OL Academy

ITCS106,113

Computer Programming I

Chapter 2

Basic Computation

Variables and Expressions, Keyboard and Screen I/O, The class String

صفحة المقرر في موقعنا

itcs106.olearninga.com

2.1 Variables and Expressions

Syntax of declaration a class

```
public class FirstProgram
{
  public static void main (String [] args)
  {
   //write your code here ..
   // declare variables
  }
}
```

class body

Note that:

- A class is declared by use of **class** Keyword.
- Every line of code that runs in java must be inside a class.
- Java is case sensitive.
- The class body is enclosed between Curly braces { }.
- each code statement must be end with a semicolon (;).
- Single line comments start with two forward slashes (//).
- Multi-line comments start with /* and ends with */, any text between them will be ignored by java.

Variables:

- Variables in a program are used to store data such as numbers and letters.
- The number, letter, or other data item in a variables is called its value, and the value can be changed.
- Each variable is assigned one memory locations.
- Every variable in java must be declared before its used for the first time.
- You should choose variable names that are helpful.

Declaring (Creating) Variables:

- A variable declarations tells the computer what type of data the variable will hold.
- To create a variable, you must specify the type

Syntax of variable declarations

```
Type variable;

or

Type variable1, variable2, variable3;

Examples:
int num1,num2;
double price;
```

Data types in Java:

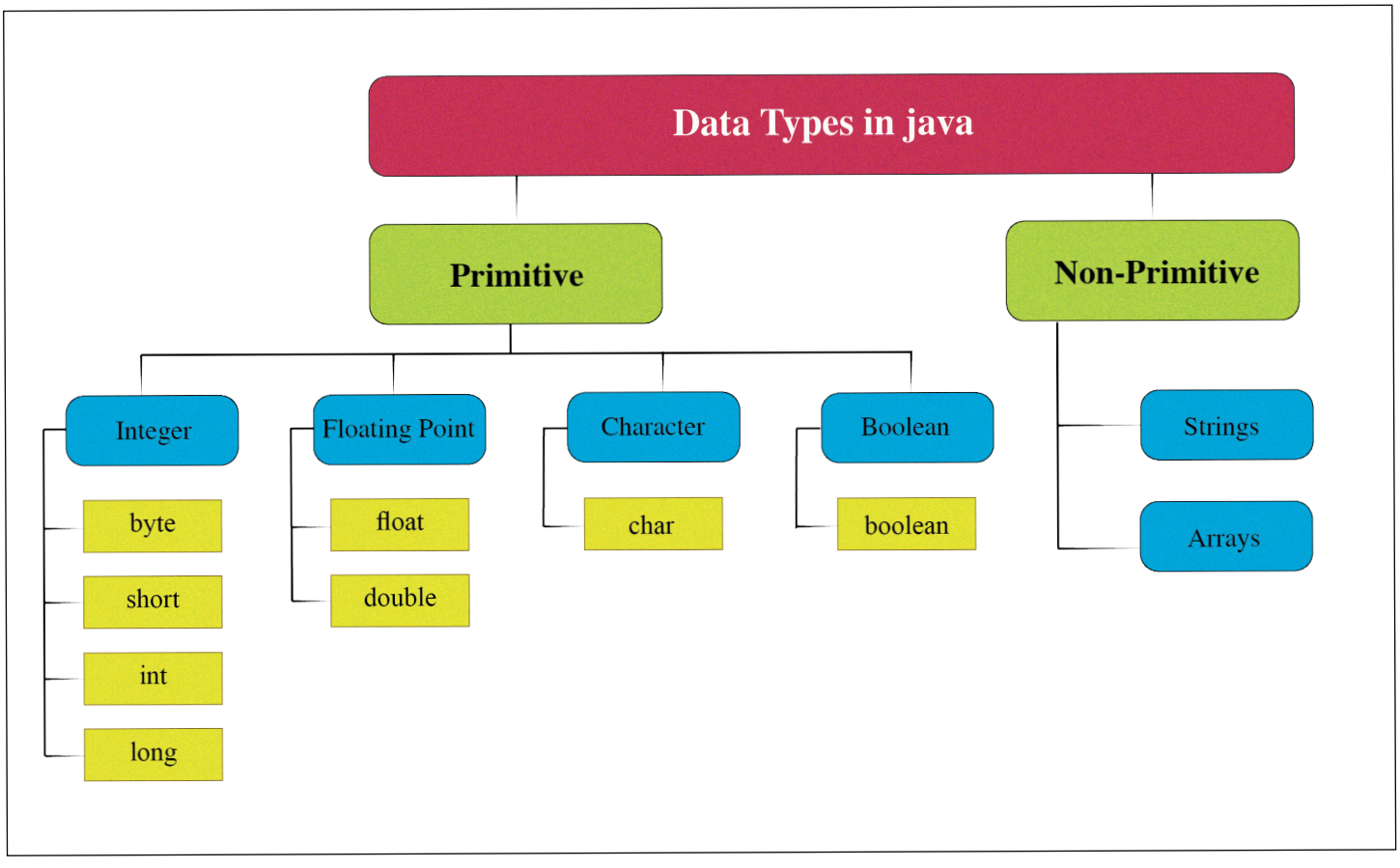
Data types are divided into two groups:

