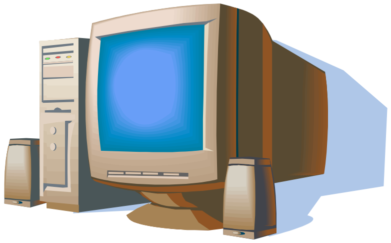
- You Will:
  - Learn the Java programming language
  - Gain more experience in object oriented programming
  - Program a large project from start to finish
  - Learn new concepts such as:
    - ✓ Multithreading
    - ✓ Remote Procedure Calls (RPC)
    - ✓ Exception Handling
    - ✓ Security
    - ✓ Concurrency

Aims and Objectives



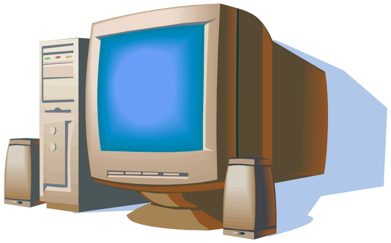
- You will:
  - Use the Java API to look up new concepts
  - Be exposed to new technologies including:
    - ✓ Web servers
    - ✓ MySQL DBMS
    - ✓ XML
    - ✓ PHP
  - Be exposed to the basics of networking
  - Program both clients and servers and have them communicate

Course Details



- Textbooks
  - Primary: Java How to Program (Fifth Edition) by Deitel & Deitel *Prentice Hall*
  - Recommended: Java Network Programming by Elliotte Rusty Harold *O'Reilly*
- Course Website:
  - http://zeus.cs.pacificu.edu/shereen/CS360Spring04/CS360.html

Course Details



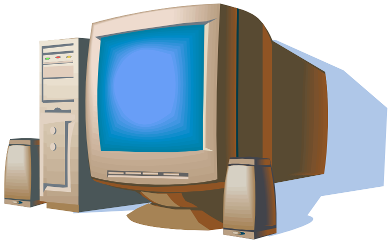
- Tentative Grading:

6 to 7 Programming Assignments	30%
Final Project....................................	10%
2 Exams	35%
1 Final	25%
TOTAL..	**100%**

- Percent Breakdown:

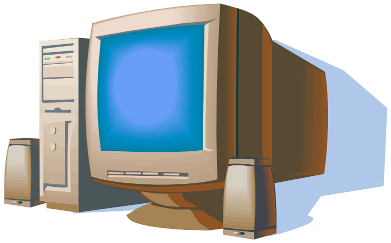
92-100	A	90-92	A-		
88-90	B+	82-88	B	80-82	B-
78-80	C+	72-78	C	70-72	C-
68-70	D+	60-68	D		
0-60	F				

Important Dates



- Tentative dates for Midterms:
  - Midterm 1: Thursday, March 4 (Week 5)
  - Midterm 2: Thursday, April 15 (Week 11)
- Date of Final:
  - Thursday, May 13, 12:00 PM - 2:30 PM (Strain 322)

Class Policies

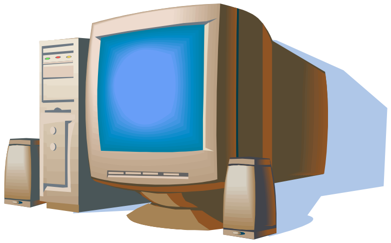


- Programs are to be submitted in the course drop box by 9.25 AM on the day in which the assignment is due.

- Programs can be turned in 24 hours late with a penalty of 20%. Anything later will not be accepted.

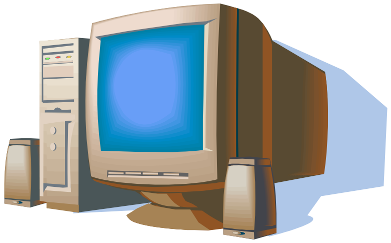
- The coding standards for this course are attached to the syllabus. You must adhere to these coding standards on every assignment to receive full credit.

Class Policies



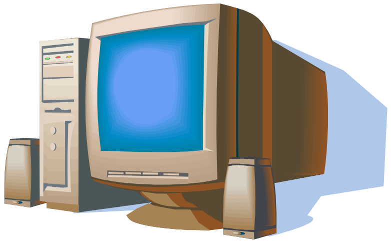
- No early or late exams/finals will be given.
- No incompletes will be given.
- All code in any form generated from this course becomes the intellectual property of Pacific University. You may not share this code with anyone without obtaining written permission from Pacific University.

