Programs & Programming

JC 01.1

(A. Nguyen)

What is a computer program or programming?

- A program is a set of instructions that tell the computer to do a task; e.g., draw a red dot at a certain location on the screen, or total the prices of the items bought, etc.
- The act of designing and writing programs is called programming. Using a computer does not require programming skills

The Anatomy of a Computer

JC 01.2

(A. Nguyen)

Computers

- A computer in the early days used to take the space of a large room, with air-conditioning.
- A computer now takes many forms that many of us are familiar with: desktops, laptops, tablets, smart phone, smart card, etc.
- A computer’s main component is the motherboard, which holds parts of the computer like CPU & memory, and allows communication among them. It also provides connectors to peripherals.

The central processing unit

- The central processing unit (CPU) is the heart of a computer
- The CPU (such as Intel Core Processor) is composed of several hundred of million transistors
- When a program is launched (e.g., when the user clicks on an icon), the CPU loads the program into memory (e.g., from the hard disk), and executes instructions, which likely involves fetching and processing of data such as arithmetic operations

Storage

- Data in **primary storage**, also called **memory** or RAM (random-access memory), is kept intact only when there is power.
- Data in **secondary storage** (such as hard disks or flash drives) persists even when there is no power.

Peripheral devices

- These devices are to for interaction between people & computers.
- Some examples of peripheral devices are keyboard, mouse, printer, speakers, etc.

The Java Programming Language

JC 01.3

(A. Nguyen)

About Java

- Java is a high-level programming language, which is easy for humans to understand. A program in a high-level language is translated into machine code for computers to understand.
- Java creator is James Gosling.
- Java first gained its popularity when it was used in the form of **applets**, the Java code that could be run anywhere on the Internet, in mid 1990s.

Important benefits of Java

- **Safe**: Java was designed to be used safely in browsers, so programs terminate if they try to do something unsafe
- **Portable**: Java programs may be used, *without changes*, in various Operating Systems (such as Windows, UNIX, Linux, or Macintosh operating systems). This is possible because the Java **compiler** translates a Java program into machine code to be run by a program called the Java Virtual Machine (JVM), which simulates a CPU

Java for beginning students

1. It is rather involved to write even a simple Java program
2. There have been many versions of Java since official version 1 in 1997
3. It is not possible to learn everything about Java in one course, especially with the very extensive Java Library – so be patient & continue to learn beyond this course

Development Environment

JC08-01.4

(A. Nguyen)

BlueJ

- BlueJ is an **integrated development environment (IDE)** for Java. So is Eclipse.
- It is "integrated" because it has all the tools ***in one place***: the programmers can type (in the **editor**), compile, and run their programs.