- **Primitive data types:**

  There are 8 primitive data types such as byte, short, int, long, float, double, char and boolean.

- **Non-primitive data types (object data types):**

  such as Strings and Arrays.



All of Java primitive types with amount of computer memory they use (size):

Type Name	Kind of value	Memory Used	Examples
byte	Integer	1 byte	Integers types 0, 127, -120
short	Integer	2 bytes	
int	Integer	4 bytes	
long	Integer	8 bytes	
float	Floating Point	4 bytes	Floating point types 0.99, 3.14159, 7.0, -25.8
double	Floating Point	8 bytes	
char	Single Character	2 bytes	'a', 'A', '@'
boolean		1 bit	true or false

Java Identifiers:

The name of something in Java program, such as a variable, class, or method is called an identifier.

The guidelines for choosing names:

- Names can contain letters, digits, underscores, but must begin with a letter or underscores.
- Names cannot contain whiteSpace and should start with a lowercase letter.
- Names are case sensitive (uppercase and lowercase letters are different - Java not same as java).
- dots(.), asterisks (*), hyphen (-) not allowed to use them.
- The symbol $ is also allowed, but it is reserved for special purpose, so you should not use it for variable name.
- cannot be used the reserved words (Keywords, such as class, int, boolean, true, import)

Legal identifiers:

firstName LastName _cost idNumber

Illegal identifiers:

google.com first-name 2Number Five* false

Assignment Statements (=)

is used to assign a value to a variable, for example:

Syntax

```
Variable = Expression;
```

Examples

```
age = 18;  // assignemnt statement
```

- The equal sign (=) is called the assignment operator.
- It means the variable on the left side will hold the value of the expression on the right side.
- The assignment statement ends with a semicolon (;).
- The expression can be another variable, a number, or a more complicated expression made up by using arithmetic operators.(+, -, *, /)

for example

```
totalPrice = price + price*0.05 ;
```

Syntax of Combining a Variable Declaration and an Assignment

```
 Type Variable = Expression;
```

or

```
Type Variable1 = Expression1, Variable1 = Expression1,...;
```

Examples

// combine the declaration and an assignment statement, Note that you can initialize some variables and not initialize others in a declaration.

```
int count = 0;
int balance = 1000, newBalance;
```

Simple Input and screen output:

You can use an object of the class Scanner to read input from the keyboard.

Example : Ask the user to enter his age from the keyboard, then display on the screen.

1. must import the scanner class at the beginning of your program file:

```
import java.util.Scanner;
```

2. You also need a statement in the following form before the first statement involving keyboard input, where ObjectName is any java identifier like **kbd**, **keyboard** or **input**.

```
Scanner ObjectName = new Scanner (System.in);
```

3. This line must be appear before the first statement that takes input from the keyboard:

```
int age = kbd.nextInt();
```

