ICS 111
Introduction to Computer Science I

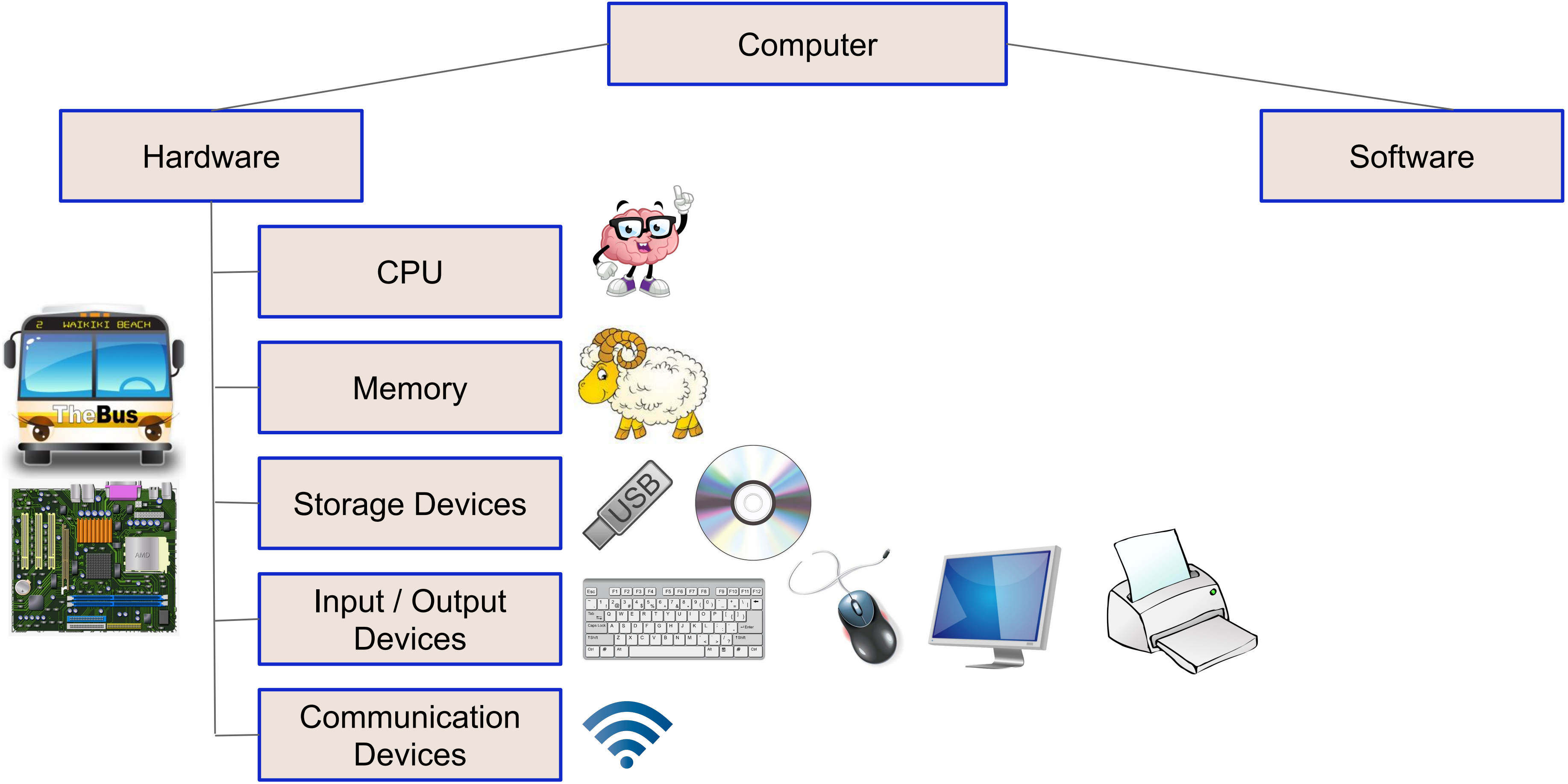
Nikki Manuel
Leeward Community College

Introduction to Computers

Week 1

What is a computer?

