Lecture 1

Primitive types, variables.

Working with console.

If-else statement

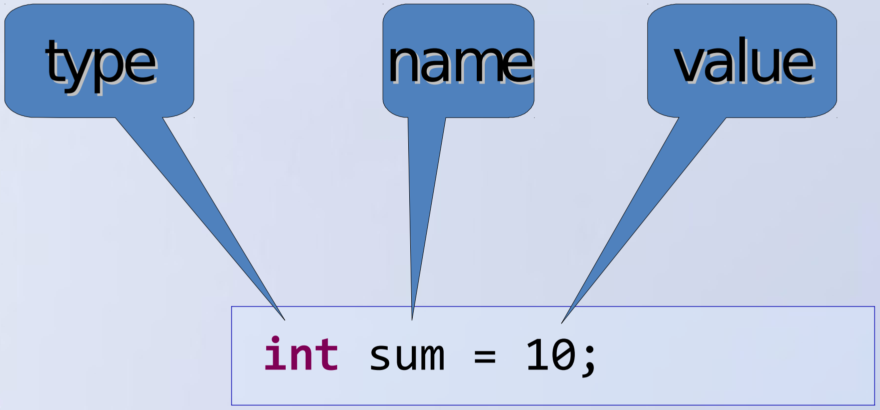
IT TALENTS
Training Camp

Contents

IT TALENTS
Training Camp

Variables

- Variables in java
  - It's purpose is to hold information
  - Has a unique name
  - Has a type
  - Has a value (can be changed)
- Declaring variable



```java
int sum = 10;
```

Primitive types in Java

- Primitives are basic java types
- Primitives can be used with basic operations
- Primitives' values can be assigned to variables

- Primitive types in java
  - byte, short, int, long
  - float, double
  - boolean
  - char

Numeric types

- Numeric types are **byte, short, long, int, double, float**
- **byte** – 8b (-128 : 127)

```
byte b = 100;
```

- **short** – 16b (-32768 : 32767)

```
short s = 10000;
```

- **int** – from integer, 32b

```
int i = 10000;
```

IT TALENTS Training Camp

Numeric types

- **long** – 64b

  long number = 100L;

  L is added as a sufix to indicate long type

- **float** - precision to 32b

  float f = 3.14f;

  f is added as a sufix to indicate float type

- **double** – precision to 64b

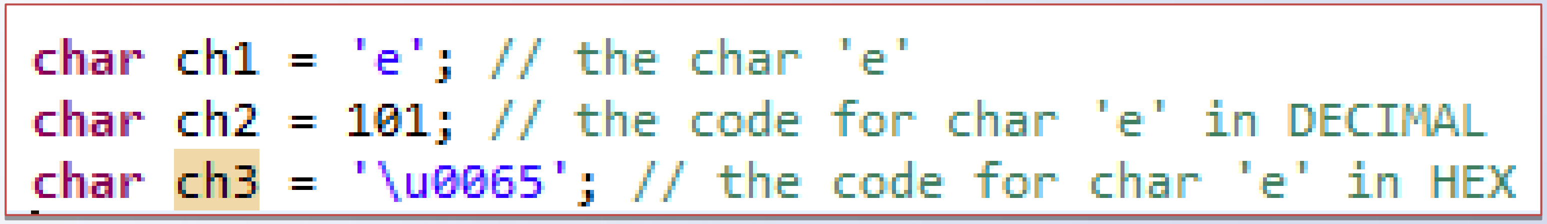
  double d= 3.14;

char and boolean

- **char** is used for 16b unicode character

  Char values are embedded in ''

  char ch = 'c';



```
char ch1 = 'e'; // the char 'e'
char ch2 = 101; // the code for char 'e' in DECIMAL
char ch3 = '\u0065'; // the code for char 'e' in HEX
```

- **boolean** has two values - true or false

  boolean bool = false;

Primitives' default values

Data type	Default value
• byte	0
• short	0
• int	0
• long	0
• float	0.0
• double	0.0
• char	'\u0000'
• boolean	false