Class Policies



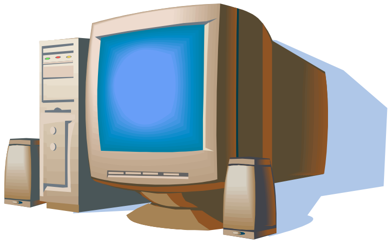
- The cheating policy is defined in Pacific Stuff & the Pacific Catalog as well as the Academic Policy that each of you signed upon entering Pacific University. Be sure you read or reread this policy carefully. All code written for CS360 is to be an original design and an original implementation. The Web, textbooks, and any other references are simply references for you. Copying source code from any source is prohibited. Further, source code is not to exchange hands in any form or by any medium except when sending your solutions to the instructor. It is OK to share high level ideas during the design phase, help someone in the class fix a bug occasionally, share information dealing with OS issues, debugger issues, in general, development issues that do not involve code writing.

Class Policies



- Neither computer failure, software failure, nor lack of computer access are accepted as excuses for late programs; therefore, start work on the programs as soon as they are assigned, don't put them off until the last minute. Further, corruption of programs due to bad disk media is also not accepted as an excuse for late programs; therefore, always keep a current backup of all programs on a separate disk.

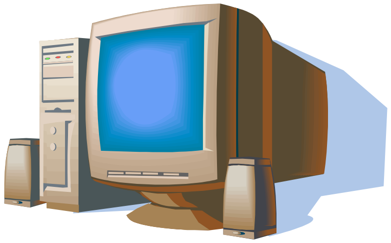
Class Policies



- The instructor reserves the right to raise or lower a student's grade based on class participation and attendance.

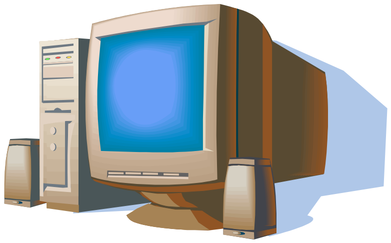
- I do not want to hear any electronic devices go off during lecture; therefore, make sure you silence these devices before lecture starts.

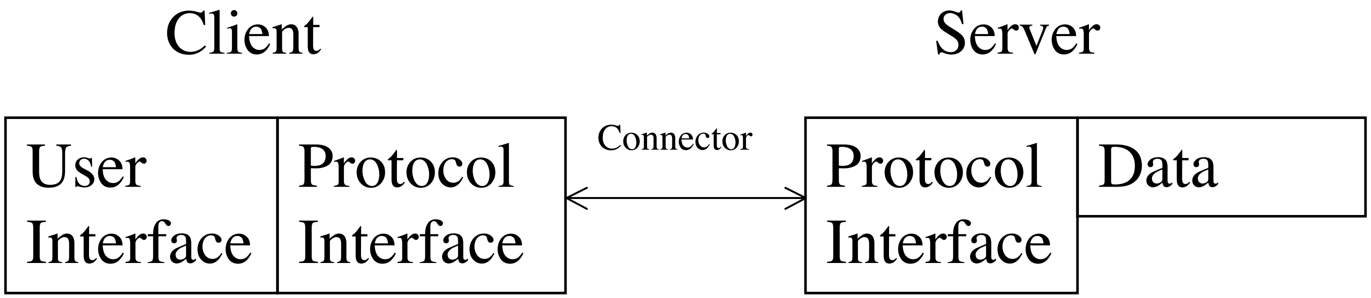
Let Us Begin!



- What do you know about clients and servers?