An electronic device
that stores
and processes
data



Computer

Hardware

Software

CPU

Memory

Storage Devices

Input / Output Devices

Communication Devices

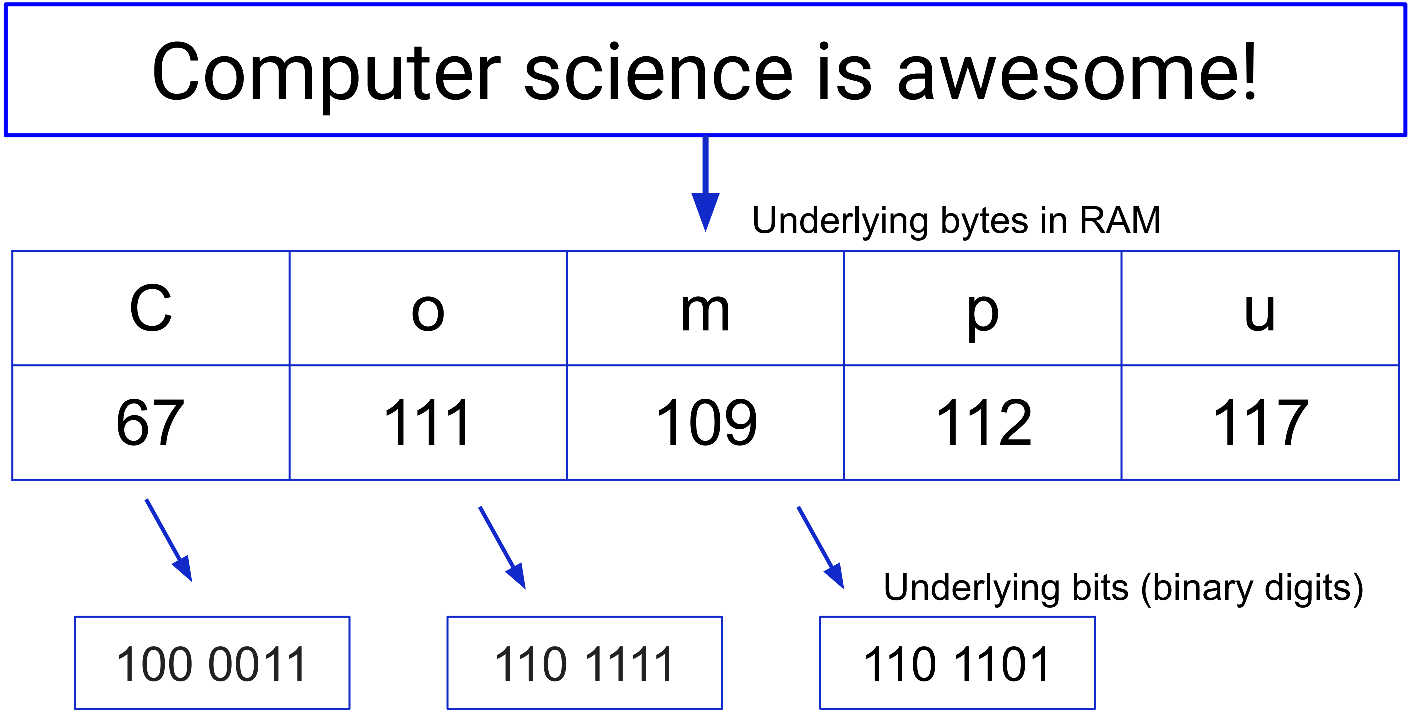
How is information stored in a computer?

- A computer = a series of switches
- Each switch has two states: OFF and ON
- OFF = 0, ON = 1
- These 0s and 1s are called **bits** (binary digits)
- 1 **byte** = 8 bits
- Data is encoded as a series of bytes
- 1 byte = one character (‘A’)

Storing Data – Bits, Bytes, and ASCII

65	A
66	B
67	C
68	D
69	E
70	F
71	G

More ASCII



Computer science is awesome!