Other data types

- Strings
- Reference types

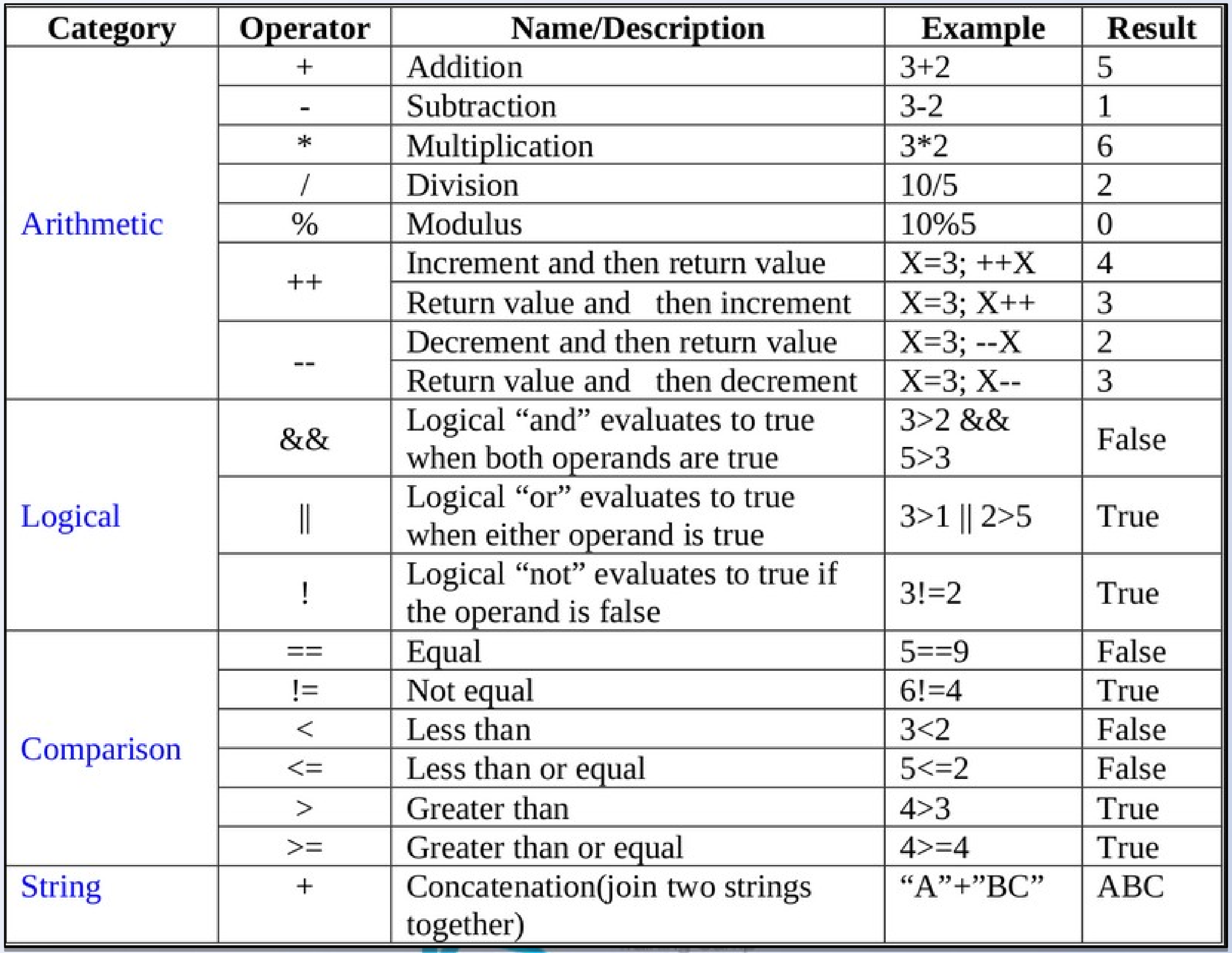
We'll talk about them later in the course!