4. Display the age on the screen:

```
System.out.println(age);
```

Some methods in the class Scanner

Method for `Scanner kbd;`	Return Type	Description
`next()`	`String`	Returns the string value consisting of the next keyboard characters up to, but not including, the first delimiter character. The default delimiters are whitespace characters.
`nextLine()`	`String`	Reads the rest of the current keyboard input line and returns the characters read as a value of type `String`. Note that the line terminator `'\n'` is read and discarded; it is not included in the string returned
`nextInt()`	int	Returns the next keyboard input as a value of type `int`.
`nextDouble()`	double	Returns the next keyboard input as a value of type `double`.
`nextFloat()`	float	Returns the next keyboard input as a value of type `float`.
`nextLong()`	long	Returns the next keyboard input as a value of type `long`.
`nextByte()`	byte	Returns the next keyboard input as a value of type `byte`.
`nextShort()`	short	Returns the next keyboard input as a value of type `short`.
`nextBoolean()`	boolean	Returns the next keyboard input as a value of type `boolean`. The values of `true` and `false` are entered as the words true and false. Any combination of uppercase and lowercase letters is allowed in spelling true and false.

println VS print

System.out.prinln and System.out.print are almost same method. The **println** method advances to a new line after it displays its output, whereas the print method does not.

```
System.out.print("one");
System.out.print(" two");
System.out.println(" three");
System.out.print("four");
```

Output
one two three
four

Named Constants:

To define a name for a constant, write the keyword ***public static final*** in front of variable declaration that includes the constant as the initializing value. Place this declaration within the class definition but outside of any method definitions, including the main method.

```
public static final Type Variable = constant;
```

Syntax

```
public class FirstProgram
{
  // declare constatnts variables here
  public static final double PI = 3.14159;
  public static final char SCALE = 'K';


// main method
  public static void main (String [] args)
  {


  }
}
```

Type Casting

In many situations, you cannot store value of one type in a variable of another type unless you use a type cast that converts the value to an equivalent value of the target type.

Syntax

```
(Type_Name) Expression
```

for Example:

```
double distance = 20.5;

int point = distance;   //(This assignment is illegal) must be add the type casting

int point = (int) distance;  //now the value of point is 20, and distance 20.5
```

Arithmetic Operators:

They are used to perform simple arithmetic operations on primitive data types.

Assume int a = 5, b=2;

Operator	Operation	Example	Result
+	Addition	a+b	7
-	Subtraction	a-b	3
*	Multiplication	a*b	10
/	Division	a/b	2
%	Mod operator (Remainder)	a%b	1

Precedence Rules

Highest Precedence		
1	the unary operators	+, -, !, ++, --
2	the binary arithmetic operators	*, /, %
3	the binary arithmetic operators	+, -
Lowest Precedence		

Increment (++) Operators:

is used to increment a value by 1, We can apply (++) operator only for variables but not for the constant values.

There are two varieties of increment operator:

1. Pre-Increment (++a) : Value is incremented first and then the result is computed.

```
int a = 7;
System.out.println(++a);
System.out.println(a);
```

Output

8

8

2. Post-Increment (a++) : Value is first used for computing the result and then incremented.

```
int a = 7;
System.out.println(a++);
System.out.println(a);
```

Output

7

8

Decrement (--) Operators:

is used for decrementing the value by 1, We can apply (--) operator only for variables but not for the constant values.

There are two varieties of decrement operators:

1. Pre-decrement (--a) : Value is decremented first and then the result is computed.

	Output

`int a = 7;`
`System.out.println(--a);`
`System.out.println(a);`
	6
	6

2. Post-decrement (a--) : Value is first used for computing the result and then decremented.

	Output

`int a = 7;`
`System.out.println(a--);`
`System.out.println(a);`
	7
	6

2.2 The class String

A value of type String is a sequence of characters, treated as a single item.

```
String greeting = "Hello";
System.out.println(greeting);
```

Output

Hello

Concatenation of Strings (+)

You can join two strings together by connecting them with the + operator.

```
String greeting = "Hello";
System.out.println(greeting + "
World");
```

Output

Hello World

String Methods

(1) length ( ) : Returns the number of character of this string, the method length() returns an int.

```
String st1 = "ITCS 113";
System.out.println(st1.length());
```

Output

8

```
String st1 = "Welcome to Java";
int n= st1.length();
System.out.println(n+2);
```

Output

17

(2) charAt (index) : Returns the character at index in the String.