Underlying bytes in RAM

C	o	m	p	u
67	111	109	112	117

Underlying bits (binary digits)

100 0011

110 1111

110 1101

Introduction to Programming

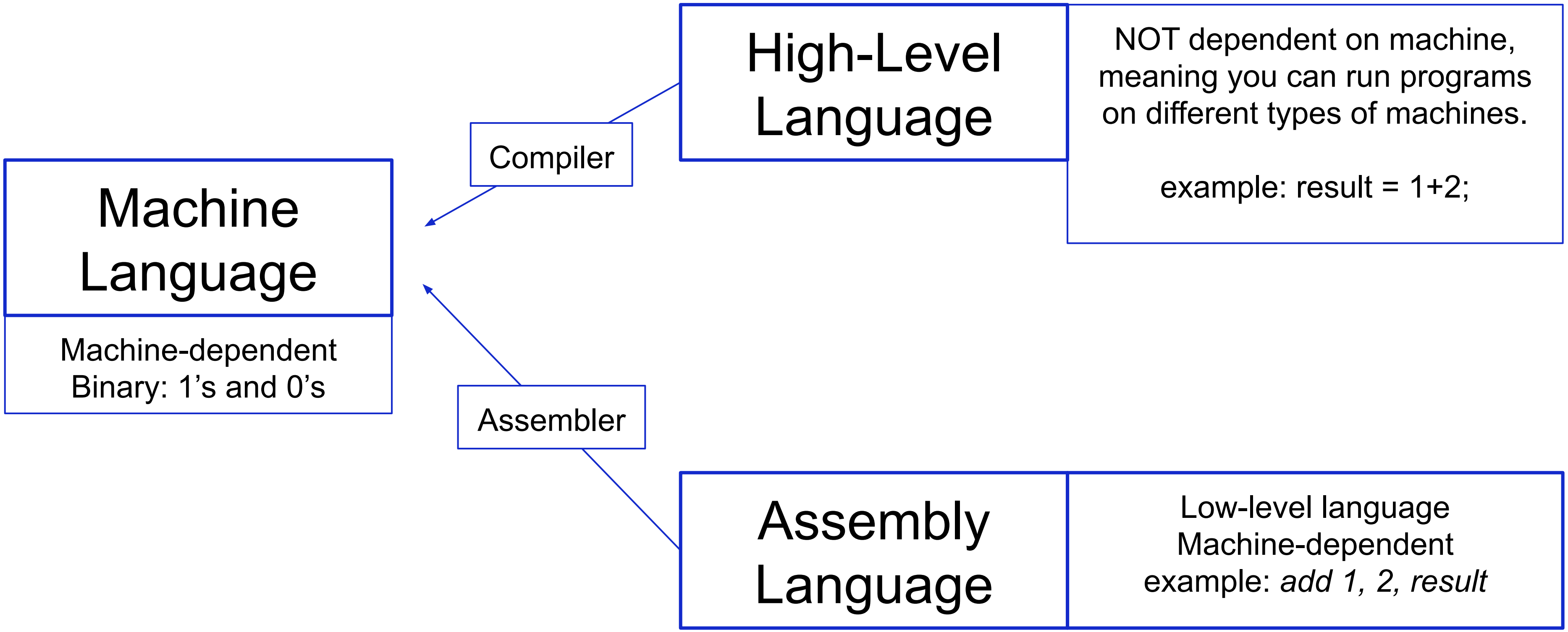
Week 1

What is programming?

To create (or “develop”) software (also called a “program”)

Software: contains instructions that tell the computer what to do

Programming Languages



High-Level Language

NOT dependent on machine, meaning you can run programs on different types of machines.

example: result = 1+2;

Compiler

Machine Language

Machine-dependent
Binary: 1’s and 0’s

Assembler

Assembly Language

Low-level language
Machine-dependent
example: *add 1, 2, result*

Introduction to Java

Week 1

Java JDK

JDK = Java Development Kit

The software for developing and running Java programs.

Runs in the background and translates your instructions into something the computer can understand.

jGrasp

IDE = Integrated Development Environment

Allows you to edit, compile, debug, and get help all in one place.