Operators

- Java offers many operators for manipulating data.
  - Unary – takes one operand
  - Binary – takes two operands
  - Ternary – takes three operands
- Operands are the elements that the operator performs an operation on
  - Example: 2 + 3
    - + is the operator.
    - 2 and 3 are the operands

IT TALENTS
Training Camp

Operators



Category	Operator	Name/Description	Example	Result
Arithmetic	+	Addition	3+2	5
	-	Subtraction	3-2	1
	*	Multiplication	3*2	6
	/	Division	10/5	2
	%	Modulus	10%5	0
	++	Increment and then return value	X=3; ++X	4
		Return value and then increment	X=3; X++	3
	--	Decrement and then return value	X=3; --X	2
		Return value and then decrement	X=3; X--	3
Logical	&&	Logical "and" evaluates to true when both operands are true	3>2 && 5>3	False
	||	Logical "or" evaluates to true when either operand is true	3>1 || 2>5	True
	!	Logical "not" evaluates to true if the operand is false	3!=2	True
Comparison	==	Equal	5==9	False
	!=	Not equal	6!=4	True
	<	Less than	3<2	False
	<=	Less than or equal	5<=2	False
	>	Greater than	4>3	True
	>=	Greater than or equal	4>=4	True
String	+	Concatenation(join two strings together)	"A"+"BC"	ABC

- Modulus returns the remainder of the division of the left operand by the right operand.
    - Example: 7 % 5 results in 2
- The operands can be literals or variables.
- Operators have precedence just like in math
- Grouping with parentheses
    - Example: (-a + b) / c
        - A would be negated first
        - -a + b would happen next
        - The result of –a + b would then be devided by c

