I WON'T LET JAVA CONFUSE YOU!

NATALIIA DZIUBENKO

NATALIIA DZIUBENKO

SOFTWARE ENGINEER @XEBIA

@WORTH_EXPLORING

INCREMENT

INCREMENT

00 : 15

```
int a = 0;
a = a++;
```

INCREMENT

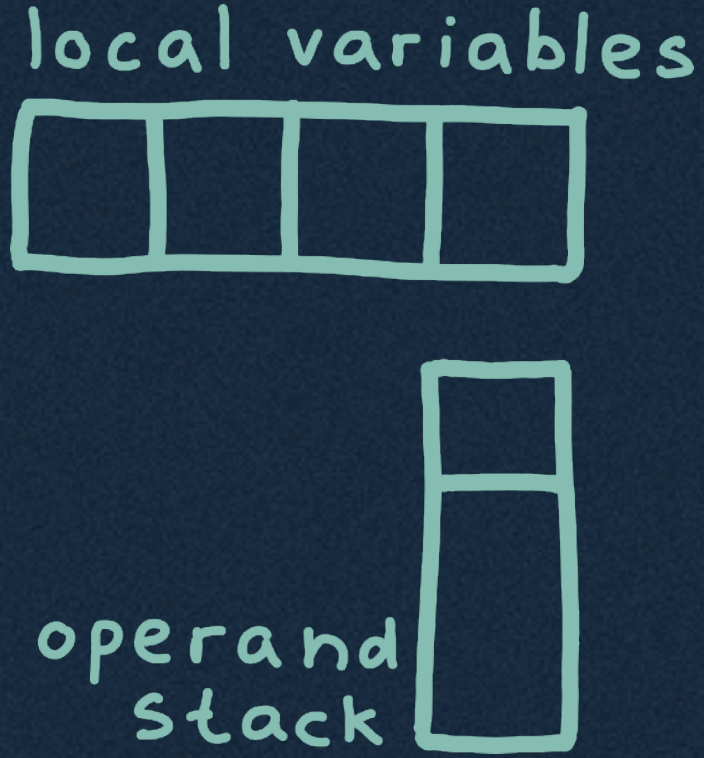
```
int a = 0;
a = a++;
```

```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: istore_0
```

INCREMENT

```
int a = 0;
a = a++;
```

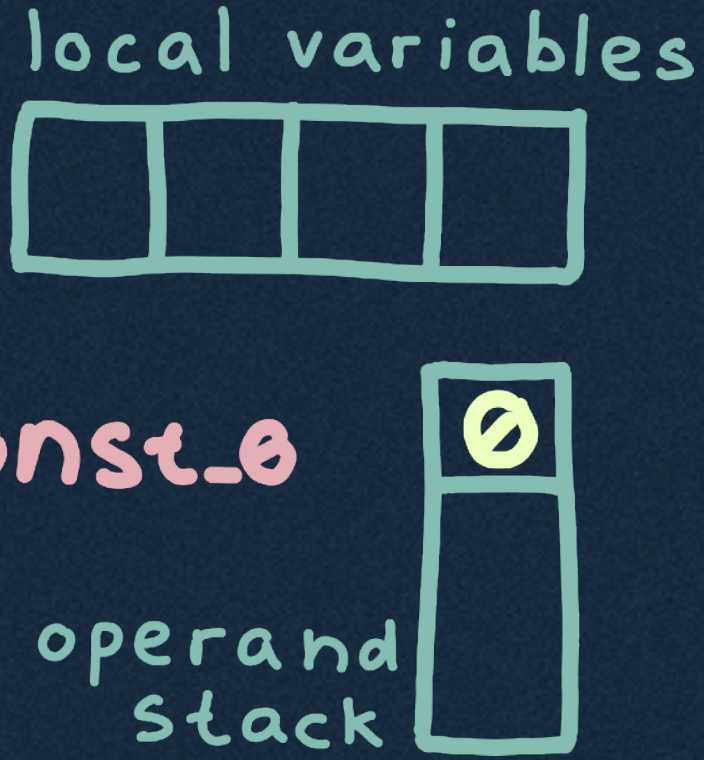
```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: istore_0
```



increment

```
int a = 0;
a = a++;
```

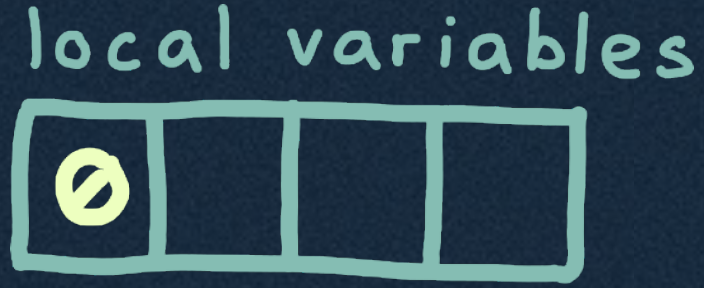
```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: istore_0
```



increment

```
int a = 0;
a = a++;
```

```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: istore_0
```

