Welcome to CSE 142!

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.

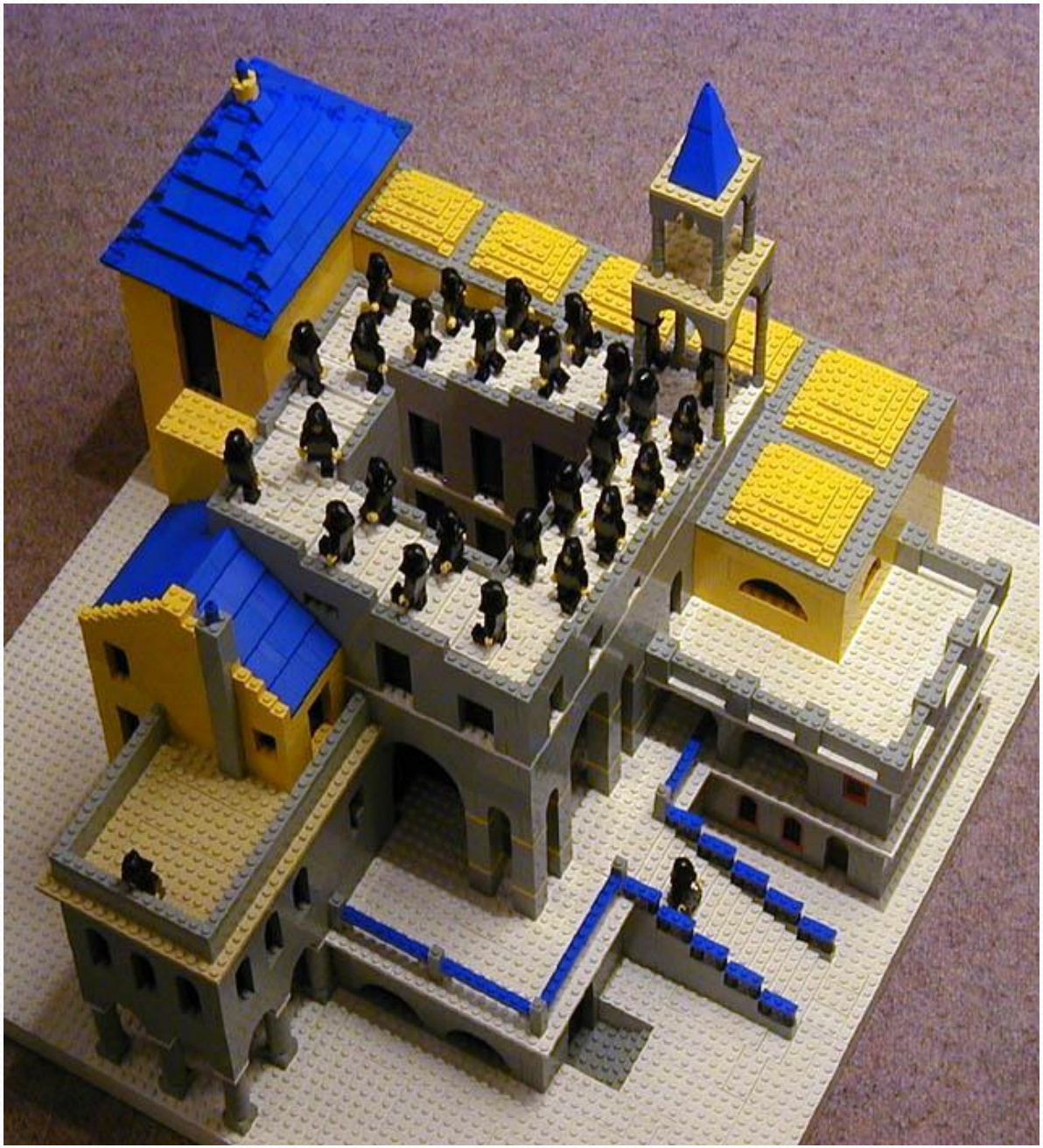
What is programming?

