Welcome to CSE 142!

Benson Limketkai and Marty Stepp

University of Washington, Spring 2010

Who cares?
Who cares?

What is computer science?

- computers?
- science?
- programming?
- late lonely nights in front of the computer?

**ALGORITHMIC THINKING**

_al-go·rithm:_  
a step-by-step procedure for solving a problem or accomplishing some end *especially by a computer*
Fields of computer science

- Graphics
- Computer Vision
- Artificial Intelligence
- Robotics
- Data Mining
- Natural Language Processing
- User Interfaces
- ...

How does this all relate to programming?
  - This course is “Introduction to Programming I” after all.

What is programming?

- **program**: A set of instructions to be carried out by a computer.

- **program execution**: The act of carrying out the instructions contained in a program.

- **programming language**: A systematic set of rules used to describe computations in a format that is editable by humans.
  - We will be studying a programming language called Java.
Programming is like Legos...
Should you take this course?

- No
  - “I hate computers.”
  - “I don’t pay attention to details.”
    - Programming is fairly detail-oriented.
  - “I refuse to think logically.”
  - “I want to take an easy class.”
    - Hard for those who find difficulty in logical thinking and who don’t pay attention to details.
Should you take this course?

- Probably not
  - “I want free gourmet meals and to make lots of money by working for Google.”
  - “World of Warcraft rocks hardcore!”
  - “Everyone, look at my Facebook farm!”

- Yes
  - “I have to take this class.”
    - Is this the only reason? Are you pursuing the right major?
    - “I like to solve problems.”
    - “Computers and robots are going to take over the world. I want to befriend them so that my life will be spared.”
How to do well in this course

• Keep up with the assignments
  • The course material is cumulative
  • From a former student: “Procrastination will eventually come around to bite you in the ass!"

• If you don’t understand something, ask questions (especially “WHY?”).
  • “There’s no such thing as a dumb question.”
  • Computers are neither magical nor mysterious. Everything can be explained!

Building Java Programs

Chapter 1
Lecture 1-1: Introduction; Basic Java Programs

reading: 1.1 - 1.3
Your first Java program!

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

• File must be named Hello.java

• What does this code output (print to the user) when you run (execute) it?

Running a program

1. Write it.
   • code or source code: The set of instructions in a program.

2. Compile it.
   • compile: Translate a program from one language to another.
   • byte code: The Java compiler converts your code into a format named byte code that runs on many computer types.

3. Run (execute) it.
   • output: The messages printed to the user by a program.
Bigger Java program!

```java
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello, world!");
        System.out.println();
        System.out.println("This program produces");
        System.out.println("four lines of output");
    }
}
```

- Its output:
  
  Hello, world!
  
  This program produces
  
  four lines of output

- **console**: Text box into which the program's output is printed.

Structure of a Java program

public class `<name>` {
    class: a program

    public static void main(String[] args) {
        method: a named group of statements
        <statement>;
        <statement>;
        ... 
        <statement>;
    }
}

- Every executable Java program consists of a **class**, that contains a **method** named **main**, that contains the **statements** (commands) to be executed.
System.out.println

- A statement that prints a line of output on the console.
  - pronounced "print-linn"

- **Two ways to use** `System.out.println`:
  - `System.out.println("<text>")`;
    - Prints the given message as output.
  - `System.out.println();`
    - Prints a blank line of output.

Syntax

- **syntax**: The set of legal structures and commands that can be used in a particular language.
  - Every basic Java statement ends with a semicolon ;
  - The contents of a class or method occur between { and }

- **syntax error (compiler error)**: A problem in the structure of a program that causes the compiler to fail.
  - Missing semicolon
  - Too many or too few { } braces
  - Class and file names do not match
  - ...
Syntax error example

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

• Compiler output:

```
Hello.java:2: <identifier> expected
    pooblic static void main(String[] args) {
    ^
Hello.java:3: ';' expected
    ^
2 errors
```

• The compiler shows the line number where it found the error.
• The error messages sometimes can be tough to understand:
  • Why can’t the computer just say “You misspelled ‘public’”?