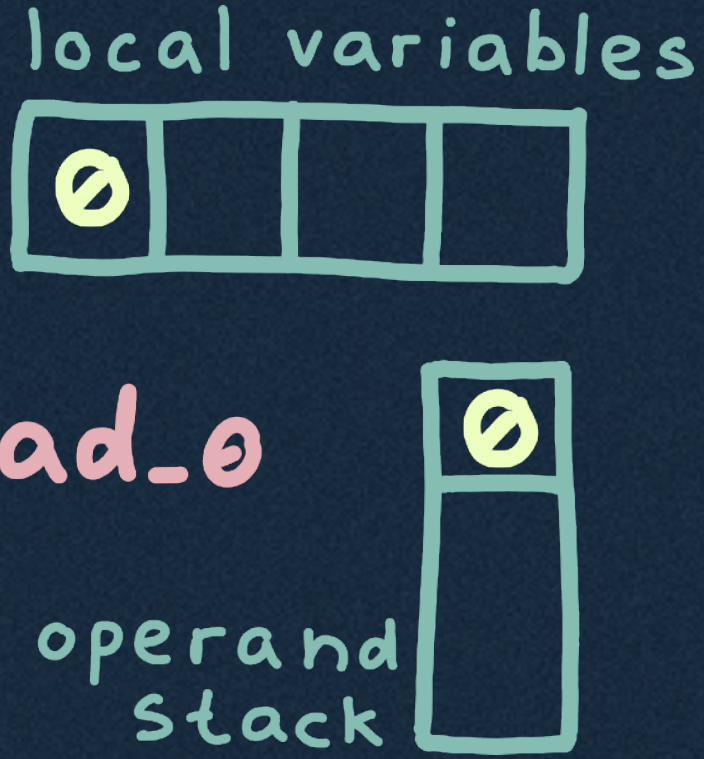


istore_0

increment

```
int a = 0;
a = a++;
```

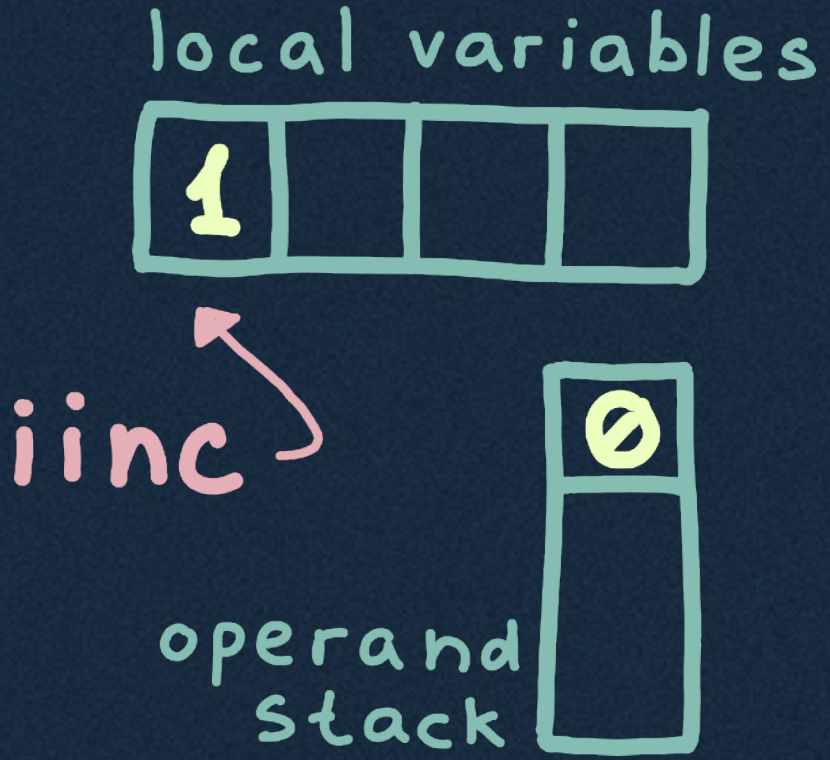
```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: istore_0
```