Numeral Systems

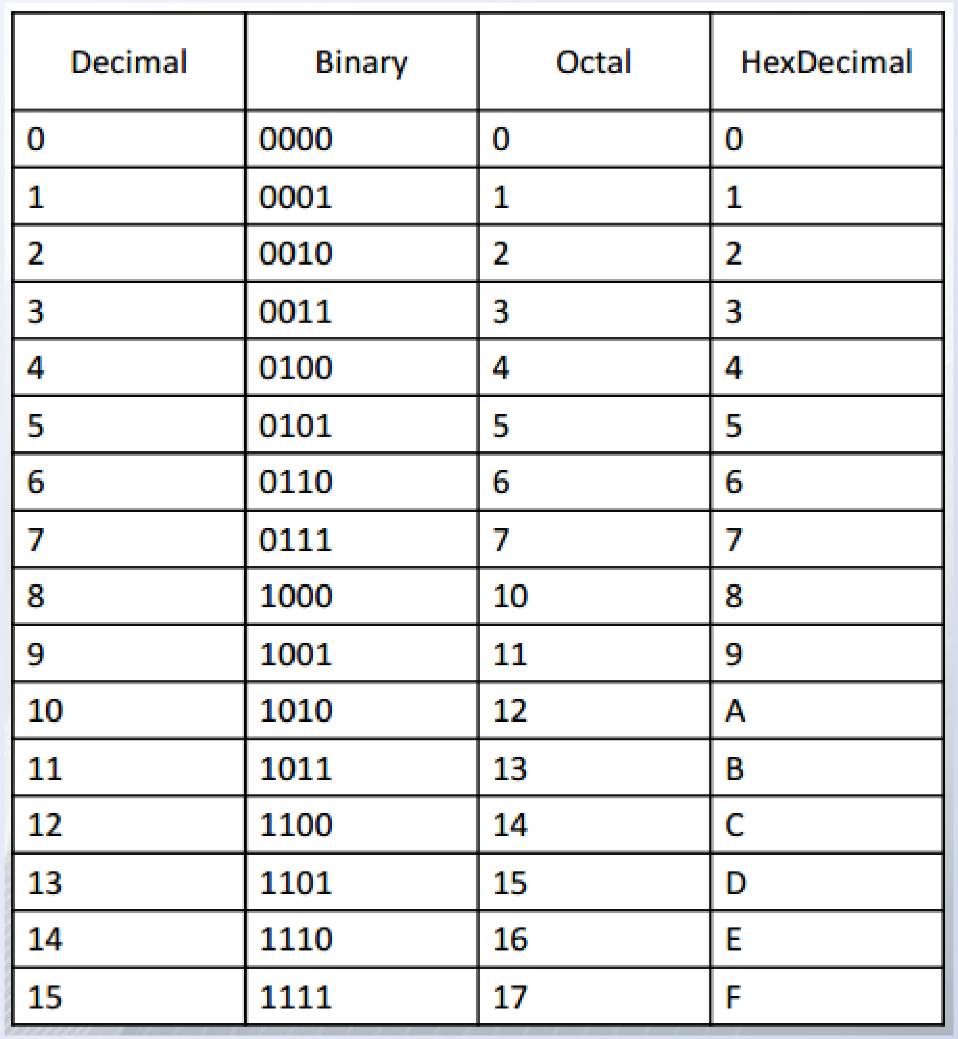
IT TALENTS
Training Camp

Definition

A numeral system is a writing system for expressing numbers, that is, a mathematical notation for representing numbers of a given set, using digits or other symbols in a consistent manner.