Terminology: Edit, Compile, Run

Edit: Write a program in Java and store it on the computer

- creates ProgramName.java (text file)

Compile: Check for errors and create Java bytecode

- creates ProgramName.class (bytecode file)

Run: Translate bytecode -> machine code and implement it

- creates screen output

Terminology: Syntax

Syntax: “Grammar” rules to follow. If we don’t follow these rules, the compiler will tell us our mistakes (bugs)

Syntax for a basic Java computer program:

If you write your program in a different way, it may have syntax errors and not compile

```java
public class ProgramName {
    public static void main(String[] args) {
        // Write your program here
    }
}
```

Syntax for a basic Java computer program:

```
public class ProgramName {
    public static void main(String[] args) {
        // Write your program here
    }
}
```

Don’t worry about what all these words mean yet

We will learn later what public, class, [], static, void, args (arguments) mean.

Just know that your program code goes inside the two inner curly brackets, i.e. "{" and "}"

Also, the file name must match the ProgramName

Naming a Program

Program has a name and an extension

- name should be meaningful
- extension is .java

MeaningfulName.java
Meaningful_Name.java

Program Name Rules

- A program name is better in CamelCase
  - MyFirstProgram.java
- A long program name is okay
- No spaces or special characters
  - Underscore is okay
- It cannot begin with a number
  - It may contain numbers

CamelCase

- Lowercase – helloworld
- Uppercase – HELLOWORLD
- Upper camel case
  – HelloWorld
- Lower camel case
  – helloWorld

Valid Names for Programs

MyFirstProgram

Program1

Program2

NotAVirus

WxPHNL2208

ManuelNikki_Assignment2

Meaningful Names

TipCalculator

SalesRecorder

TimeZoneConverter

RandomNumberGenerator

Is this name valid?

