جامعة التنمية البشرية
UNIVERSITY OF HUMAN DEVELOPMENT

COMPUTER PROGRAMMING CONCEPTS

CS&IT 1101
Instructor: Shakar H. Salih
E-mail: shakar.salih@uhd.edu.iq

LECTURE 6

OUTLINES

- Java conventions
- Java Keywords
- Quadratic Equation
- Variable types in java

JAVA NAMING CONVENTIONS (1/3)

- Java **naming convention** is a rule to follow as you decide what to name your identifiers such as class, package, variable, constant, method etc.
- But, it is not forced to follow. So, it is known as convention not rule.
- All the classes, interfaces, packages, methods and fields of java programming language are given according to java naming convention.

JAVA NAMING CONVENTIONS(2/3)

Major naming convention in java:

- **package** in java should be in **small** letters as **java.util.**
- Each word of **interfaces** should start with **capital** letter as **Serializable**
- Each word of **classes** should start with **capital** letter as **String**
- **Method** name should be in **small** letters as **println()**.
- **Constants** should be in **all capital** letters like **MAX_VALUE**
- **Variable** name should be start with **small** letter and change as per word change.

JAVA NAMING CONVENTIONS(3/3)

Name	Convention
class name	should start with uppercase letter and be a noun e.g. String, Color, Button, System, Thread etc.
interface name	should start with uppercase letter and be an adjective e.g. Runnable, Remote, ActionListener etc.
method name	should start with lowercase letter and be a verb e.g. actionPerformed(), main(), print(), println() etc.
variable name	should start with lowercase letter e.g. firstName, orderNumber etc.
package name	should be in lowercase letter e.g. java, lang, sql, util etc.
constants name	should be in uppercase letter. e.g. RED, YELLOW, MAX_PRIORITY etc.

KEYWORDS IN JAVA

- In Java, a keyword is a word with a predefined meaning in Java programming language syntax. Reserved for Java, keywords may not be used as identifiers for naming variables, classes, methods or other identifier.
- **Examples:**
  - final
  - class
  - this
  - for

A COMPLETE LIST OF JAVA KEYWORDS

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

VARIABLE ASSIGNMENT (1/2)

- Value is assigned to a variable if that is already declared or initialized.

```
int a= 100;
int b;
b = 25; // direct assigned variable
b = a; // assigned value in term of variable
b = a+15; // assigned value as term of expression
```

VARIABLE ASSIGNMENT (2/2)

- Value is assigned to a variable if that is already declared or initialized.
- a + 2 = b + 10; **Wrong Statement**
- z = x + 4; **Ok Statement**
- a + 14 = m; **Wrong Statement**
- m = x + y; **Ok Statement**

QUADRATIC EQUATION

- In algebra
  - Quadratic = $ax^2+bx + c$
- In Java
  - Quadratic = a*x*x + b*x + c

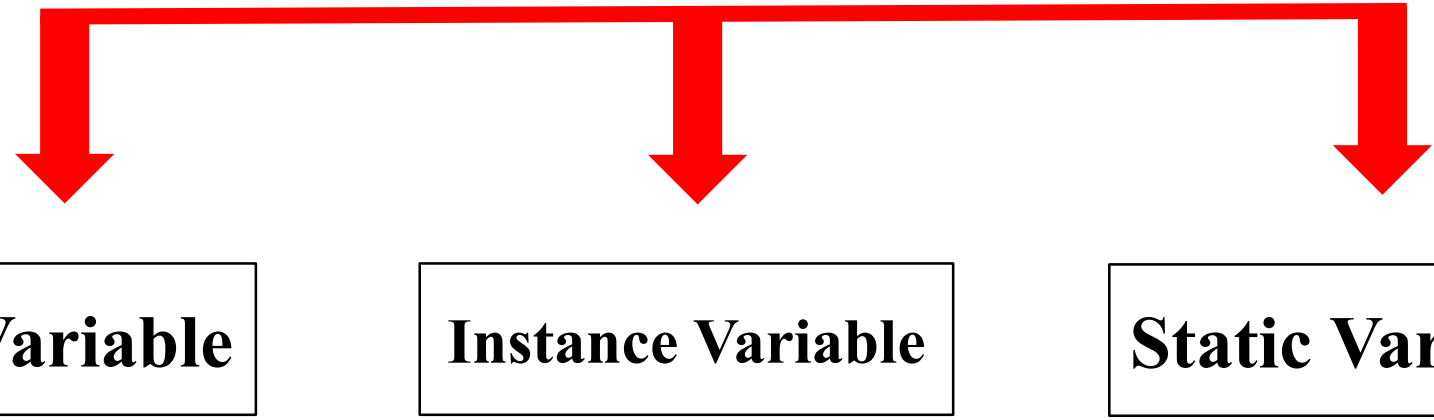
QUADRATIC FORMULA

- In algebra
  - $X=-b\pm\sqrt{b^2-4ac/2a}$
- In Java
  - x = (-b +- Math.sqrt((b^2)-4*a*c))/2*a;

```
Import java.lang.Math;
```

TYPES OF VARIABLE

Types of variable



LOCAL VARIABLE (1/2)

- A variable that is declared inside the method is called local variable.
- Access modifiers cannot be used for local variables.
- Local variables are visible only within the declared method, constructor, or block.
- Local variables are implemented at stack level internally.
- There is no default value for local variables, so local variables should be declared and an initial value should be assigned before the first use.

LOCAL VARIABLE (2/2)

- A variable that is declared inside the method is called local variable.

Example:

```
float getDiscount(int price)
{
    float  discount;  // here discount is a local variable
    discount=price &(20/100);
    return discount;
}
```

INSTANCE VARIABLE (1/2)

- A variable that is declared inside the class but outside the method is called instance variable . It is not declared as static.
- **Instance variable** in java is automatically initialized with default value
- **Instance variable** will be accessible any where within class.

INSTANCE VARIABLE (2/2)

- A variable that is declared inside the class but outside the method is called instance variable . It is not declared as static.

Example:

```
class Student
{
        String name; // here name is the instance variable
        int age;  // here age is the instance variable
}
```

STATIC VARIABLE/CLASS VARIABLE (1/2)

- A variable that is declared as static is called static variable. It cannot be local.
- Class variable has only one copy for each class, regardless of how many objects are created from it.
- Class variable is created when program starts and ends with the program.
- It has same default values as instance variable.
- Static can only be accessed by using it's fully qualified name as: ClassName.VariableName

STATIC VARIABLE/CLASS VARIABLE (2/2)

- A variable that is declared as static is called static variable. It cannot be local.

Example:

```
class Student
{
        String name;
        int age;
        static int CourseCode=1101; // here CourseCode is static variable
}
```

EXAMPLE TO UNDERSTAND THE TYPES OF VARIABLES

```
class Test
{
        int id; // instance variable
        static float salary; // static variable
    public static void main (String args [])
     {
        int a = 10; // local variable
     }
}
```

STATIC VARIABLE EXAMPLE

```
class A
{
    static int number=10; //static variable
}
class Test
{
  public static void main (String args [ ])
  {
    System.out.println(A.number); // static variable called
  }
}
```

LOCAL VARIABLE EXAMPLE

```
class Test
{
   public static void main (String args [ ])
   {
      int number =10; // local variable
      System.out.println(number);
   }
}
```

STATIC VARIABLE CAN NOT BE LOCAL

- In Java applications, static variables are always class level variables, they never be local variables.

```
class Test
{
  public static void main(String args[])
  {
    static int number=10; //Compiler Error
    System.out.println(number);
  }
}
```

THANK YOU.....

DO YOU HAVE ANY QUESTIONS ?