Programming Errors

- Syntax Errors
  - Detected by the compiler
- Runtime Errors
  - Causes the program to abort
- Logic Errors
  - Produces incorrect result

Syntax Errors

```
public class ShowSyntaxErrors {
  public static void main(String[] args) {
    i = 30;
    System.out.println(i + 4);
  }
}
```

Runtime Errors

```
public class ShowRuntimeErrors {
  public static void main(String[] args) {
    int i = 1 / 0;
  }
}
```

Logic Errors

```
import java.util.Scanner;

public class ShowLogicErrors {
  // Determine if a number is between 1 and 100 inclusively
  public static void main(String[] args) {
    // Prompt the user to enter a number
    int numre;
    Scanner input = new Scanner (System.in);
    System.out.print("Please enter an integer:");
    number = input.nextInt();

    // Display the result
    if ( (1 < number) || (number < 100))
       System.out.println("The number is between 1 and 100");
 }
}
```

Precedence of Arithmetic Operators

1. `()` Evaluated first. If the parentheses are nested, the expression in the innermost pair is evaluated first. If there are several pairs of parentheses on the same level (i.e., not nested), they are evaluated from left to right.
2. `* / %` Evaluated second. If there are several such operations, they are evaluated from left to right.
3. `+ -` Evaluated third. If there are several such operations, they are evaluated from left to right.
4. `=` Evaluated last.

Example

X=**3+5**+9−3;

X=**8+9**−3;

X=**17−3;**

X=14;

X=3+**5*3**−8/2;

X=3+15−**8/2;**

X=**3+15**−4;

X=18−4;

X=14;

X=3+5*(3−8)/2;

X=?

X=2+8/(1+3)*2*(2+4)−12%6;

X=?

X=5−5*3/(1+(8−6))−8+3%2*6/(2−4*2)*2−1;

X=?

Exercise

Mathematical Formula	Java Expression
$z = b^2 + 4ac$	?
$p = x(y + z)$	
$y = \dfrac{1}{x^2 + x + 3}$	
$x = \dfrac{a + b}{c - d}$	

Example

```
a = 5;
x= 3 + ++a +4;
x = ?
a = ?
```

```
a = 5;
x= 3 - a++ +4;
x = ?
a = ?
```

```
a = 7;
x= 10 - --a +3;
x = ?
a = ?
```

```
a = 7;
x= 7 + a-- -3;
x = ?
a = ?
```

A Possible Bug

Never use combination of `a++`, `++a`, `a--`, `--a`, and `a` in a single statement.

a = 5;

x = 5 + ++a + 2+ a + a++; ← Possible bug

The result may be different in different compilers.

Exercise

```
y=3;
y*=5+2;       y = ?

y=6, x=5;
y-=x++ +3;   y = ?

y=6, x=5;
y%=--x +3;   y = ?
```

Primitive Types of Data

The eight primitive data types are called primitive types because they are simple and uncomplicated

- boolean
- float
- byte
- int

- char
- long
- double
- short

Variable Declaration

Variable names should stand for what they are.

for example:

```
double x, y;
```

Instead use

```
double radius, interestRate;
```

By convention, variable names are in lowercase.

If a name consists of several words, concatenate all of them and capitalize the first letter of each word except the first one.

Working with the char Data Type

- char data type is used to hold a single character
- Uses single quotation marks

Example:

```
char x = 'a';
 x =  '%';
 x =  '9';
```

String Data Structures

- A string can contain a string of characters
- Uses double quotation marks

Example:

```
x = "a";
x = " This is a test ";
```

HINT: "a " != 'a'

Implicit Type Casting

Example:

```
float x = 5.9;
short y = 10;
int z = 12;

? =  z * y + x;
```

1. double
2. float
3. long
4. int
5. short
6. byte

Explicit Type Casting

The unifying type can be overridden by explicitly stating a type cast

Place the desired type result in parentheses followed by the variable or constant to be cast

Example:

```
float x = 123.12;
int  y = (int)x /4;
```

Lab Preparation

- Read Chapter 2
- You will work on Lab 1 and Assignment 1