3LittlePigs

123_ok

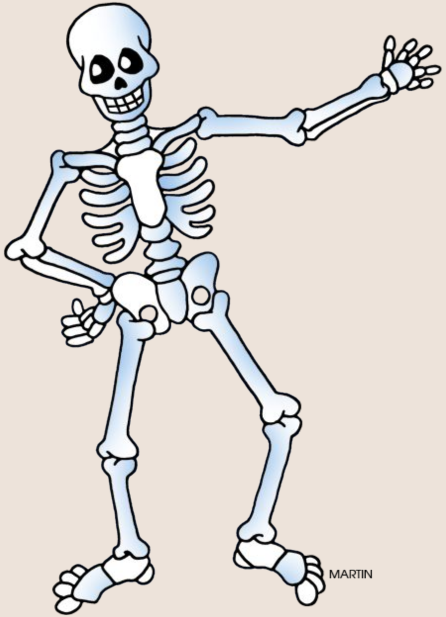
Ok_123

Ok$123

X98721w

MathCalculations7

Program Skeleton



Java Program Skeleton

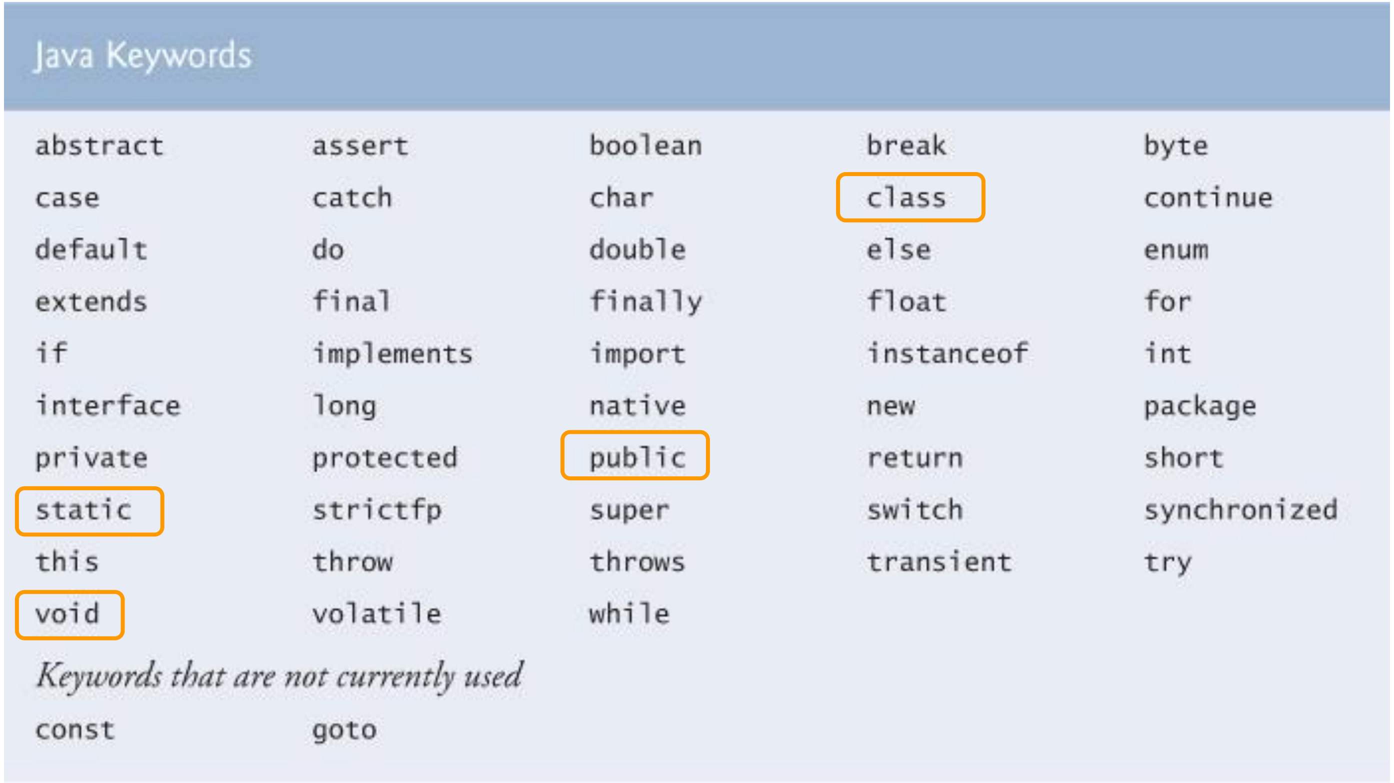
```java
public class ProgramName {
   public static void main(String[] args) {

   }
}
```

Reserved Words aka Keywords

- Have a specific meaning to the compiler
- Cannot be used for other purposes in the program
- When the compiler sees the word **class**, it understands that the word after class is the name for the class.

Reserved Words aka Keywords



Java Keywords

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

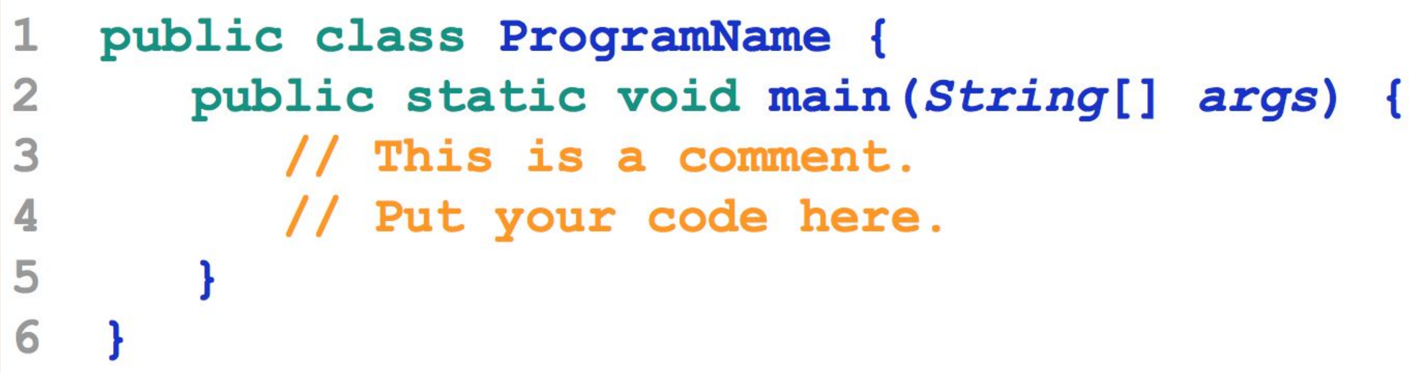
Keywords that are not currently used

const	goto

Java Program Skeleton

```java
public class ProgramName {
   public static void main(String[] args) {
      // This is a comment.
      // Put your code here.
   }
}
```