increment

```
int a = 0;
a = a++;
```

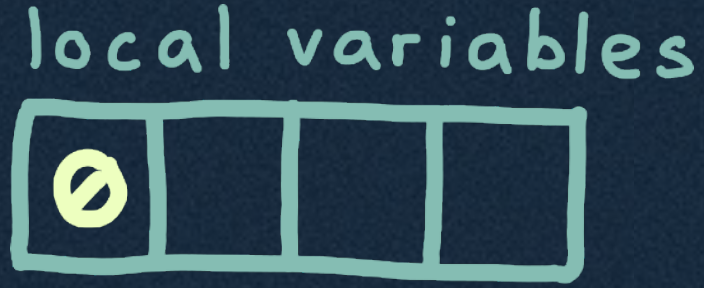
```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: istore_0
```



increment

```
int a = 0;
a = a++;
```

```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: istore_0
```



istore_0

INCREMENT


```
int a = 0;
a = a++ + ++a;
```

INCREMENT

```
int a = 0;
a = a++ + ++a;
```

```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: iinc      0, 1
9: iload_0
10: iadd
11: istore_0
```

INCREMENT

```
int a = 0;
a = a++ + ++a;
```

```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: iinc      0, 1
9: iload_0
10: iadd
11: istore_0
```

INCREMENT

```
int a = 0;
a = a++ + ++a;
```

```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: iinc      0, 1
9: iload_0
10: iadd
11: istore_0
```

istore_0

increment

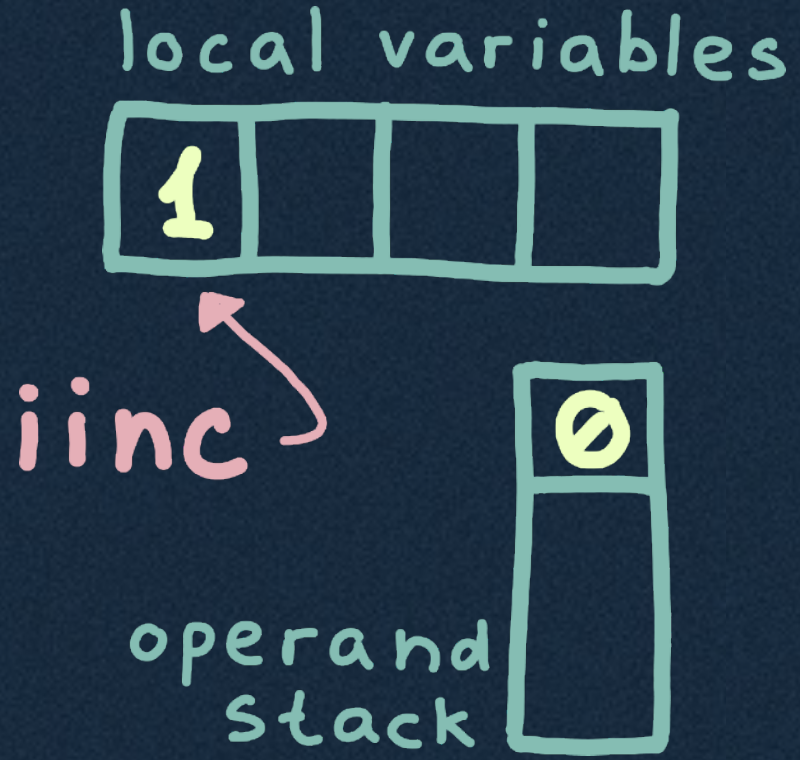
```
int a = 0;
a = a++ + ++a;
```

```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: iinc      0, 1
9: iload_0
10: iadd
11: istore_0
```

