Lec 3

Compilers, Debugging, Hello World, and Variables
Announcements

• First book reading due tonight at midnight
  – Complete 80% of all activities to get 100%
• HW1 due Saturday at midnight
• Lab hours posted
Programming Levels

- Machine code

- Assembly languages

- High level languages
You compile a non-java program on your desktop (it uses a Intel x86 processor) and then email the program to your friend to run on his system (it uses an AMD processor). Will it execute? Why?

A) Yes. The program is already down to machine code and can execute anywhere

B) Yes. Your friend’s system will automatically translate the program into a format it can execute

C) No. The machine code will be different on the 2 systems

D) No. Your friend’s system may not have the compiler/assembler installed.
How is Java different
Compiling
You have the file Hello.class how do you make it run on your computer?

A) javac Hello.class
   java Hello
B) javac Hello.java
   java Hello
C) java Hello
D) javac Hello
E) Can’t run without the .java file
Your first program
Your first program

/* Your first program */
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}

Your first program

/* Your first program */
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}

Comments

• Exist to help the programmer
  – Today or 2 years in the future
  – Should add value to the reader
- Single line //
- Block comments /* */
Which is not an example of a helpful comment

A) `i = 0; //set i to 0`
B) `//TODO: implement feature to increase BMI accuracy`
C) `/* This program takes two ints as input from the user and calculates their BMI */`
D) `/* @return double - the calculated BMI */`
Your first program

/* Your first program */
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}

Your first program

/* Your first program */
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}

Your first program

/* Your first program */
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}

Public class Hello {
    public .. main() {
    }
}
Your first program

/* Your first program */
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}

Question

What happens if there is no main method, e.g. you have the code:

```java
public class Hello {
    public static void notMain(String[] args) {
        System.out.println("Hello world!");
    }
}
```

A)  Compiler will report an error  
B)  Program will compile and print “Hello World”  
C)  Program compiles but nothing happens  
D)  Program compiles, but once executed reports an error similar to “no main method”
Your first program

/* Your first program */
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}

Summary: Program must haves

• Filename and class name match
• { and } start and stop sections
• Instructions end with a ;
• public static void main()
  – For now: always put in your code. This where execution starts
Style

• Indentation in Java is there for user readability

```java
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
```

• Style requirements for this class
  – Each time you use a '{' indent farther. Go back when closing with '{'}
  – 2 spaces per indent
  – All lines less than 80 characters
  – Use comments to clarify logical parts of program
  – Use blank spaces around logical chunks of code
int x;
int x;
x = 200;
int x;
x = 200;
int y = 100;
Variables

• `int` is a “primitive” types:
  – `int` – standard integers, e.g. 5, 100, -37
  – `double` – decimal numbers, e.g. 5.32, -100.2
  – `char` – single character, e.g. ‘C’, ‘!’
  – `boolean` – true/false
    • boolean flag = true;
= VS. ==

• =
  – Does the assignment of a value, if you do:
    ```java
    int j = 20; //you are giving a value of 20 to j
    int b = j;  // also gives the value of 20 to b
    ```

• ==
  – Used to do comparisons, outcome is either true or false. (Think of as ‘are these equal’)
    ```java
    int a=10; int b=20; boolean b = (a == b);
    ```
• What is the output of this code?

```java
int x = 3;
int y = x;
System.out.println(y == 3);
int x = 4;
System.out.println(y == 3);
```

A) true  B) false  C) true  D) false  E) None of the above