From Java Concepts, 8e, C. Horstmann

Becoming Familiar with Your Programming Environment

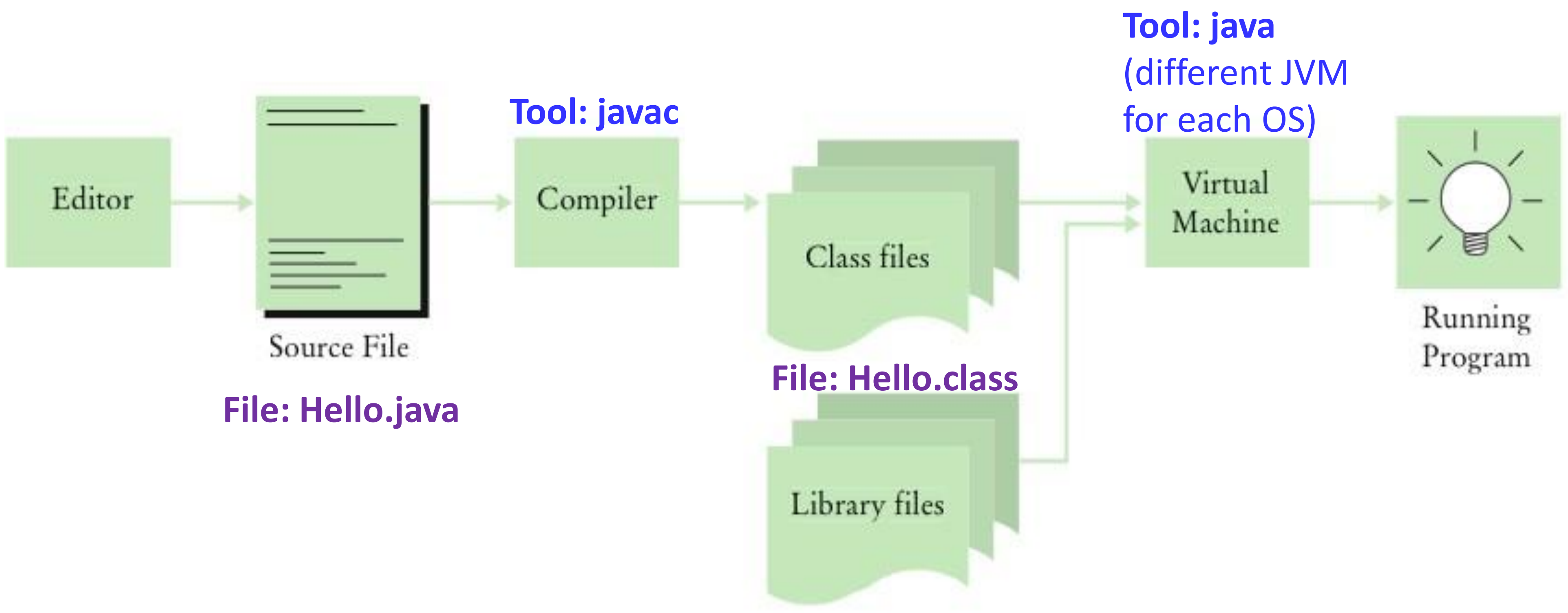


Figure 6 From Source Code to Running Program

Backing up

JC08 01.4

(A. Nguyen)