increment

```
int a = 0;
a = a++ + ++a;
```

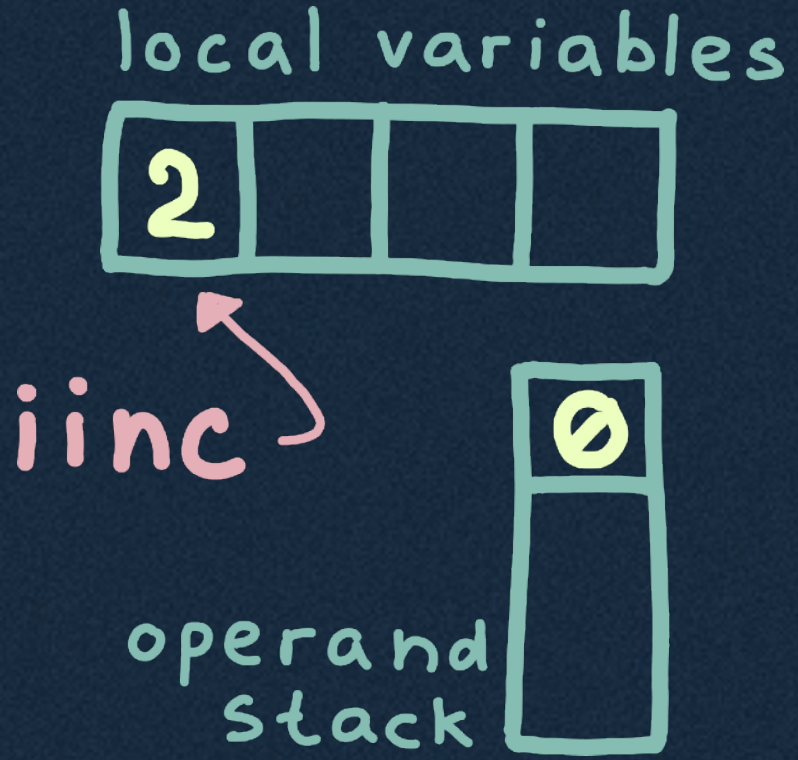
```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: iinc      0, 1
9: iload_0
10: iadd
11: istore_0
```



INCREMENT

```
int a = 0;
a = a++ + ++a;
```

```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: iinc      0, 1
9: iload_0
10: iadd
11: istore_0
```



increment

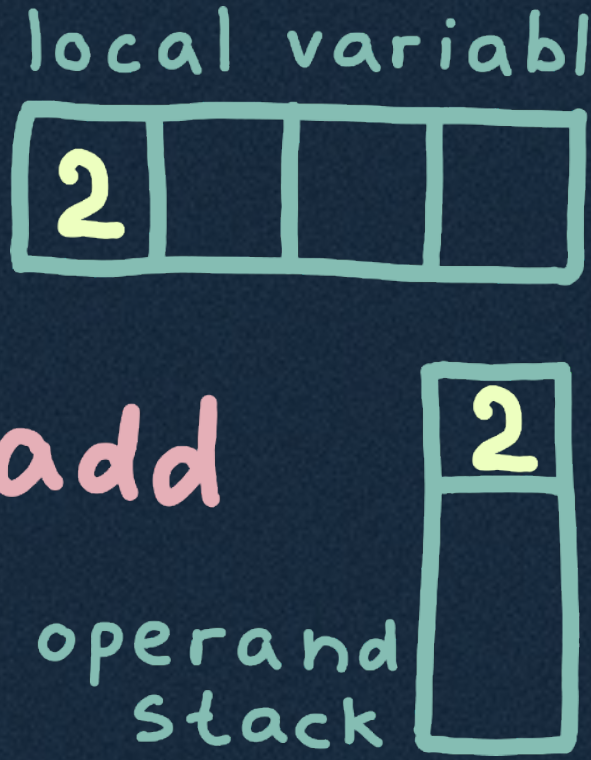
```
int a = 0;
a = a++ + ++a;
```

```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: iinc      0, 1
9: iload_0
10: iadd
11: istore_0
```

INCREMENT

```
int a = 0;
a = a++ + ++a;
```

```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: iinc      0, 1
9: iload_0
10: iadd
11: istore_0
```



INCREMENT

```
int a = 0;
a = a++ + ++a;
```

```
0: iconst_0
1: istore_0
2: iload_0
3: iinc      0, 1
6: iinc      0, 1
9: iload_0
10: iadd
11: istore_0
```

istore_0

conclusions

- increment/decrement in expressions is BAD

conclusions

- increment/decrement in expressions is BAD
- assignment is redundant

CONCATENATION OPERATOR +

CONCATENATION OPERATOR +

00 : 15

```
System.out.println(1 + 5 + "text" + 5 + 1);
```

CONCATENATION OPERATOR +

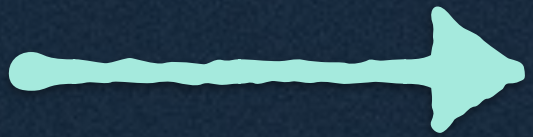
```java
System.out.println(1 + 5 + "text" + 5 + 1);
```

```
0: getstatic     #7
3: ldc           #13 // String 6text51
5: invokevirtual #15
8: return
```