IT TALENTS
Training Camp

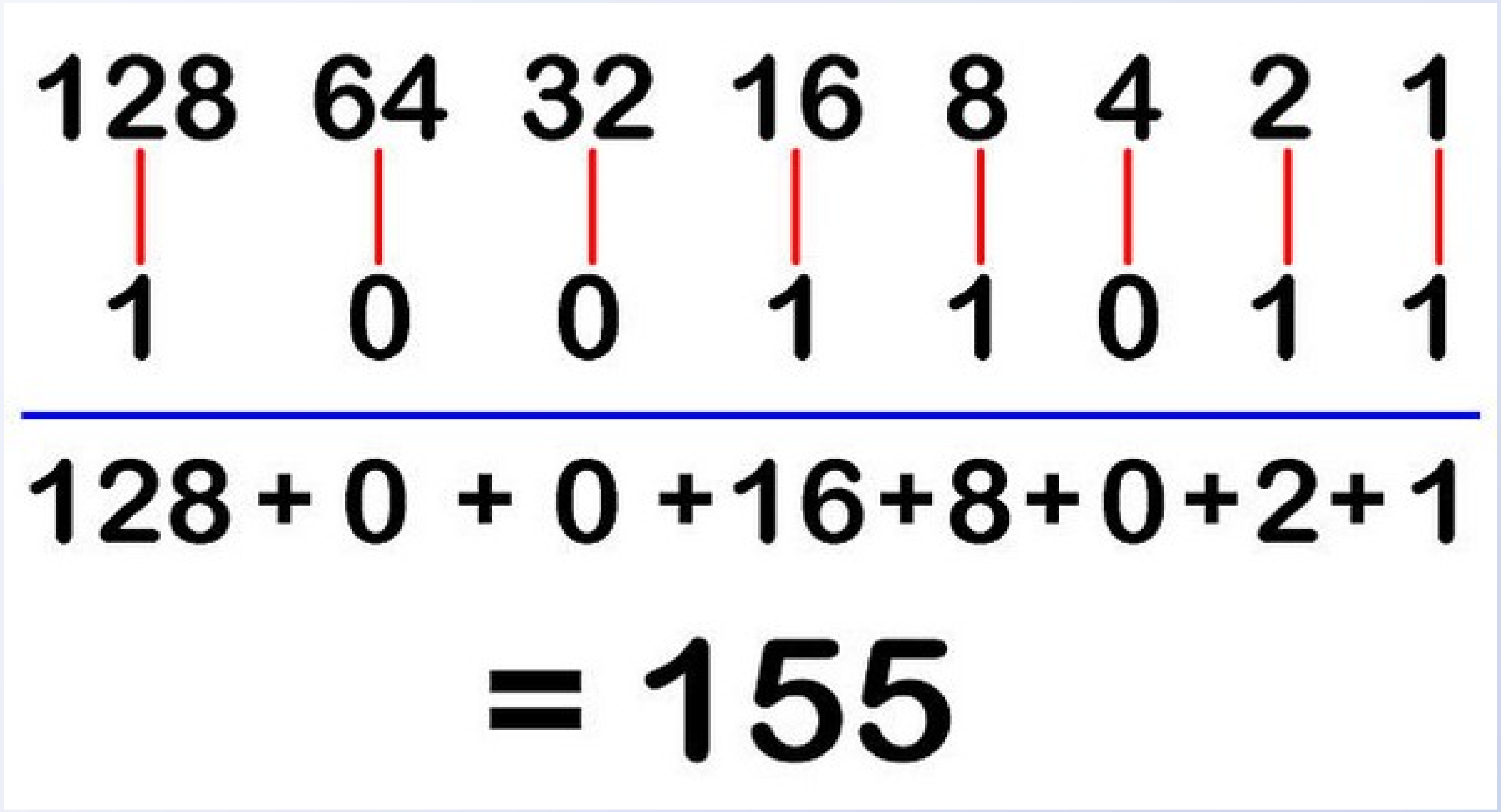
Different Numeral Systems



Decimal	Binary	Octal	HexDecimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