Why & how to back up

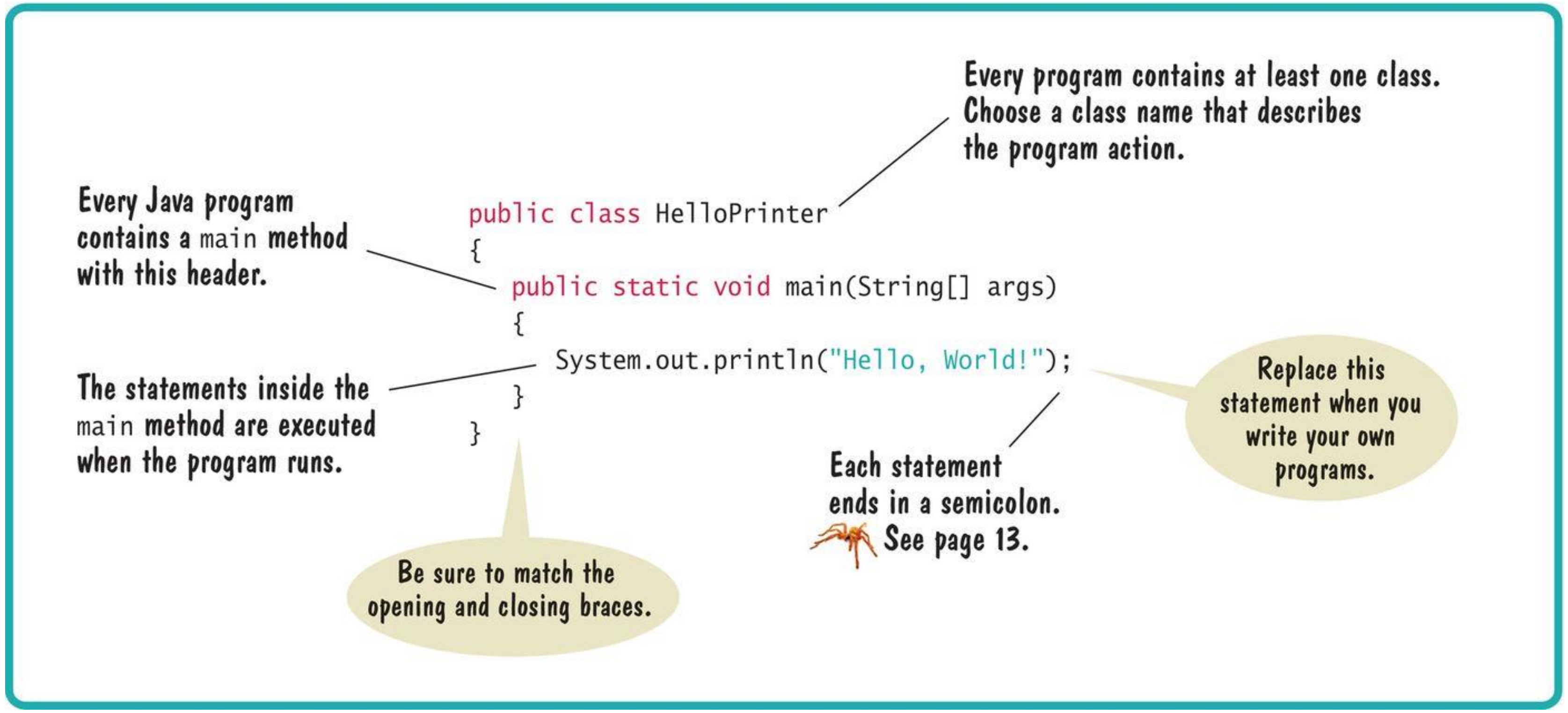
- Back up your files so that you can go back to them if/when the file you are working on is (1) was accidentally deleted, or (2) is somehow corrupt, or (3) is totally messed up with the updates you just made.
- Back up often, perhaps after each day/week.
- Rotate your back-ups; for example, have one back-up for 1 day/week ago, one for 2 days/weeks ago, one for 3 days/weeks ago.
- Be careful: replace the oldest back-up, not the other way around
- Check you back-up once in a while to ascertain that the files are useable.

Your First Program

JC08 01.5

(A. Nguyen)

Syntax 1.1 Java Program



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

Every program contains at least one class. Choose a class name that describes the program action.
Every Java program contains a main method with this header.
The statements inside the main method are executed when the program runs.
Replace this statement when you write your own programs.
Each statement ends in a semicolon. See page 13.
Be sure to match the opening and closing braces.

String

- A Java `String` is a sequence of characters, expressed inside a pair of **double** quotes in Java: `"Greetings!"`
- `System.out.print` or `println` expects **one** parameter/input, such as **one** `String`
- If you have various pieces of text to be printed with 1 statement, you can concatenate them with + (i.e., plus sign). For example, if you have a person’s name in a variable called `name`, you can do:

```
System.out.println("Hello, " + name);
```

- The plus sign is also used for addition, when there are **2 numeric operands**; e.g., `3+4` will result in `7`

Characters

- A Java `char`acter may be typed with a key from the keyboard, or expressed as a hexadecimal value
- A Java `char` is contained in **single** quotes – ***NOTE***: `String`, in **double** quotes
- Example of the upper-case letter **A**:
  - Typed from the keyboard: **`'A'`**
  - Expressed as a hexadecimal value from the Unicode Table: **`'\u0041'`**
- Note that the same letter written inside double quotes is a String of length 1

Characters (cont.)

