

# Fundamentals of Programming

---

## Laboratory 2

**The basic components  
of computer programs,  
definitions and vocabulary**



# Important terminology

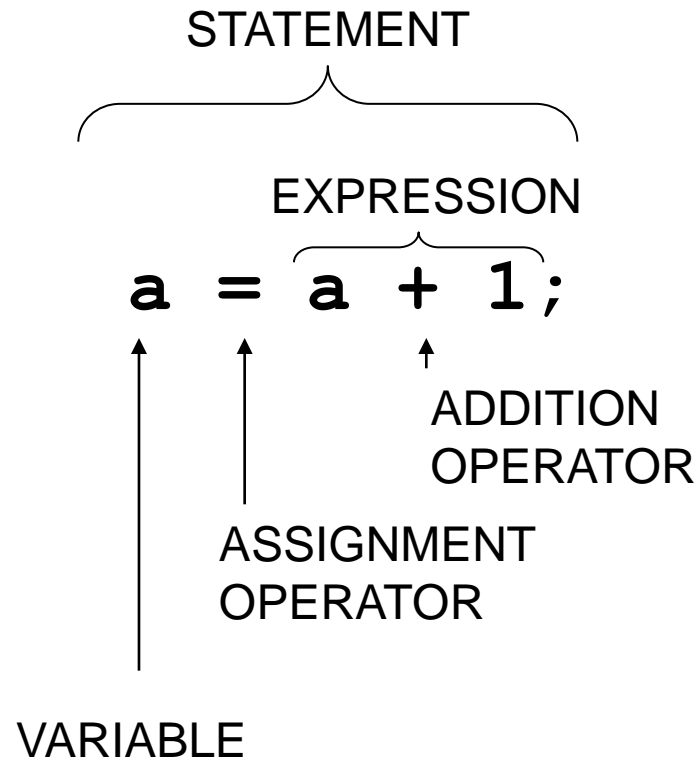
## STATEMENT

- smallest standalone piece of a programming language (usually one line of code)
- **semicolon terminated** (eg. C, Java)
- or **newline terminated** (eg. Basic, Javascript)
- can contain **keywords, variables, operators, methods, etc.**

*Example:*

**a=a+1 ;**

# Important terminology



# Important terminology

## EXPRESSION

- part of a statement that always returns (is “evaluated to”) a value
- usually cannot function as a statement itself (the returned value needs to „go somewhere”)

*Example:*

the “**a+1**” in the previous example (returns incremented a)

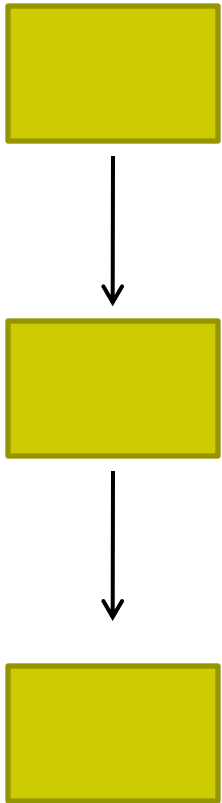
# OPERATORS

- Arithmetic
- Unary
- Assignment
- Relational and conditional
- Shift and logical
- Other

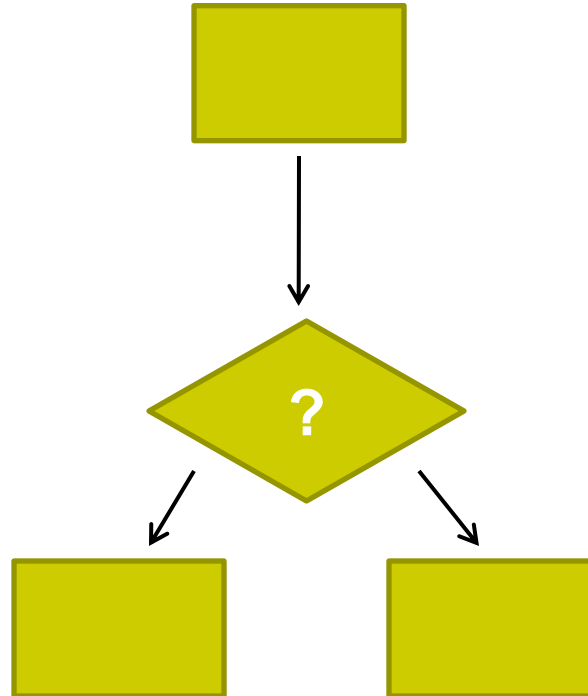
<i>Arithmetic</i> (used between two variables or expressions)		<i>Unary</i> (used on a single value as a prefix or suffix)	
+	Addition	-	Arithmetic negation
-	Subtraction (also unary minus)	!	Logical negation
*	Multiplication	++	Increment
/	Division	--	Decrement
%	Modulus (remainder from division)		