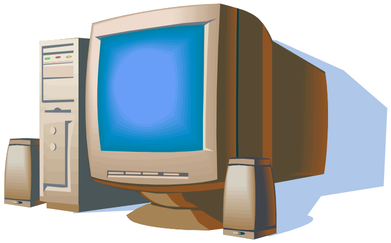
Client-Server





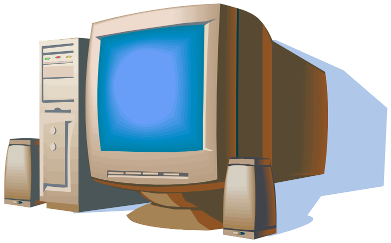
- Client
  - Application that initiates peer-to-peer communication
  - Translate user request into requests for data from server via protocol
  - GUI often used to interact with user
- Server
  - Any program that waits for incoming communication requests from a client
  - Extracts requested information from data and return to client

Common Issues



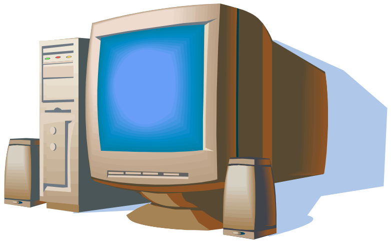
- Authentication
- Authorization
- Data Security
- Privacy
- Protection
- Concurrency

Java



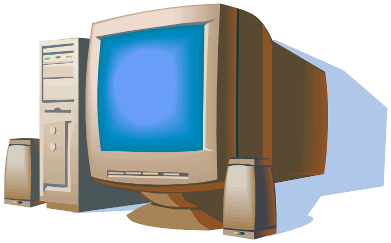
- What do you know about Java?

A Little Bit of History



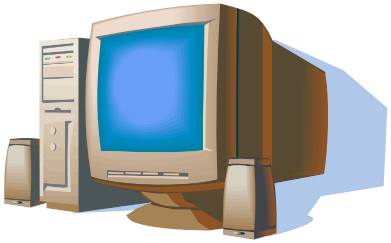
- 1950: FORTRAN
- 1959: COBOL
  - Half business software still programmed in COBOL
- Mid 1960s: Basic
- 1971: Pascal
  - Structured programming language
- 1970s & Early 1980s: Ada

A Little Bit of History



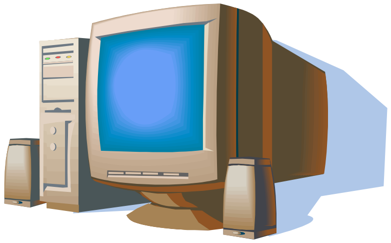
- 1972: C Programming Language
  - What was there before C?
- 1983: ANSII C
- Early 1980’s: C++
  - Object-oriented programming language
  - Examples of other OO languages:
    - ✓ Smalltalk
    - ✓ Java
- 1991: Java (originally Oak) Came Into Being

A Little Bit of History



- 1993: World Wide Web Became Popular
  - Java was saved!
- 1995: Java Formally Announced
- 2000: C#

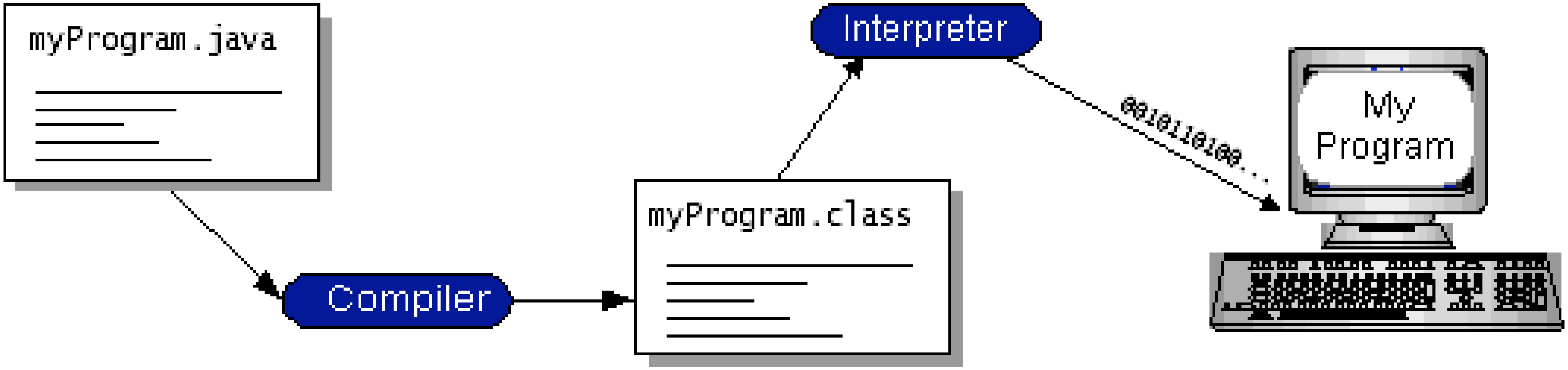
Why Java?