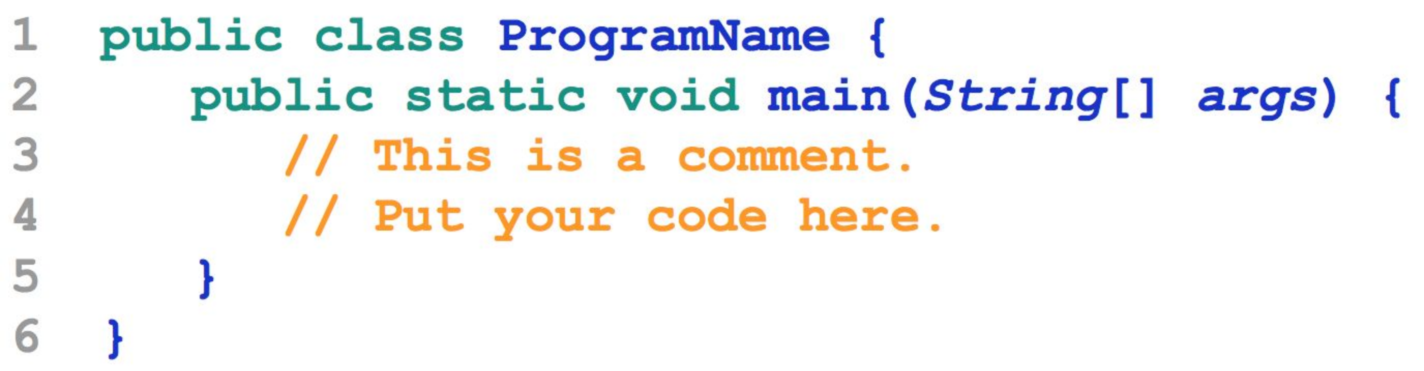
Line 1 - defines the **class**



```
1  public class ProgramName {
2     public static void main(String[] args) {
3        // This is a comment.
4        // Put your code here.
5     }
6  }
```

Every Java program must have at least one class.
Each class has a name.
By convention, class names start with an uppercase letter.
For this example, the class name is ProgramName.

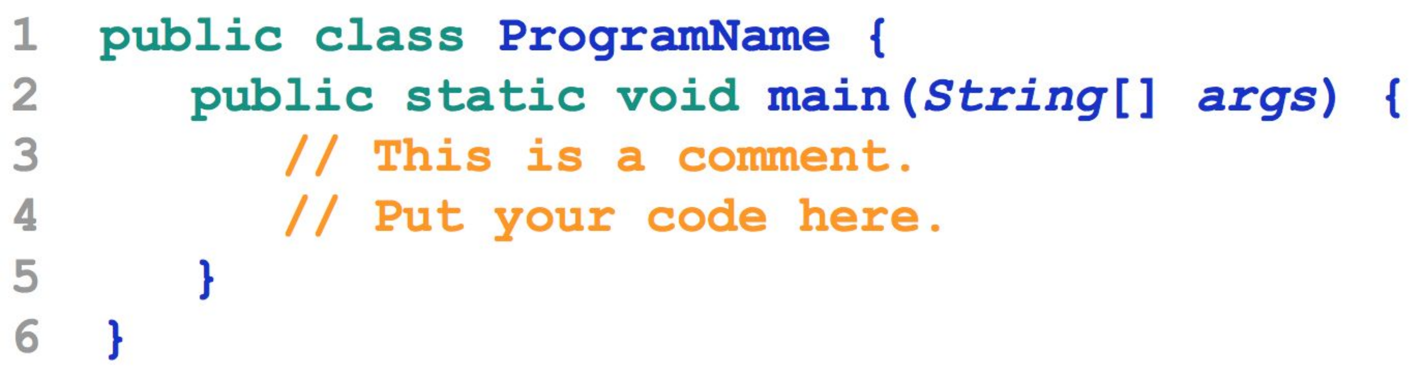
Line 2 - defines the **main method**



```
1  public class ProgramName {
2      public static void main(String[] args) {
3          // This is a comment.
4          // Put your code here.
5      }
6  }
```