# Logic structures

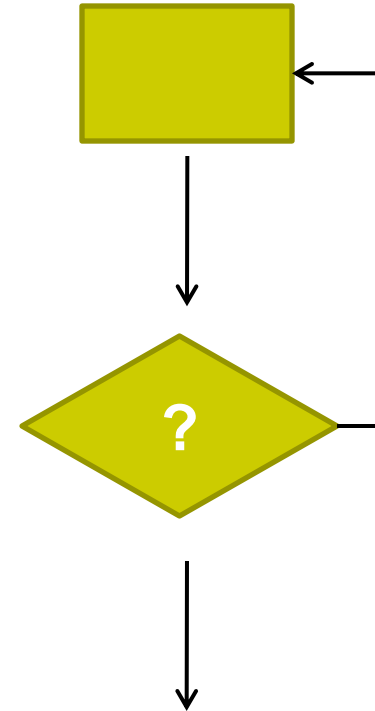
Sequence



Selection



Loop



# Important terminology

## BLOCK/COMPOUND STATEMENT

- statements can be grouped together by a **block** by curly braces

*Example:*

```
if (a==0)
{
    a++;
}
else
{
    b=a-1;
    a=a+1;
}
/*one of the two statement blocks
is executed depending on the conditional
branch statement "if" */
```

# Important terminology

## COMMENT

- information included in a program but not processed by a compiler
- makes code understandable and more readable for others (and the author)
- put at the end of a line after a “//”
- or inside /\* ... \*/ or /\*\* ... \*/

*Example:*

```
a = a+1; //increments "a" identifier  
/* a = a+1; this statement has been  
"commented out" */
```



# Important terminology

## VARIABLE

- a reservation of space in memory to store specific **type** of data
- can be referenced or modified by using the **variable's name**
- **variable's scope** – section of a program in which they can be referenced (usually within the bracketed block they were defined in)

# Important terminology

## VARIABLES

- must be **declared** (assigned a **name**)
- are of specific **data type**
- are often **initialized** (given a value from the start)

*Examples:*

```
float b;           //declaration (default 0)
                   b = 2.1f; //initialization
                               //(assigning a first value)
```

```
float a = 2.1f;
```

```
float c,d,e;
```

```
float f=1,g,h=2;
```



# BASIC JAVA LANGUAGE RULES

- Java is **CASE-SENSITIVE**  
`Main` is not the same as `main`
- Java **IGNORES WHITE SPACE** (spaces or tabs)  
`Main(args);` is the same as `Main ( args );`
- Java is **SEMICOLON TERMINATED**  
Each statement ends with a semicolon terminator ;
- The source file **must have the EXTENSION \*.java**  
`Hello.java`
- Every Java program **must have a method named `main( )`**



# BASIC JAVA LANGUAGE RULES

- The class containing the main method should be named the same as the program file and is referred to as the DRIVING CLASS.
- All braces { } must be matched.
- All VARIABLE NAMES must start with either a letter, an underscore \_ or a dollar sign \$.
  - ❑ They can contain numbers, but cannot contain non-alphanumeric characters aside from the underscore or \$.
  - ❑ They cannot be the same as reserved words, e.g. `class`, `if`

# Programming standards

- ⑩ CamelCase – compound words are capitalized, e.g.:
  - widthOfRectangle, endResultSum
- ⑩ All identifiers aside from classes start with lower case

# SMALL REVIEW

State whether the following variable names are valid or invalid. If they are invalid, state the reason.

Proda

\_c3

while

alb2c3d4

average

c1234

12345

\$total

9ab6

grade1

abcd

newbal

new bal

sum.of

finGrad



ORAL  
EXERCISE

# DATA TYPES

Groups	Variable Type	Storage	Approximate Range
Integrals	<code>int</code>	4 bytes	+/- 2 billion
	<code>short</code>	2 bytes	+/- 32,767
	<code>long</code>	8 bytes	+/- $9.2 \times 10^{18}$
	<code>byte</code>	1 byte	-128 to 127
Floating-point	<code>float</code>	4 bytes	7-8 digits
	<code>double</code>	8 bytes	16-17 digits
Character	<code>char</code>	2 bytes	65,536
Boolean	<code>boolean</code>	1 bit	<code>true</code> and <code>false</code>

# SMALL REVIEW

Write a declaration statement to declare that the variable **count** will be used to store an **integer**.





# SMALL REVIEW

Write a declaration statement to declare that the variable **grade** will be used to store a **floating-point** number. Initialize it to 100.



## SMALL REVIEW

Write a declaration statement to declare that the variable **yield** will be used to store a **double-precision** number.



# SMALL REVIEW

Write a declaration statement to declare that the variable **initial** will be used to store a **character**.



# SMALL REVIEW

Rewrite each of these declaration statements as three individual declarations:

```
int month, day = 30, year;
```

```
double hours, rate, otime = 15.62;
```