CONCATENATION OPERATOR +

```
System.out.println(1 + 5 + "text" + 5 + 1);
```

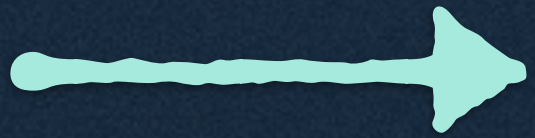
1 + 5 + "text" + 5 + 1



CONCATENATION OPERATOR +

```
System.out.println(1 + 5 + "text" + 5 + 1);
```

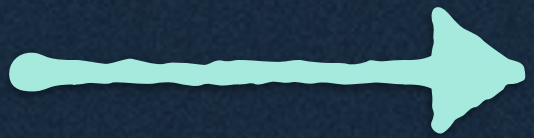
6 + "text" + 5 + 1



CONCATENATION OPERATOR +

```
System.out.println(1 + 5 + "text" + 5 + 1);
```

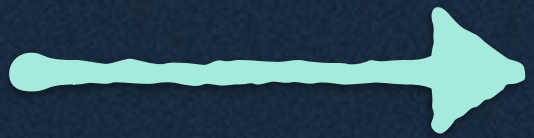
"6text" + 5 + 1



CONCATENATION OPERATOR +

```
System.out.println(1 + 5 + "text" + 5 + 1);
```

"6text5" + 1



CONCATENATION OPERATOR +

```
System.out.println(1 + 5 + "text" + 5 + 1);
```

"6text51"

CONCATENATION OPERATOR +

```
boolean cond = args[0].contains("true");
var unknown = cond ? 1 : "text";
System.out.println(unknown + "text" + 5 + 1);
```

CONCATENATION OPERATOR +

```
boolean cond = args[0].contains("true");
var unknown = cond ? 1 : "text";
System.out.println(unknown + "text" + 5 + 1);
```

```
boolean cond = args[0].contains("true");
var unknown = cond ? 1 : "text";
System.out.println(unknown + 1 + "text" + 5 + 1);
```

conclusions

- TWO OPERANDS AT A TIME

conclusions

- TWO OPERANDS AT A TIME
- FROM LEFT TO RIGHT

EXPRESSIONS

EXPRESSIONS

```
System.out.println(1 / 2 + 3.0 * 1 / 2);
```

EXPRESSIONS

```
System.out.println(1 / 2 + 3.0 * 1 / 2);
```

```
0: getstatic     #7
3: ldc2_w        #13 // double 1.5d
6: invokevirtual #15
9: return
```

EXPRESSIONS

```
System.out.println(1 / 2 + 3 * 1 / 2);
```

1 / 2 + 3 * 1 / 2

EXPRESSIONS

```
System.out.println(1 / 2 + 3 * 1 / 2);
```

1 / 2 + 3 * 1 / 2

EXPRESSIONS

```
System.out.println(1 / 2 + 3 * 1 / 2);
```

0 + 3 * 1 / 2

EXPRESSIONS

```
System.out.println(1 / 2 + 3 * 1 / 2);
```

EXPRESSIONS

```
System.out.println(1 / 2 + 3 * 1 / 2);
```

0 + 3 / 2

EXPRESSIONS

```
System.out.println(1 / 2 + 3 * 1 / 2);
```

0 + 3 / 2

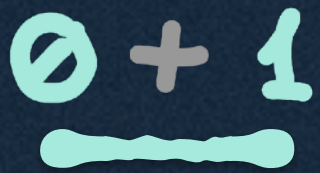
Expressions

```
System.out.println(1 / 2 + 3 * 1 / 2);
```

0 + 1

EXPRESSIONS

```
System.out.println(1 / 2 + 3 * 1 / 2);
```



EXPRESSIONS

```
System.out.println(1 / 2 + 3 * 1 / 2);
```

EXPRESSIONS

```
System.out.println(1 / 2 + 3.0 * 1 / 2);
```

1/2+3.0*1/2

EXPRESSIONS

```
System.out.println(1 / 2 + 3.0 * 1 / 2);
```

1/2+3.0*1/2

EXPRESSIONS

```
System.out.println(1 / 2 + 3.0 * 1 / 2);
```

0 + 3.0 * 1 / 2

EXPRESSIONS

```
System.out.println(1 / 2 + 3.0 * 1 / 2);
```

0 + 3.0 * 1 / 2

EXPRESSIONS

```
System.out.println(1 / 2 + 3.0 * 1 / 2);
```

0 + 3.0 / 2

EXPRESSIONS

```
System.out.println(1 / 2 + 3.0 * 1 / 2);
```

0 + 3.0 / 2

EXPRESSIONS

```
System.out.println(1 / 2 + 3.0 * 1 / 2);
```

0 + 1.5

EXPRESSIONS

```
System.out.println(1 / 2 + 3.0 * 1 / 2);
```

0 + 1.5

EXPRESSIONS

```
System.out.println(1 / 2 + 3.0 * 1 / 2);
```

1.5

BONUS!