Note that: Index numbers begin at 0.

```
String st1 = "ITCS 113";
System.out.println(st1.charAt(5);
```

Output
1

(3) equals (a_String) : Returns true if two strings are equal, otherwise return false.

```
String st1 = "Java";
String st2 = "java";

System.out.print(st1.equals(st2));
```

Output
false

(4) equalsIgnoreCase (a_String) : Returns true if two strings are equal, considering upper and lower case of a letter ti be the same, otherwise return false.

```
String st1 = "Java";
String st2 = "java";

System.out.print(st1.equalsIgnoreCase(st2));
```

Output
true

(4) indexOf (a_String) : Returns the index of the first occurrence of the substring or -1 if not found.

Note that: Index numbers begin at 0.

```
String st1 = "ITCS 113";
System.out.println(st1.indexOf("C"));
System.out.println(st1.indexOf("5"));
System.out.println(st1.indexOf("1"));
```

Output
2
-1
5

(5) lastIndexOf (a_String) : Returns the index of the last occurrence of the substring or -1 if not found.

Note that: Index numbers begin at 0.

```
String st1 = "ITCS 113";
System.out.println(st1.lastIndexOf("1"));
```

Output
6

(7) toLowerCase () : Returns a new String having the same characters but convert any uppercase letters to lowercase.

(8) toUpperCase () : Returns a new String having the same characters but convert any lowercase letters to uppercase.

```
String st1 = "Java";
System.out.println(st1.toLowerCase());
System.out.println(st1.toUpperCase());
System.out.println(st1);
```

Output
java
JAVA
Java

(9) replace(oldchar, newchar) : Returns a new string having the same characters as this string, but with each occurrence of oldchar replaced by newchar.

```
String st1 = "Java";
System.out.println(st1.replace('a','A'));
```

Output
JAvA

(10) substring (start) : Returns a new string having the same characters as the substring that begins at index start through to the end of the string.

(10) substring (start, end) : Returns a new string having the same characters as the substring that begins at index start through to the end - 1.

Note that: Index numbers begin at 0.

```
String st1 = "Java";
System.out.println(st1.substring(2));
System.out.println(st1.substring(0,1));
System.out.println(st1.substring(0,3));
System.out.println(st1.substring(0,st1.length()));
```

Output
va
J
Jav
Java

(11) trim ( ) : Returns a new string having the same characters but with leading and trailing whitespace removed.

```
String st1 = "   Java    ";
System.out.println(st1);
System.out.println(st1.trim());
```

Output
Java
Java

(12) concat (a_String) : Returns a new string concatenated with a_string . You can use the + operator insted.

```
String st1 = "ITCS";
System.out.println(st1.concat("113"));
//same as when we use
System.out.println(st1 + "114");
```

Output

ITCS113

ITCS114

(13) compareTo (a_String) :

compareTo () method compares two strings lexicographically.

Each character of both the strings is converted into a Unicode value for comparison.

Dec	Char	Dec	Char
65	A	97	a
66	B	98	b
67	C	99	c
68	D	100	d
69	E	101	e
70	F	102	f
71	G	103	g
72	H	104	h
73	I	105	i
74	J	106	j
75	K	107	k
76	L	108	l
77	M	109	m
78	N	110	n
79	O	111	o
80	P	112	p
81	Q	113	q
82	R	114	r
83	S	115	s
84	T	116	t
85	U	117	u
86	V	118	v
87	W	119	w
88	X	120	x
89	Y	121	y
90	Z	122	z

It compares the strings character by character, starting from the beginning, The first difference determines the result.

The method returns:

Zero: if both the strings are equal.

Negative number: if the first string is lexicographically less than the second string.

Positive number: if the first string is lexicographically greater thean the second string.

```
String st1 = "A";
String st2 = "Z";
System.out.println(st1.compareTo(st1));
System.out.println(st1.compareTo(st2));
System.out.println(st2.compareTo(st1));
```

Output

```
0
-25
25
```

```
String st1 = "HELLO";
System.out.println(st1.compareTo("hello"));
```

Output

```
-32
```

This is because the letter **"H"** in "hello" comes before the letter **"h"** in "hello" in lexicographical order.