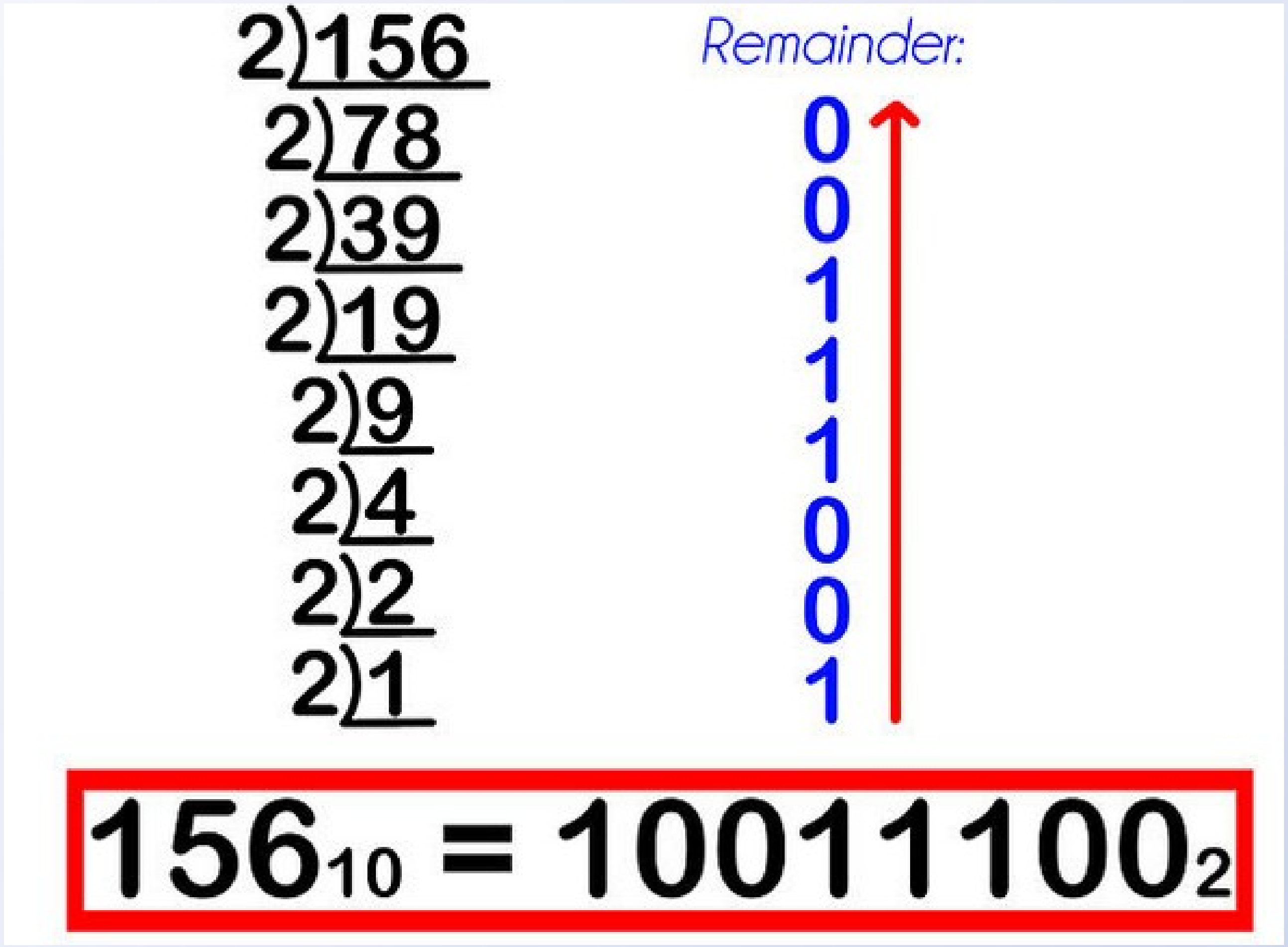
IT TALENTS
Training Camp

Converting From Binary to Decimal



IT TALENTS
Training Camp

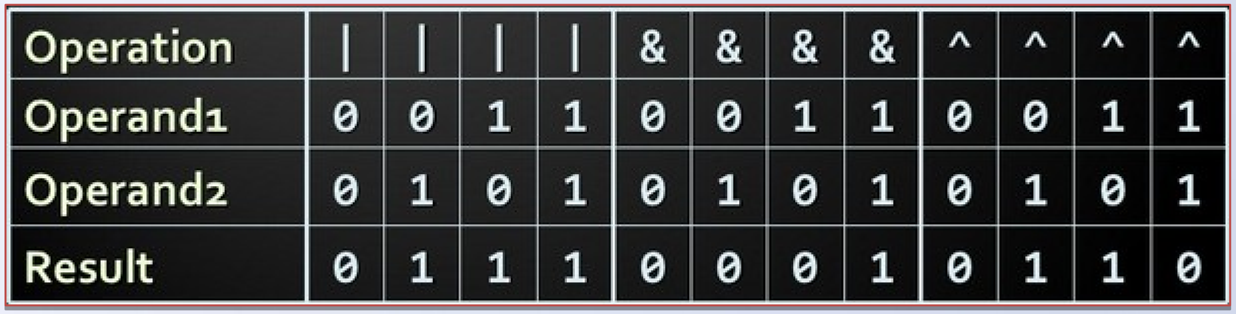
Converting From Decimal to Binary



$$156_{10} = 10011100_2$$

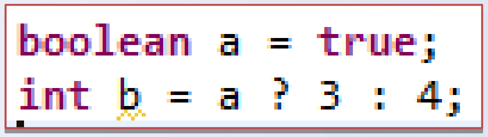
Other operators

- Bitwise operators
  - The |, & and ^ behave like ||, && and ^ for boolean expressions, but bit by bit
  - The << and >> move the bits (left or right)
-
-
-



Operation	|	|	|	|	&	&	&	&	^	^	^	^
Operand1	0	0	1	1	0	0	1	1	0	0	1	1
Operand2	0	1	0	1	0	1	0	1	0	1	0	1
Result	0	1	1	1	0	0	0	1	0	1	1	0

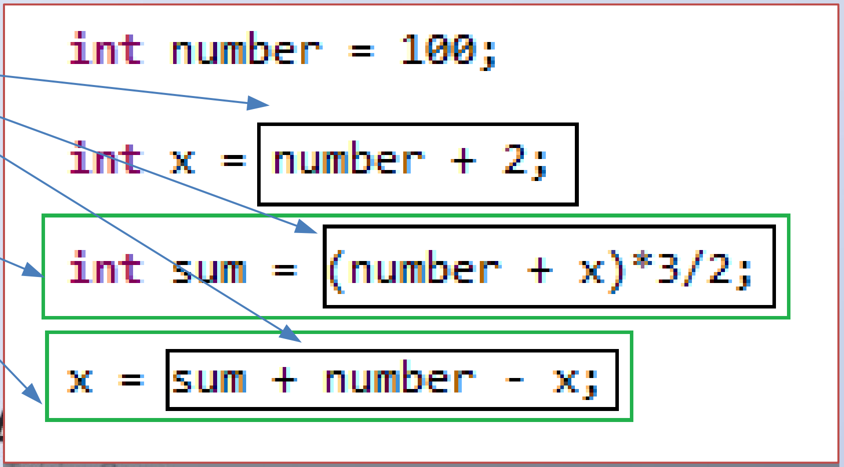
- Conditional operator ? :



```
boolean a = true;
int b = a ? 3 : 4;
```