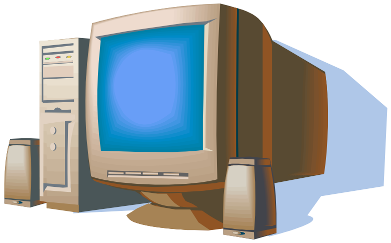
- The Java programming language is robust and versatile, enabling developers to:
  - Write software on one platform and run it on another.
  - Create programs to run within a web browser.
  - Develop server-side applications for online forums, stores, polls, processing HTML forms, and more.
  - Write applications for cell phones, two-way pagers, and other consumer devices.

- It's no wonder the Java platform attracts so many new developers.

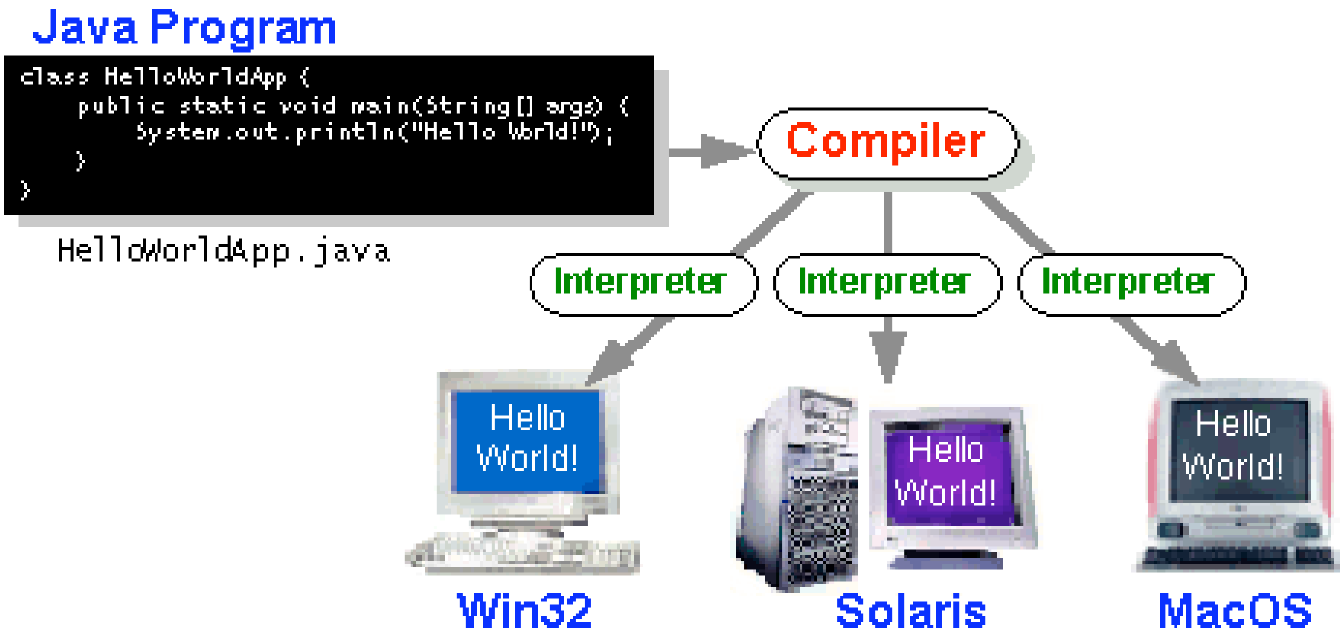
Java


myProgram.java
Compiler
myProgram.class
Interpreter
00101101100...
My
Program

Java

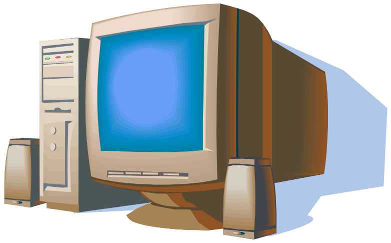


- "write once, run anywhere"



```
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

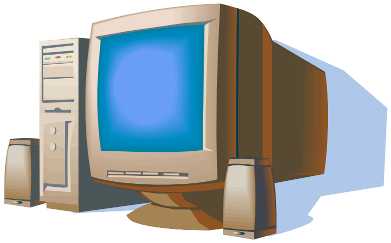
Java Programs



- Consist of:
  - Classes
  - Methods (or functions)

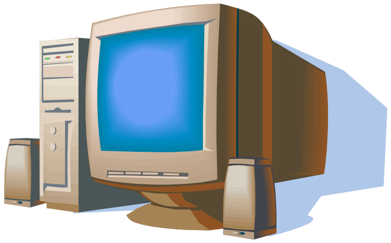
- You could:
  - Create everything yourself (becomes tedious after a time)
  - Use Java class libraries (Java APIs)

Java Programs



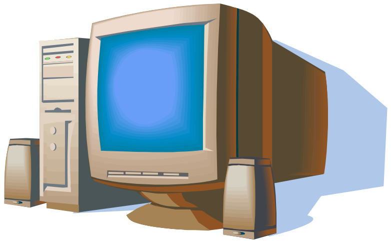
- You can create and run Java programs on any platform for free
- Download and Install Java SDK from
  - http://java.sun.com
- The latest version is Java 1.4.2
  - This is what is installed in the lab
- The book uses Java 1.4.1
  - This will not make any difference

Java Programs



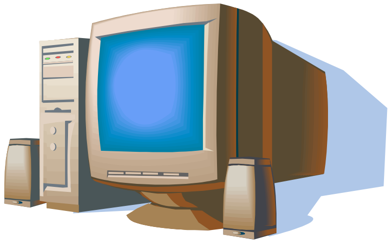
- Create or write your Java programs using any text editor. It is always best to use a text editor that recognizes Java instructions.
