Handout 1

Introduction to Java programming language.

1. *Java* is a language for Object-Oriented Programming.

Object-Oriented Programming - is a powerful and popular design and programming technique. We will start by exploring the basics of *procedural* programming, then follow up with *object-oriented* programming.

2. Program development consists of
   a. analysis and design:
      - identify problem data, inputs, outputs
      - design the algorithm – set of steps that solves the problem, i.e. computes outputs from inputs.
      - top-down design - gradually elaborate the details.
   b. implementation of the algorithm in a computer language.
   c. testing

**Example:** Compute and display the gross and net amount due to an employee who has worked 12 hours at the hourly rate of $10.5. 5% tax must be withheld from the gross amount due to the employee.

Problem analysis and algorithm design.

**Data:**

**Inputs:**

**Outputs:**

**Algorithm:**
Implementation:

```java
/** Example for the first lecture. Computes gross and net wages based on the hourly rate, hours worked and tax. */

class WageCalculation {
    public static void main(String[] args) {
        // Declare variables to store problem data:
        int hoursWorked = 12;       // employee hours
        double hourlyRate = 10.5;   // pay rate per hour
        double taxRate = 0.05;      // tax rate for the amount withheld

        // Declare variables to store problem outputs
        double grossPay, netPay;

        // Compute grossPay and netPay
        grossPay = hoursWorked * hourlyRate;
        netPay = grossPay - taxRate * grossPay;

        System.out.println("The gross amount due is " + grossPay);
        System.out.println("The net amount due is " + netPay);
    }
}
```

3. Programming Language Concepts

Programs are written in a high level programming language. Computers only understand machine instructions that are sequences of 0 and 1’s.

- **Machine Language**
  - least natural language for humans, most natural language for hardware
  - just 0s and 1s
  - directly understood by hardware
  - not portable (hardware dependent)

- **High-Level Language**
  - closer to natural language
  - words, numbers, and math symbols
  - not directly understood by hardware
  - “portable” (hardware independent) source code
  - Java, C, C++, Python, BASIC, Lisp, Ada, etc.

**Getting from High-Level Language Source to Machine Code**

**Compiling a program** - translating from a high-level language source code to machine (object, or executable) code.

**Compiler** - a program that translates source code to machine (object, or executable) code.

**Interpreter** – a program that translates source code instructions into machine code one by one as it executes the program.
**JAVA Program Translation**

- Involves Both Compilation and Interpretation
- Java Compiler (in Sun’s Java Development Kit (JDK) program called `javac`) generates Intermediate Code called “Byte Code”
  Byte Code is the language of the so-called Java Virtual Machine.
  Byte Code is
    - low-level code – easily translatable into machine language
    - hardware independent
- An interpreter (program called `java`) translates from byte code to hardware-specific machine code

**Diagram:**

```
Java Program (hello.java)  Input for Java Program
                     ↓
Java Compiler (javac)    
                     ↓
Byte-Code Program (hello.class)  
                      ↓
Byte Code Interpreter (java)  
                      ↓
Machine-Language Instructions  
                      ↓
Computer Execution of Machine-Language Instructions  
                      ↓
Output of Java Program
```