- Escape sequence is a character that, when typed with a key on the keyboard, will cause confusion in the Java syntax. Thus, it is expressed with a backslash: `\t` (tab), `\n` (new line), `\"` (double quotes).
- Different ways to print blank line:

```
System.out.println("");   // empty String
System.out.println();     // no parameter
System.out.println("\n"); // "new line" escape seq.
```

- Examples of escape sequence:

```
System.out.println("Hello, World\n");
System.out.println("He said \"hi\"");
```

Unicode Characters

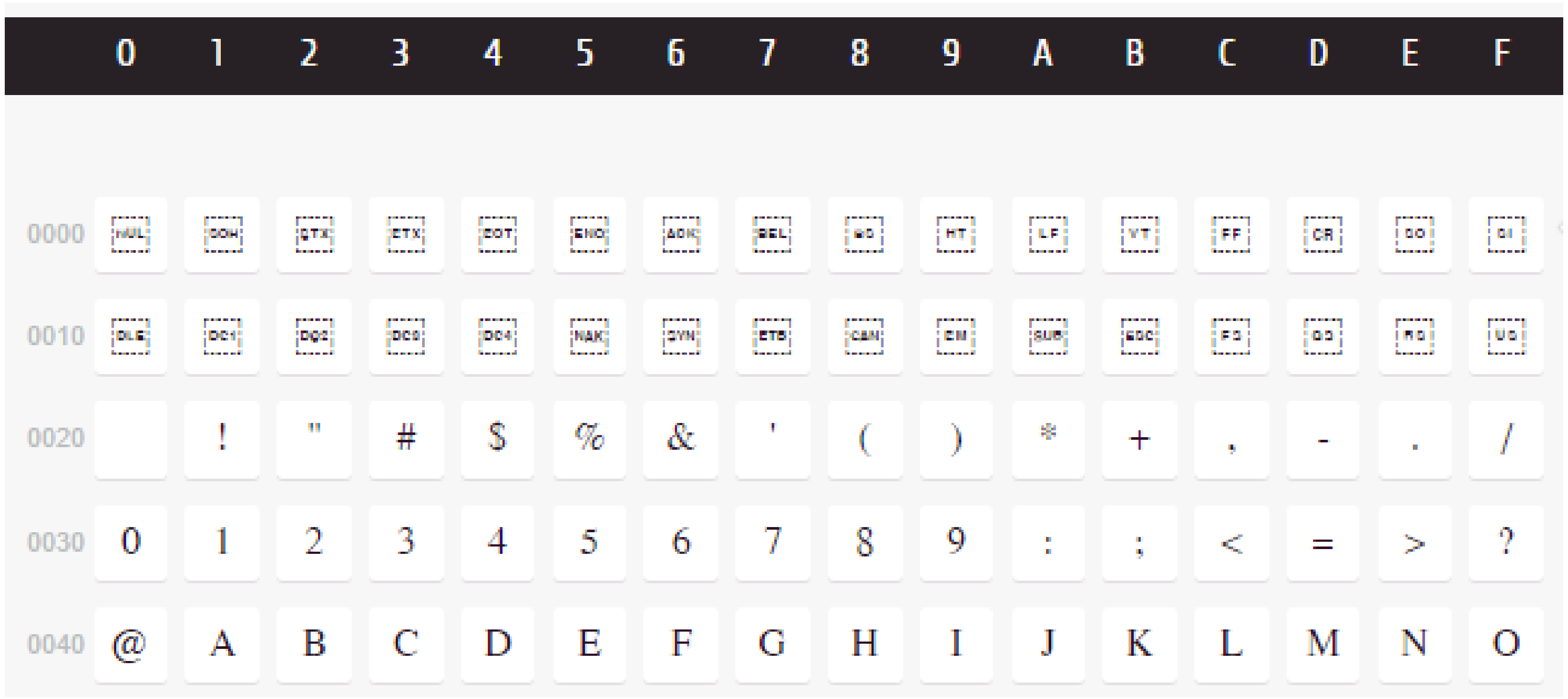
(A. Nguyen)