Remember:

- It compares strings based on their Unicode code points.
- It is case-sensitive, so uppercase letters are considered to be less than lowercase letters.

Some of Escape Characters using with Strings in java	
\"	double quote
\'	Single quote
\\	Backslash
\n	New line. Go to the beginning of the next line.
\t	Tab. Add whitespace

```
String st1 = "ITCS\t113";
System.out.println(st1);
System.out.println();
String st2 = "\"ITCS113\"";
System.out.println(st2);
System.out.println();
String st3 = "ITCS\n113";
System.out.println(st3);
```

Output

ITCS 113

"ITCS113"

ITCS
113

Questions 1: Find the output of the following codes

A-

```
int a=4;
int b=6;
int c;
a++;
--b;
c= a*b;

System.out.println(++a);
System.out.println(c);
System.out.println(b%2);
```

Output

6
25
1

B-

```
int a=7, b=5;
double c=5.3,d;
d= c-a/4*2;
System.out.println("d= " + d);
b *= a+32;
c = ++a * 3;
System.out.println("a= " + a);
System.out.println("b= " + b);
System.out.println("c= " + c);
System.out.println(b%4 + "\t" + b%10);
```

Output

d= 3.3
a= 8
b=195
c=24.0
3 5

C-

	Output

String line = "You have 75 minutes.";
System.out.println(line.length());
int n= line.indexOf("7");
System.out.println("n=" +n);
String msg = line.substring(4,n);
System.out.println(msg.toUpperCase());
msg = msg.concat("fun.");
System.out.println(msg);
	20
	n=9
	HAVE
	have fun.

D-

	Output
String s= ("if the plan doesn't work, change the plan, but never the goal"); System.out.println ("length = " + s.length()); System.out.println ("char = " + s.charAt(24)); System.out.println (s.lastIndex-Of("plan")); String s2 = s.substring(26,41); System.out.println (s2.toUpper-Case());	**length = 61** **char = ,** **37** **CHANGE THE PLAN**

Questions 2: Writing a Java program

(A) Write a program that ask the user to Enter 3 integer numbers. The program should then display on screen the following:

1. The summation of three numbers.
2. The multiplication of three numbers.
3. The average of three numbers.

Note that the average should be display with fractions, also should be don't declare more than 3 variables in your program.

SAMPLE INPUT/OUTPUT

```
Enter Number 1: 5
Enter Number 2: 3
Enter Number 3: 2

The Summation is: 10
The multiplication is: 30
The Average is: 3.3333333333333335
```

```
import java.util.Scanner;
public class ThreeNumbers {
    public static void main(String[] args) {
        Scanner kbd = new Scanner(System.in);
        int n1,n2,n3;
         System.out.print("Enter Number 1: ");
         n1 = kbd.nextInt();
         System.out.print("Enter Number 2: ");
         n2 = kbd.nextInt();
         System.out.print("Enter Number 3: ");
         n3 = kbd.nextInt();

         System.out.println();
         System.out.println("The Summation is: " + (n1+n2+n3));

      System.out.println("The multiplication is: " + n1*n2*n3);

      System.out.println("The Average is: " +
                          (double) (n1+n2+n3) / 3);

    }
}
```

(B) Write a program that computes the tax and tip on a restaurant bill. The program should ask the user to enter the charge for the meal. The tax should be 5% of the meal charge and the tip should be 7% of the total after adding the tax. Display the tax amount, tip amount, and total bill on the screen.

SAMPLE INPUT/OUTPUT

```
Enter meal charge: 17.320
The tax is 0.866
The tip is 1.2730
Total charge is 19.459
```

```
import java.util.Scanner;
public class RestaurantBill {
    public static void main(String[] args) {
        Scanner kbd = new Scanner (System.in);

        System.out.print("Enter meal charge: ");
        double charge = kbd.nextDouble();

        double tax = charge * 0.05;
        System.out.println("The tax is: " + tax);

        double tip = (charge+tax) *0.07;
        System.out.println("The tip is: " + tip);

        double totalCharge = charge+tax+tip;
        System.out.println("Total charge is: " + totalCharge);

    }
}
```

(C) Write a Java program that reads from the keyboard a distance in meters and the time taken as three numbers, hours, minutes and seconds. Your program should display the time in hours only, the distance into miles, and the speed in miles per hours.