  - I would recommend that you use the CodeWarrior editor that is installed on the machines in the lab
- Save your programs with a .java extension
  - `welcome.java`
- Compile your java program in the command window by typing
  - `javac welcome.java`
  - You will need to go to the directory containing your file

Java Programs



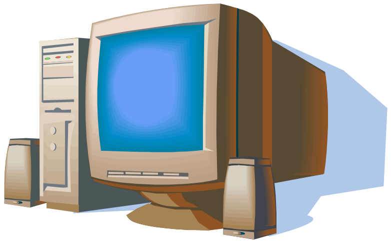
- Compiling Java programs creates a new file with the same name and the extension .class
  - `welcome.class`
- This file contains the bytecode that will be interpreted during the next step
- You run the Java program by typing
  - `java welcome`
- In this stage the program is loaded into memory and executed

Applets vs. Applications



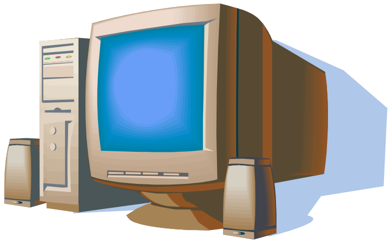
- An application is a program that is normally stored and executed from a user’s personal computer
- All programs that you have created in CS150 and CS250 are applications
- Applets are small programs that are normally stored on a remote computer and users connect to these programs via the WWW browser

Essentials of Java Programs



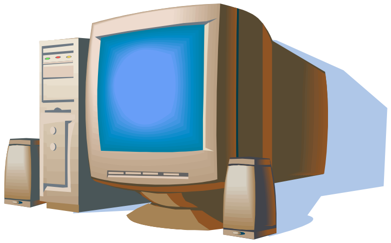
```java
// Fig. 2.1: Welcome1.java
// Text-printing program.

public class Welcome1 {

   // main method begins execution of Java application
   public static void main( String args[] )
   {
      System.out.println("Hello " + args[0] + " " + args[1]);
      System.out.println( "Welcome to Java Programming!" );