Expressions and statements

- Expression is:
  - A construct, made up of variables, operators and method invocations, that evaluates to a single value.
- Statement is:
  - A complete unit of execution. Terminate with ;
- Example expressions:
- Example statements:



```
int number = 100;

int x = number + 2;

int sum = (number + x)*3/2;

x = sum + number - x;
```

Reading from console

Using Scanner

```java
Scanner sc = new Scanner(System.in);
```

Read user input with sc.nextXXX();

```java
sc.nextInt();
sc.nextDouble();
sc.nextLong();
```

Control flow

- Control flow is the way a program goes – execution of predifined statements
- Control flow may differ each time in dependance of conditions – either input data, or predifined conditions by the programer(i.e – time and so on)
- During the program execution decisions are being met – the program flow branches

Conditional Statement

- All logical operators

  NOT (!), AND (&&), OR (||)

- All comparison operators

  EQUAL (==), NOT EQUAL (!=)

  GREATER THAN (>), GREATHER OR EQUAL (>=)

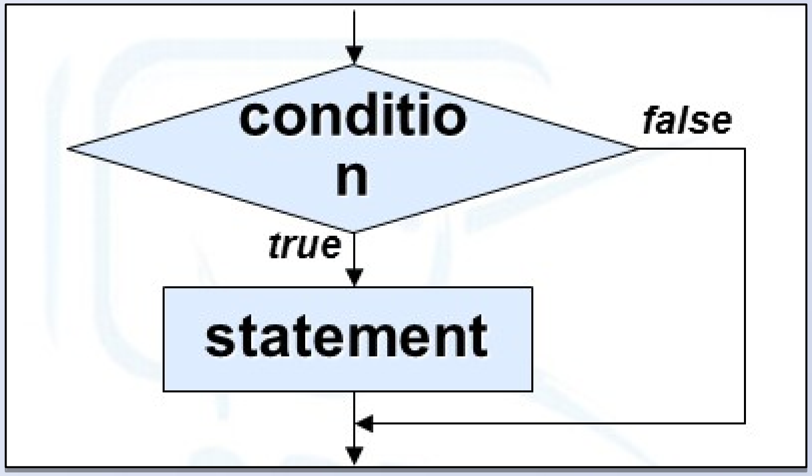
  LES THAN (<), LESS OR EQUAL (<=)

IT TALENTS
Training Camp

if-else statement

```
If (condition) {
        statement
}

if (condition) {
        executionA
} else {
        executionB
}
```



if-else statement

- If can exist without else
- But else can't exist without if
- Nested if-else statement

```
double a = 7.5;

if (a < 0) {
    System.out.println("a is smaller than 0");
} else {
    if (a == 0) {
        System.out.println("a is 0");
    } else {
        System.out.println("a is bigger than 0");
    }
}
```

Blocks

A block is a group of zero or more statements between balanced braces and can be used anywhere a single statement is allowed

```java
if (a > 10) {
    System.out.println("a is " + a);
    System.out.println("a is bigger than 10");
} else {
    System.out.println("a is not bigger than 10");
}
```

Always format your code! Do not write code like this:

```java
if (a > 10) {
System.out.println("a is " + a);
System.out.println("a is bigger than 10");}
else {System.out.println("a is not bigger than 10");
}
```

Mistake

```
int a = 7;

if (a > 10); {
    System.out.println("a is " + a);
    System.out.println("a is bigger than 10");
}
```

In this case println statements will be executed no matter the condition!

```
int a = 7;

if (a > 10);

{
    System.out.println("a is " + a);
    System.out.println("a is bigger than 10");
}
```

Summary

- Startup
- Variables
- Primitive types
- Operators
- Working with the console
- If-else statement and blocks