Note that 1 mile = 1609 meters.

SAMPLE INPUT/OUTPUT

```
Enter distacne in meters: 2500
Enter time: 5    56    23

The time in hours is 5.939722
The distance in miles is 1.553760

Speed in mile/hour = 0.261588
```

```
import java.util.Scanner;
public class DistanceAndTime {
    public static void main(String[] args) {

        Scanner kbd = new Scanner(System.in);

        System.out.print("Enter distacne in meters: ");
        double distance = kbd.nextDouble();

        System.out.print("Enter time: ");
        int hours = kbd.nextInt();
        int minutes = kbd.nextInt();
        int seconds = kbd.nextInt();

        double totalHours = (hours + (minutes/60.0) +
(seconds/3600.0));
        System.out.println("The time in hours is: " + totalHours);

        double distanceInMiles = distance /1609;
        System.out.println("The distance in miles is: " +
distanceInMiles);

        double speed = distanceInMiles/totalHours;
        System.out.println("\n Speed in mile/hour: " + speed);
    }
}
```

(D) Write a program that prompts the user to enter airplane speed in (m/s) and the acceleration in (m/s^2). Your program should display the minimum runway length needed for the airplane to take-off using the formula:

length = speed² / 2 x acceleration

SAMPLE INPUT/OUTPUT

Enter speed and acceleration: 75 3.25 Min runway length is 865.38

```
import java.util.Scanner;
public class MinRunWay {
    public static void main(String[] args) {

        Scanner kbd = new Scanner(System.in);

        System.out.print("Enter speed and acceleration: ");
        double speed = kbd.nextDouble();
        double acceleration = kbd.nextDouble();



        double length = (speed*speed) / (2*acceleration);
        System.out.print("Min runway length is: " + length);


  }
}
```

(E) Write a Java program that asks the user to input two words of at least four letters. Your program then should do the following:

a. Convert all the letters of the first word to small letters and of the second word to all capital letters.

b. Combine the two converted strings into one string with a space between them, then print the result.

c. Print the first two letters of the first word followed by a space followed by the last two letters of the second word.

SAMPLE INPUT/OUTPUT

```
Enter the first word: hello
Enter the second word: World

hello WORLD
he LD
```

```
import java.util.Scanner;
public class TwoWords {
    public static void main(String[] args) {

        Scanner kbd = new Scanner(System.in);

        System.out.print("Enter the firt word: ");
        String first = kbd.next();
        System.out.print("Enter the second word: ");
        String second = kbd.next();

        first = first.toLowerCase();
        second = second.toUpperCase();

        System.out.println(first + " " + second);
        System.out.println(first.substring(0,2) +" "
+ second.substring(second.length()-2));
    }
}
```

Exercise (1)

Write a program that converts a distance measure from inches to cents. Note that one inch is equal to 2.54 cm?

SAMPLE INPUT/OUTPUT

```
Enter the distance in inches: 85
85.0 inches is equal to 215.9 cm
```

```
import java.util.Scanner;
public class InchesToCentimeters {

    public static void main(String[] args) {
        Scanner kbd = new Scanner(System.in);
        System.out.print("Enter the distance in inches: ");
        double inches = kbd.nextDouble();
        double centimeters = inches * 2.54;
        System.out.println(inches + " inches is equal to " +
centimeters + " cm" );
    }
```

Exercise (2)

The area of a triangle is computed by taking half the product of the base and height.

Area = 1/2 x base x height

Write a Java program that reads the base and height of a triangle and displays the area.

SAMPLE INPUT/OUTPUT

```
Enter the base of the triangle: 4
Enter the height of the triangle: 5
The area of the triangle is:10.0
```

```
import java.util.Scanner;

public class TriangleArea {

    public static void main(String[] args) {
        Scanner kbd = new Scanner(System.in);

        System.out.print("Enter the base of the triangle: ");
        double base = kbd.nextDouble();

        System.out.print("Enter the height of the triangle: ");
        double height = kbd.nextDouble();

        // Calculate the area using the formula
        double area = 0.5 * base * height;

        System.out.println("The area of the triangle
```