Unicode Character

- A Unicode character represents a text character, and occupies 2 bytes, or 16 bits
- There are 2^{16} (i.e., 65,536) different possible values
- The original ASCII character set is now part of and at the beginning of the Unicode character set

Unicode Table

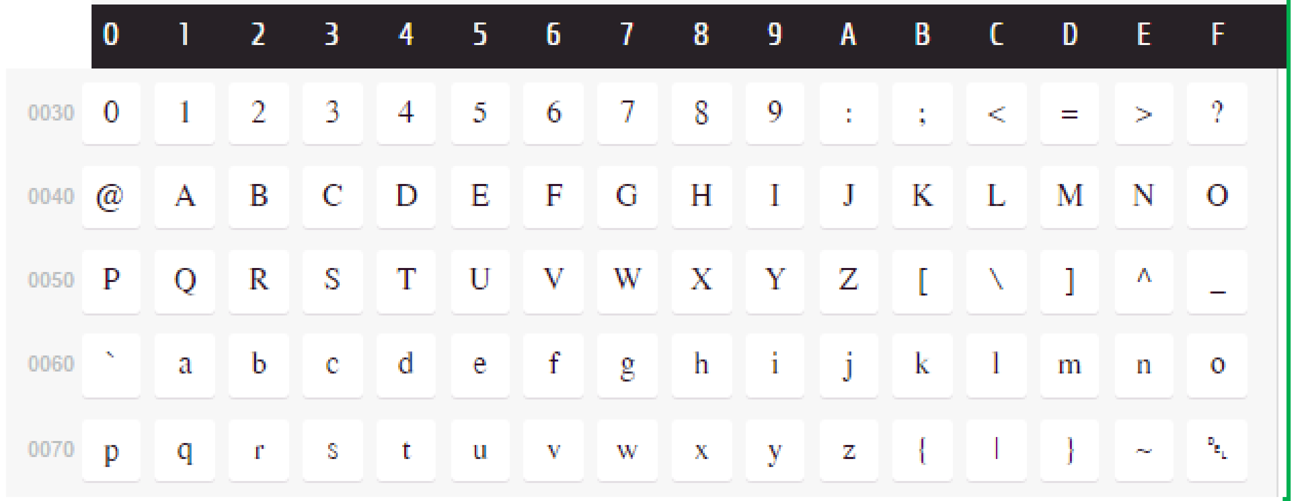
- Search for “unicode table” on the Internet:



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
0010	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
0020		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
0030	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0040	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O

Unicode Table (cont.)

- The first row & the first column (i.e., headers) identify the values of the characters, in hex
- To identify a character/symbol, read the two values and add them; e.g., character **A** is stored as `0040 + 1 =` **`0041`**
- **Note**:
  - **B** is 1 more than **A**
  - **C** is 1 more than **B**, etc.
  - character **1** is stored as **0030**
  - lower-case chars are "larger"



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0030	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0040	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0050	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
0060	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0070	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	␡

Unicode Table (cont.)

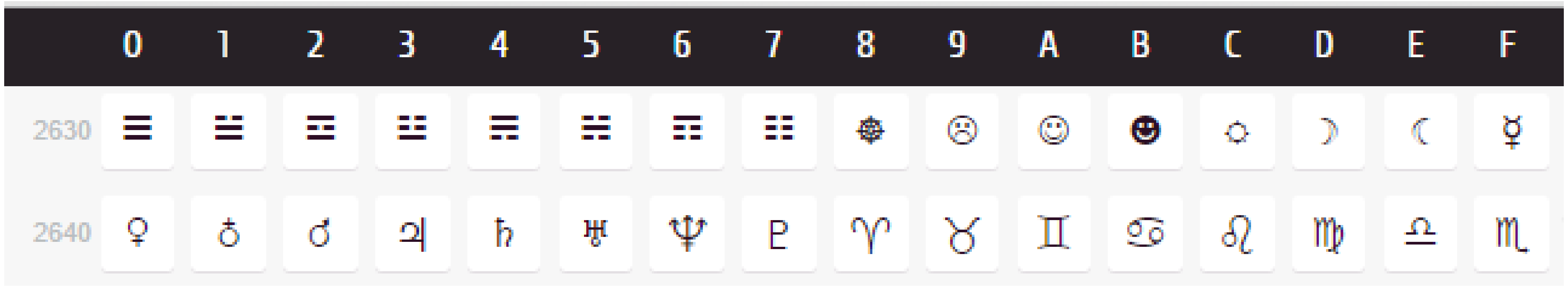
- To print character A, you can do one of the following:

```
System.out.print('A');// use this: easy to understand
System.out.print('\u0041');// equivalent but don’t use
```