EXPRESSIONS


```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

(10/4)*10.0/4+10/4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

(10/4) * 10.0 / 4 + 10/4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

2 *10.0/4+10/4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

2.0 * 10.0 / 4 + 10 / 4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

2.0 * 10.0 / 4 + 10 / 4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

2.0 * 10 /4 + 10/4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

2.0 * 10 /4 + 10/4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

20.0 /4 + 10/4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

20.0 / 4 + 10 / 4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

5.0 + 10/4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

5.0 + 10/4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

5.0 + 10.0 / 4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

5.0 + 10.0 / 4

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

5.0 + 2.5

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

5.0 + 2.5

EXPRESSIONS

```
System.out.println((double) (10 / 4) * (int) 10.0 / 4 + (double) 10 / 4);
```

7.5

conclusions

- WHAT IS NEXT BY PRIORITY

conclusions

- WHAT IS NEXT BY PRIORITY
- WHAT IS THE DOMINATING TYPE

WIDENING

WIDENING

00 : 15

```
short a = 10000;
short b = 20000;
short c = a + b;
```

WIDENING

```
short a = 10000;
short b = 20000;
short c = (short)(a + b); // c == 30000
```

WIDENING

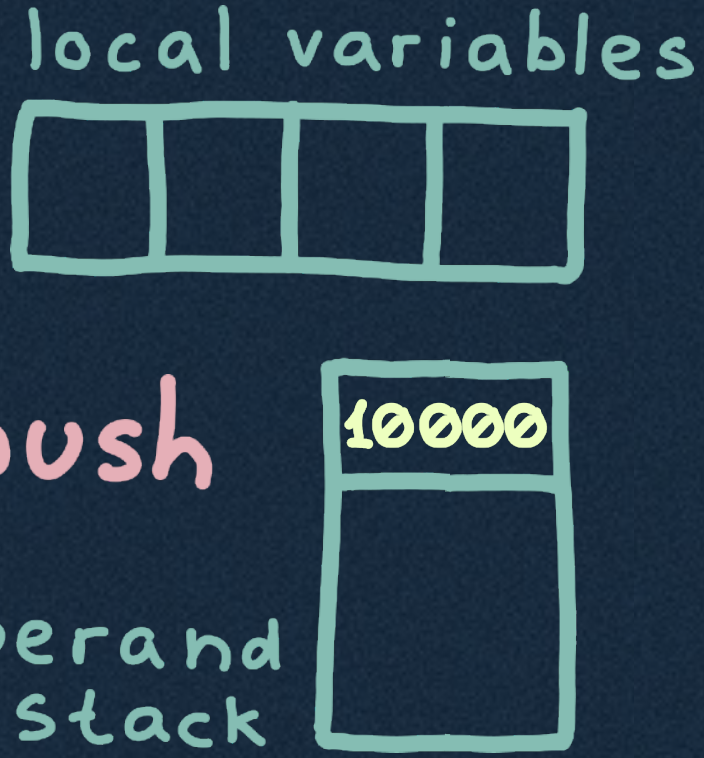
```
short a = 10000;
short b = 20000;
short c = (short)(a + b); // c == 30000
```

```
0: sipush        10000
3: istore_0
4: sipush        20000
7: istore_1
8: iload_0
9: iload_1
10: iadd
11: i2s
12: istore_2
```

WIDENING

```
short a = 10000;
short b = 20000;
short c = (short)(a + b);
```

```
0: sipush        10000
3: istore_0
4: sipush        20000
7: istore_1
8: iload_0
9: iload_1
10: iadd
11: i2s
12: istore_2
```



WIDENING

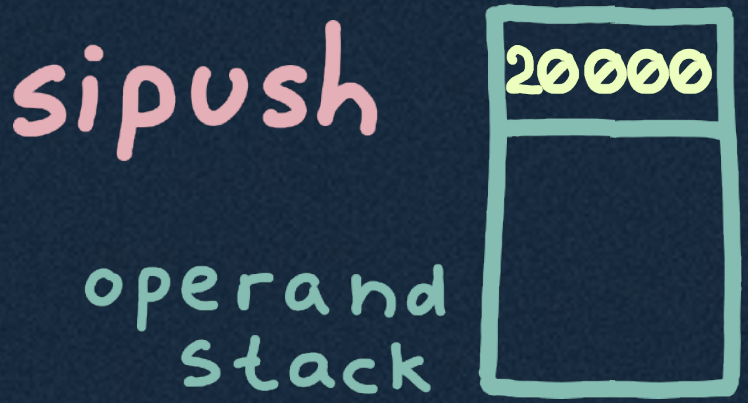
```
short a = 10000;
short b = 20000;
short c = (short)(a + b);
```