- **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."
  - "Candy Crush is awesome!"
  - "If I wrote WhatsApp, I would have made how many billion dollars???"
- 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."

Tips for Success

- Visit website often: http://cs.washington.edu/142
- Read syllabus carefully
- Do lots of problems on http://practiceit.cs.washington.edu/
- If you're stuck, review lecture and book examples
- Remember: assignments must be **your own work**!

Tips for Success (cont'd)

- 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

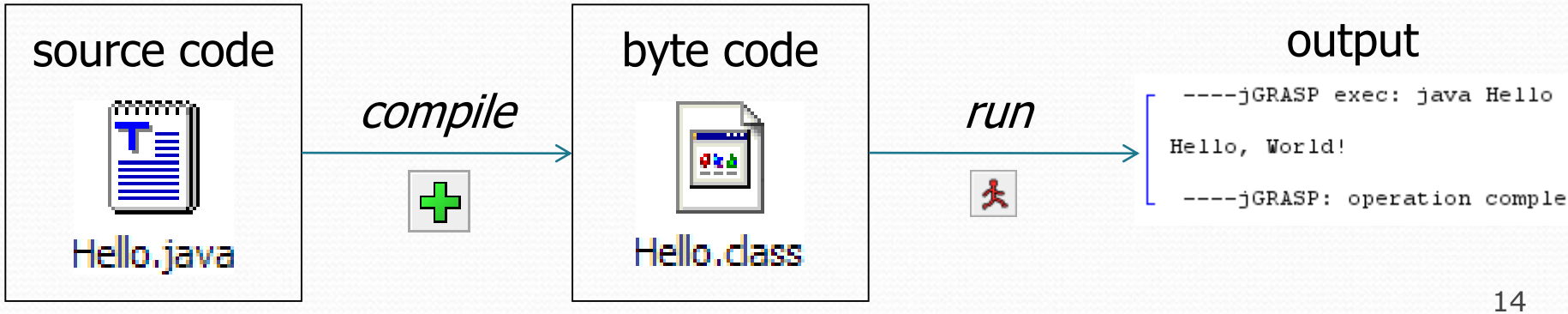
Your first Java program!

```
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!

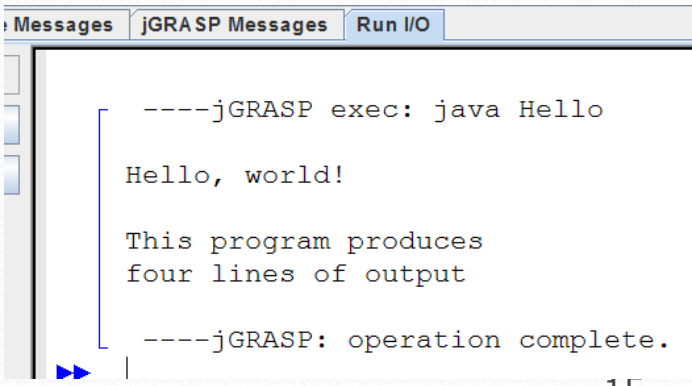
```
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello, world!");
        System.out.println();
        System.out.println("This program produces");
        System.out.println("four lines of output");
    }
}
```

- Its output:

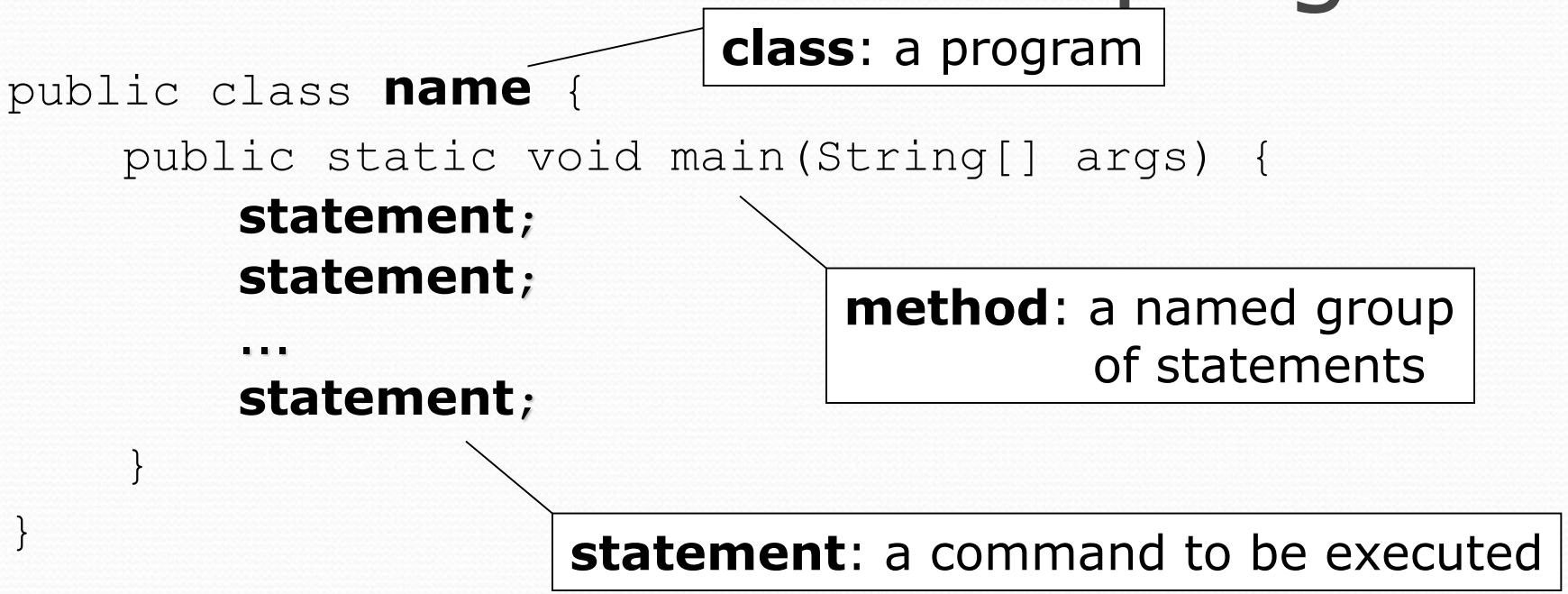
```
Hello, world!

