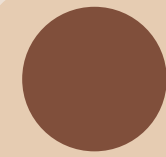


Topic 1.1: Primitive Operators

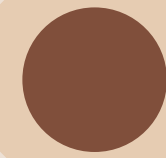
Learning Goals (Week 1):



Identify data types based on value



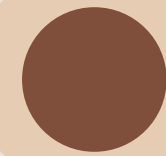
Map variables to the current values



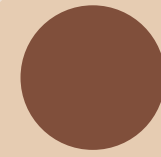
Perform basic operations on variables



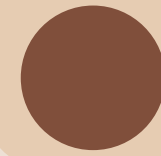
Create and use Java and user-defined methods



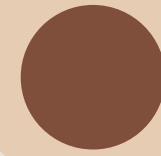
Format Printed Output



Obtain and process user input from the console



Use booleans, conditionals, and compound conditionals correctly



Select and implement different types of loops depending on scenario



Use special String and Math operations



Successfully implement and manipulate java arrays

Primitive Operators:

+ (addition) (String, Integer, & Floating-point Values)

- / * (subtraction, division, multiplication) (Integer & Floating-point Values)

% (modulo) (Integer & ~~Floating-point Values~~)

++ (increment) (Integer & Floating-point Values)

-- (decrement) (Integer & Floating-point Values)

Primitive Operators: +

- binary operator (needs two operands)
- can accept any two of (char, byte, short, double, int, long) OR Strings
- some combinations work, some don't
- String + anything (including boolean) = works like concatenation
- any of the above (not including boolean) + any of the above (not including boolean) = ?
- Remember: double > float > long > int > short > byte
- Using + on two types will result in the value stored in the larger of the two types (String is the largest)

`"Hello" + 1 = Hello1`

`"Hello" + false = Hellofalse`

`1 + 1 = 2`

`1 + 2 = 3`

`1 + "Hello" + false = 1Hellofalse`

`1 + 's' = 'b' // 1 more char over is 'b'`

`'a' + 'b' = 195 // (char)195 is Ã this is an ascii thing`

Pause & Practice:

- What will each of the following outputs be when run inside a `System.out.println` statement? If an error, say “ERROR”
- First, make a guess, then try it on your own (no solutions given)
- If you are wrong, make sure you understand WHY
 - Ask chatGPT “Why does `a+b=y` instead of `z`?” (where `a+b` is what I give you, `y` is the real answer, and `z` is your guess)

`'z' + 1` = ?

`“hello” + “z”` = ?

`“a” + “b”` = ?

`1.0 + 5` = ?

`5.3 + 1.1` = ?

`1.0 + 'B'` = ?

`false + 1` = ?

`'b' + false` = ?

Primitive Operators:

- / * (subtraction, division, multiplication) (Integer & Floating-point Values)

% (modulo) (Integer & ~~Floating-point Values~~)

- Work just as you'd expect
- Remember that rounding errors may exist when dealing with floating-point values (floats and doubles)
 - Values seems particularly obscure when using %, which is why I crossed it out
- There is no exponent/power operator like Python; you need to use a special method (see 1.8)

Primitive Operators:

`++` (increment) (Integer & Floating-point Values)

`--` (decrement) (Integer & Floating-point Values)

- increment or decrement by 1

```
int a = 1;
```

```
a++; // a now equals 2
```

```
double b = 1.5;
```

```
b--; // b now equals 0.5
```

Primitive Operators:

`++` (increment) (Integer & Floating-point Values)

`--` (decrement) (Integer & Floating-point Values)

- increment or decrement by 1

```
int a = 1;
```

```
a++; // a now equals 2
```

```
double b = 1.5;
```

```
b--; // b now equals 0.5
```

`a++;` is equivalent to `a += 1` or `a = a + 1`

`b--;` is equivalent to `b -= 1` or `b = b - 1`

This means we can also do things like:

```
c += 5;    // adds 5 to c
```

```
c *= 100;  // multiplies 100 to c
```

```
c /= 2;    // divides c by 2
```