The program is executed from the main method.
A class may contain several methods.
A method is a construct that contains statements.
The main method is the entry point where the program begins execution.

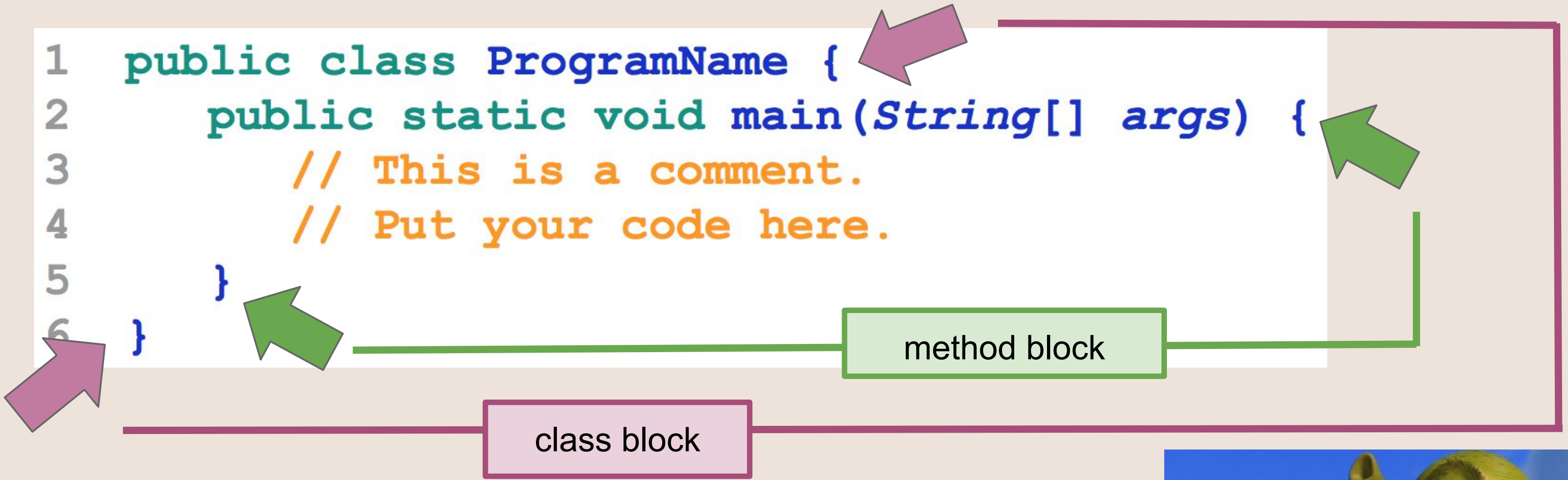
Line 3 and 4 - Comments



```
1  public class ProgramName {
2     public static void main(String[] args) {
3        // This is a comment.
4        // Put your code here.
5     }
6  }
```

Documents what the program is and how it is constructed.
Comments help programmers to communicate and understand the problem.
They are not programming statements and thus are ignored by the compiler.
Line comments start with //
Block comments are between /* and */

Each pair of curly braces forms a **block** that groups the program’s components.