This program produces
four lines of output
```

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



Structure of a Java program



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

Names and identifiers

- You must give your program a name.

```
public class HelloWorld {
```

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

- **identifier**: A name given to an item in your program.
  - must start with a letter or `_` 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**
char	finally	long	strictfp	volatile
class	float	native	super	while
const	for	new	switch	
continue	goto	package	synchronized	

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

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.
  - The "spelling" and "grammar" of a programming 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

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

- 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 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
```

Strings and escape sequences

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\nhow\tare \"you\"?\\\\");
```

- Output:

```
\hello
how     are "you"?\\
```

Questions

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

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

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

```
/ \ // \\ /// \\\
```

Answers

- Output of each `println` statement:

```
        a       b       c
\\
'
"""
C:
in      he downward spiral
```

- `println` statement to produce the line of output:

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

Questions

- What `println` statements will generate this output?

```
This quote is from
Irish poet Oscar Wilde:

"Music makes one feel so romantic
- at least it always gets on one's nerves -
which is the same thing nowadays."
```

- 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 "
```

Answers

- `println` statements to generate the output:

```
System.out.println("This quote is from");
System.out.println("Irish poet Oscar Wilde:”);
System.out.println();
System.out.println("\"Music makes one feel so romantic");
System.out.println("- at least it always gets on one's nerves -");
System.out.println("which is the same thing nowadays.\"");
```

- `println` statements to generate the output:

```
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 \"");
```