First lesson in this class

• Computers are stupid.

• Computers can’t read minds.

• Computers don’t make mistakes.

• If the computer is not doing what you want, it’s because YOU made a mistake.
More on syntax errors

- Java is case-sensitive
  - Hello and hello are not the same

```
1 Public class Hello {
2   public static void main(String[] args) {
3     System.out.println("Hello, world!");
4   }
5 }
```

compiler output:
```
Hello.java:1: class, interface, or enum expected
Public class Hello {
    ^
1 error
```

Names and identifiers

- You must give your program a name.

```
public class GangstaRap {
```

- Naming convention: capitalize each word (e.g. MyClassName)
- Your program's file must match exactly (GangstaRap.java)
  - includes capitalization (remember, Java is "case-sensitive")

- identifier: A name given to an item in your program.
  - must start with a letter, underscore (_) or $
  - subsequent characters can be any of those or a number

- legal: _myName TheCure ANSWER_IS_42 $bling$
- illegal: me+u 49ers side-swipe Ph.D's
Keywords

- **keyword**: An identifier that you cannot use because it already has a reserved meaning in Java.

```
abstract    default    if           private      this
boolean     do         implements   protected    throw
break       double     import      public       throws
byte        else       instanceof  return       transient
case        extends    int          short        try
catch       final      interface    static       void
class       float      native      super        while
char        finally    long         strictfp     volatile
class       float      native      super        while
continue    goto       package     synchronized
```

- NB: Because Java is case-sensitive, you could technically use `Class` or `cLaSs` as identifiers, but this is very confusing and thus strongly discouraged.

Strings

- **string**: A sequence of text characters.
  - Starts and ends with a " (quotation mark character).
    - The quotes do not appear in the output.
  - Examples:
    
    "hello"
    "This is a string. It's very long!"

- Restrictions:
  - May not span multiple lines.
    "This is not a legal String."
  - May not contain a " character.
    "This is not a "legal" String either."
  - This begs the question...
Escape sequences

• **escape sequence**: A special sequence of characters used to represent certain special characters in a string.

  \t  tab character
  \n  new line character
  \"  quotation mark character
  \\  backslash character

• Example:
  System.out.println("\hello\n\t\r\n\"you"?\\");

• Output:
  \hello
  how is "you"?\n
Questions

• What is the output of the following `println` statements?

  System.out.println("\ta\tb\tc");
  System.out.println("\\\n\\")
  System.out.println("\n");
  System.out.println("\\\n\\")
  System.out.println("C:\nin\the downward spiral");

• Write a `println` statement to produce this output:

  / \ // \ /// \ /// \ \\
Answers

• **Output of each `println` statement:**

```
<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>\</td>
<td></td>
<td>/</td>
</tr>
</tbody>
</table>
| / \ // \// \\
```

C: in he downward spiral

• **`println` statement to produce the line of output:**

```java
System.out.println("/ \ / \ \ / / \ \ / / \ / /");
```

Questions

• **What `println` statements will generate this output?**

This program prints a quote from the Gettysburg Address.

"Four score and seven years ago, our 'fore fathers' brought forth on this continent a new nation."

• **What `println` statements will generate this output?**

A "quoted" String is 'much' better if you learn the rules of "escape sequences."

Also, "" represents an empty String. Don't forget: use "\" instead of " ! "' is not the same as ""
• **println statements to generate the output:**

```java
System.out.println("This program prints a");
System.out.println("quote from the Gettysburg Address.");
System.out.println();
System.out.println("\"Four score and seven years ago,\")
System.out.println("our 'fore fathers' brought forth on");
System.out.println("this continent a new nation.\"\")
```

• **println statements to generate the output:**

```java
System.out.println("A \"quoted\" String is");
System.out.println("'much' better if you learn");
System.out.println("the rules of \"escape sequences.\"");
System.out.println();
System.out.println("Also, \"\" represents an empty String.");
System.out.println("Don't forget: use \\" instead of \" !");
System.out.println("'\' is not the same as \"");
```