```
0: sipush        10000
3: istore_0
4: sipush        20000
7: istore_1
8: iload_0
9: iload_1
10: iadd
11: i2s
12: istore_2
```

WIDENING

```
short a = 10000;
short b = 20000;
short c = (short)(a + b);
```

```
0: sipush        10000
3: istore_0
4: sipush        20000
7: istore_1
8: iload_0
9: iload_1
10: iadd
11: i2s
12: istore_2
```



WIDENING

```
short a = 10000;
short b = 20000;
short c = (short)(a + b);
```

```
0: sipush        10000
3: istore_0
4: sipush        20000
7: istore_1
8: iload_0
9: iload_1
10: iadd
11: i2s
12: istore_2
```

WIDENING

```
short a = 10000;
short b = 20000;
short c = (short)(a + b);
```

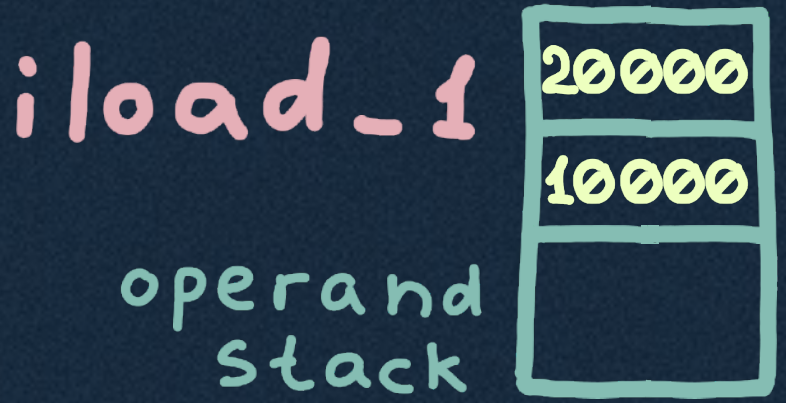
```
0: sipush        10000
3: istore_0
4: sipush        20000
7: istore_1
8: iload_0
9: iload_1
10: iadd
11: i2s
12: istore_2
```

iload_0

WIDENING

```
short a = 10000;
short b = 20000;
short c = (short)(a + b);
```

```
0: sipush        10000
3: istore_0
4: sipush        20000
7: istore_1
8: iload_0
9: iload_1
10: iadd
11: i2s
12: istore_2
```

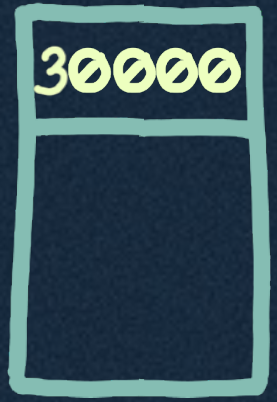


WIDENING

```
short a = 10000;
short b = 20000;
short c = (short)(a + b);
```

```
0: sipush        10000
3: istore_0
4: sipush        20000
7: istore_1
8: iload_0
9: iload_1
10: iadd
11: i2s
12: istore_2
```

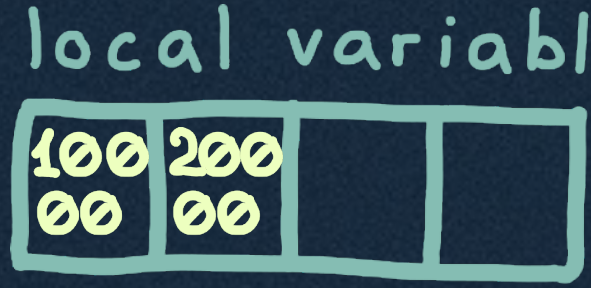
iadd



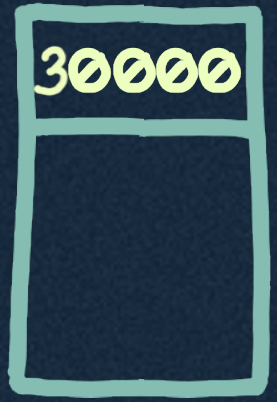
WIDENING

```
short a = 10000;
short b = 20000;
short c = (short)(a + b);
```

```
0: sipush        10000
3: istore_0
4: sipush        20000
7: istore_1
8: iload_0
9: iload_1
10: iadd
11: i2s
12: istore_2
```



i2s



WIDENING

```
short a = 10000;
short b = 20000;
short c = (short)(a + b);
```

```
0: sipush        10000
3: istore_0
4: sipush        20000
7: istore_1
8: iload_0
9: iload_1
10: iadd
11: i2s
12: istore_2
```

WIDENING PRIMITIVE CONVERSION

LITTLE TRICK

DOESN'T COMPILE!