- If you can type a character or symbol from your keyboard, do that; otherwise, use the Unicode hex

Unicode Table (cont.)

- The value of the smiley is `2630 + A = 263A`



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2630	☰	☱	☲	☳	☴	☵	☶	☷	☸	☹	☺	☻	☼	☽	☾	☿
2640	♀	♁	♂	♃	♄	♅	♆	♇	♈	♉	♊	♋	♌	♍	♎	♏

Unicode Table (cont.)

- To print the character as `char` or as `String`:

```
System.out.print('\u2263A'); // as char
```

  or

```
System.out.print("Hello, World \u263A"); // as String
```

- The `print` statement can accept `char` or `String`
- Part of the `String` may be escape sequence (as in previous examples) or Unicode hex

Convention & Errors

JC08 01.6

(A. Nguyen)

Convention

- Convention are “rules” used by programmers for other programmers
- Always write comments to **explain** what you do in Java code, for yourself & fellow programmers
- There are 3 ways to signal a comment:
    - Text within /** … */ is for documentation purpose understood by Javadoc
    - Text within /* … */ is for documentation purpose for a block of code, or to disable Java code
    - Text after // is for documentation purpose for one statement, or to disable one Java statement
- Indent for each block – BlueJ does this automatically

Compile-time/syntax errors

- Upper-cases and lower-cases are **not** interchangeable; e.g., `totalCount` and `totalcount` are two different variables
- Reserved words must be spelled as-is; e.g., `class` is not `Class`
- There must be a semi-colon at the end of each statement
- To see whether the syntax is correct or not, you can do it <u>**by yourself**</u> by writing the code in your IDE and run it. In each chapter or BlueJ project, have a program called **Verifier** for this purpose.

Run-time/logic errors

- If “Hello, Word” was accidentally typed (instead of “Hello, World”), the compile would be successful and program would run. However, the result would not be the programmer’s intention. This is called run-time or logic error.
- To avoid this kind of errors, the programs must be tested thoroughly, and different test cases considered. Even so, it is difficult to have 100% bug-free programs

Algorithms

JC08 01.7

(A. Nguyen)

About algorithms

- An algorithm is a sequence of steps with 3 properties: unambiguous, executable, terminating
- Pseudocode is a way to describe an algorithm in concise English (not in any programming language), understood by humans but not computers
- Pseudocode is the draft, to be used for coding later
- The equal sign (=) in pseudocode & programming languages means "assigned to" or "gets"; it does NOT mean "equals to" as in math
- A better alternative assignment symbol in pseudocode is the left arrow $\leftarrow$ (i.e., instead of the equal sign)

Example 1: Calculate amount of time to save $

Save $200 at 5% annual interest rate until doubled:

```
Let rate = .05
Let yearValue = 0 (i.e., starting year)
Let balance = 200 (i.e., starting balance)
Repeat the following until the balance is over 400 (i.e., target balance):
    Calculate interest amount: balance * rate
    Add the interest amount to the balance
```

Example 2: Find max

```
Find the highest score, among a list of scores:
  Let the first score be the highest score
  Repeat the following, from the 2nd score until the last score:
      If the score is higher than the highest score,
          Replace the highest score
```

THE END