   } // end method main

} // end class Welcome1
```

Essentials of Java Programs



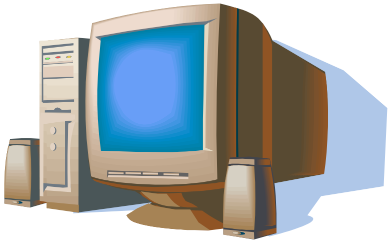
- Every program must have at least one class, and this class name must be the same as the file name
  - Case sensitive
- Every program must have a main method, but there should only ever be one main method per program
- Comments are the same as in C++
- `System.out.println` outputs whatever is in the parenthesis to the standard output (prompt)

GUI



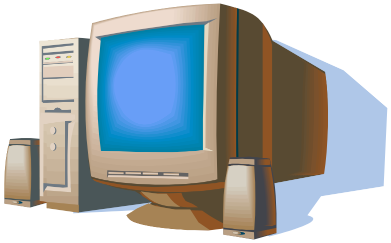
- Since Java has an abundance of ready made classes (the API), it is easy to use these classes to write programs that use a graphical user interface rather than the standard output

Java GUI Program



```
import javax.swing.JOptionPane;

public class Welcome4 {

    public static void main( String args[] )
    {
        JOptionPane.showMessageDialog( null,
  "Welcome\nto\nJava\nProgramming" + args[0]);

        System.exit( 0 );

    }
}
```

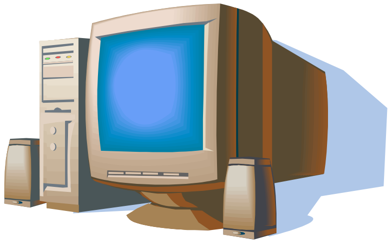
Variables in Java



- Java contains similar variables to those you find in C++
  - int, float, char, String

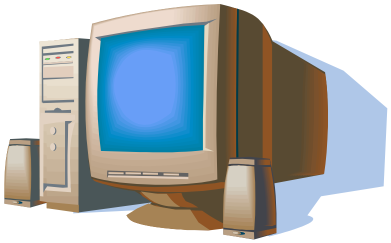
- Java uses the same arithmetic and comparison operators as C++
  - `+ - * / %`
  - `> < >= <= == !=`

Program



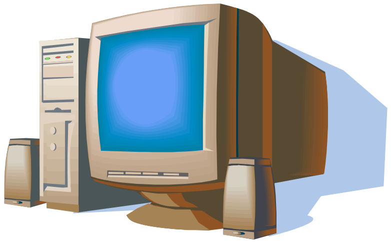
- Write a program that reads in two numbers from the user and sums them

Addition Program



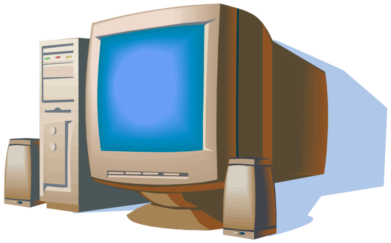
```
import javax.swing.JOptionPane;   // program uses JOptionPane

public class Addition {

public static void main( String args[] )
   {
      String firstNumber;   // first string entered by user
      String secondNumber;  // second string entered by user

      int number1;          // first number to add
      int number2;          // second number to add
      int sum;              // sum of number1 and number2

      firstNumber =
         JOptionPane.showInputDialog( "Enter first integer" );

      secondNumber =
         JOptionPane.showInputDialog( "Enter second integer" );
```

Addition Program



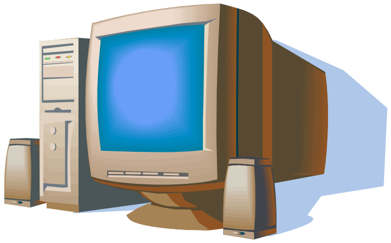
```
     number1 = Integer.parseInt( firstNumber );
      number2 = Integer.parseInt( secondNumber );

     sum = number1 + number2;

     JOptionPane.showMessageDialog( null,
                    "The sum is " + sum,
                    "Results",
                    JOptionPane.INFORMATION_MESSAGE );

      System.exit( 0 );   // terminate application

   } // end method main

} // end class Addition
```

Calculating Average



- Write a program that will allow the user to enter 10 numbers and then calculates the average.

Summary



- Today we covered:
  - How to create, compile and run Java programs
  - How to write to the standard output
  - How to read and write from a simple GUI

- We covered chapter 1 and chapter 2 from the textbook

- Next time:
  - We will program our first Java applet