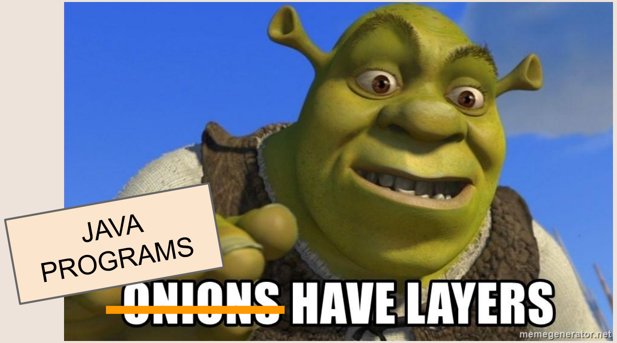
```
public class ProgramName {
    public static void main(String[] args) {
        // This is a comment.
        // Put your code here.
    }
}
```

method block

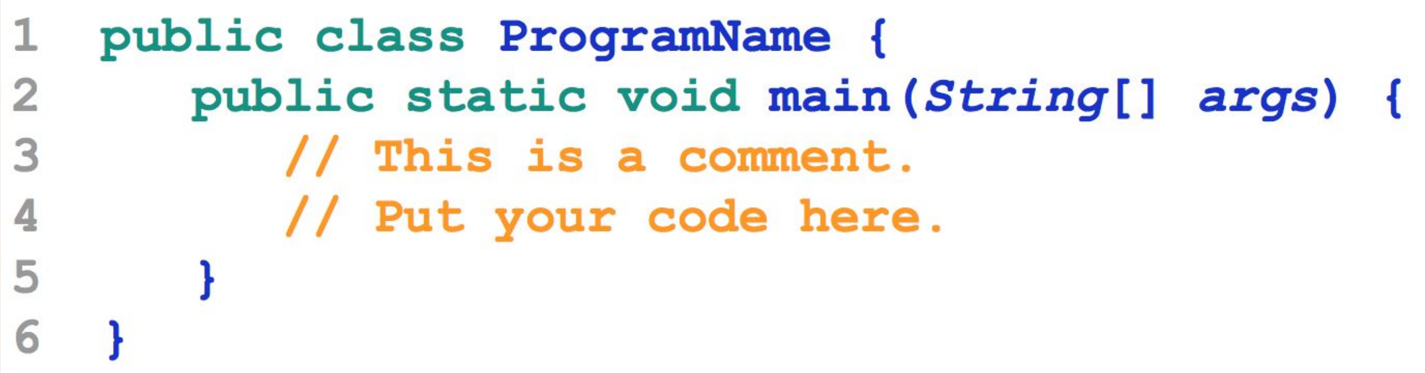
class block

Blocks can be *nested*, meaning that one block can be placed within another, just like the example above: the method block is inside the class block.

If it helps, you can think of these blocks as being different layers.



Remember: Every detail matters!



```java
public class ProgramName {
    public static void main(String[] args) {
        // This is a comment.
        // Put your code here.
    }
}
```

Pay attention to details such as capital letters and symbols such as { }, [], //, ;

In the above example, the letter S in String[] is capitalized.
This is important - if you do not capitalize the S, you will get an error.

Now let’s learn how to output some text.

Printing a Message

- **`System.out`**
  - Sends a message
- **`print`**
  - Prints the message
- **`println`**
  - Prints the message and skips to the next line
  - Similar to writing something and then pressing the “Enter” key.

The Message

- The message must be quoted -- between quotation marks.
- What is inside the quotes will be printed "as is".

Now try it out yourself!

Printing a Message : `print`

```java
public class ProgramName {
   public static void main(String[] args) {
      // This is a comment.
      // Put your code here.
      System.out.print(“Hello Hawaii!”);
      System.out.print(“My name is Lane.”);
   }
}
```

Printing a Message : `println`

```java
public class ProgramName {
    public static void main(String[] args) {
        // This is a comment.
        // Put your code here.
        System.out.println(“Hello Hawaii!”);
        System.out.println(“My name is Lane.”);
    }
}
```

```java
/** * Short description of the program.
 *
 * @author     Last Name, First Name
 * @assignment ICS 111 Assignment XX
 * @date       Today's Date
 * @bugs       Short description of bugs in the program, if any.
 */

    public class ProgramName {
        public static void main(String[] args) {
            // Write your program here
            System.out.println(“Hello World!”);
        }
    }
```

ManuelNikki02.java