```
short a = 10000;
short b = 20000;
short c = a + b;
```

COMPILES!

```
final short a = 10000;
final short b = 20000;
short c = a + b;
```

DOESN'T COMPILE!

```
short a = 10000;
short b = 20000;
short c = a + b;
```

COMPILES!

```
final short a = 10000;
final short b = 20000;
short c = a + b;
```

```
 0: sipush        10000
 3: istore_1
 4: sipush        20000
 7: istore_2
 8: sipush        30000
11: istore_3
```

DOESN'T COMPILE!

```
short a = 10000;
short b = 20000;
short c = a + b;
```

COMPILES!

```
final short a = 10000;
final short b = 20000;
short c = a + b;
```

CONSTANT FOLDING

```
 0: sipush        10000
 3: istore_1
 4: sipush        20000
 7: istore_2
 8: sipush        30000
11: istore_3
```


```
final short a = 20000;
final short b = 20000;
short c = a + b;
```

conclusions

JVM DOESNT DEAL WITH:

conclusions

JVM DOESNT DEAL WITH:

- SHORT

conclusions

JVM DOESNT DEAL WITH:

- SHORT
- BYTE

conclusions

JVM DOESNT DEAL WITH:

- SHORT
- BYTE
- CHAR

conclusions

JVM DOESNT DEAL WITH:

- SHORT
- BYTE
- CHAR
- BOOLEAN

conclusions

JVM DOESNT DEAL WITH:

- SHORT
- BYTE
- CHAR
- BOOLEAN

THEY ARE CONVERTED TO INT

UNREACHABLE STATEMENTS

UNREACHABLE STATEMENTS

00 : 15

```
int a = 0;
while (false) {
    a = 10;
}
```

UNREACHABLE STATEMENTS

```
int a = 0;
while (false) {
    a = 10;
}
```

```
int a = 0;
if (false) {
    a = 10;
}
```

UNREACHABLE STATEMENTS

```
int a = 0;
while (false) {
    a = 10;
}
```

DOESN'T COMPILE!

```
int a = 0;
if (false) {
    a = 10;
}
```

COMPILES!

UNREACHABLE STATEMENTS

```
int a = 0;
if (false) {
    a = 10;
}
```

```
static final boolean FEATURE = false;
```

UNREACHABLE STATEMENTS

```
int a = 0;
int b = 10;
while (b + b < 10) {
    a = 1;
}
```

COMPILES!

```
int a = 0;
final int b = 10;
while (b + b < 10) {
    a = 1;
}
```

DOESN'T COMPILE!

UNREACHABLE STATEMENTS

CONSTANT FOLDING

```
int a = 0;
int b = 10;
while (b + b < 10) {
    a = 1;
}
```

COMPILES!

```
int a = 0;
final int b = 10;
while (b + b < 10) {
    a = 1;
}
```

DOESN'T COMPILE!

CONCLUSIONS & TIPS

CONCLUSIONS & TIPS

- PUT THE SIDE EFFECTS ... ASIDE

CONCLUSIONS & TIPS

- PUT THE SIDE EFFECTS ... ASIDE
- HOW TO THINK LIKE A COMPILER:

CONCLUSIONS & TIPS

- PUT THE SIDE EFFECTS ... ASIDE
- HOW TO THINK LIKE A COMPILER:
    - TWO AT A TIME

CONCLUSIONS & TIPS

- PUT THE SIDE EFFECTS ... ASIDE
- HOW TO THINK LIKE A COMPILER:
  - TWO AT A TIME
  - HIGHEST PRIORITY FIRST

CONCLUSIONS & TIPS

- PUT THE SIDE EFFECTS ... ASIDE
- HOW TO THINK LIKE A COMPILER:
  - TWO AT A TIME
  - HIGHEST PRIORITY FIRST
  - TYPE BATTLE!

CONCLUSIONS & TIPS

- PUT THE SIDE EFFECTS ... ASIDE
- HOW TO THINK LIKE A COMPILER:
  - TWO AT A TIME
  - HIGHEST PRIORITY FIRST
  - TYPE BATTLE!
- COMPILERS LIE... YET, TRUST THEM

CONCLUSIONS & TIPS

- PUT THE SIDE EFFECTS ... ASIDE
- HOW TO THINK LIKE A COMPILER:
  - TWO AT A TIME
  - HIGHEST PRIORITY FIRST
  - TYPE BATTLE!
- COMPILERS LIE... YET, TRUST THEM
- APPRETIATE WHEN COMPILER COMPLAINS

CONCLUSIONS & TIPS

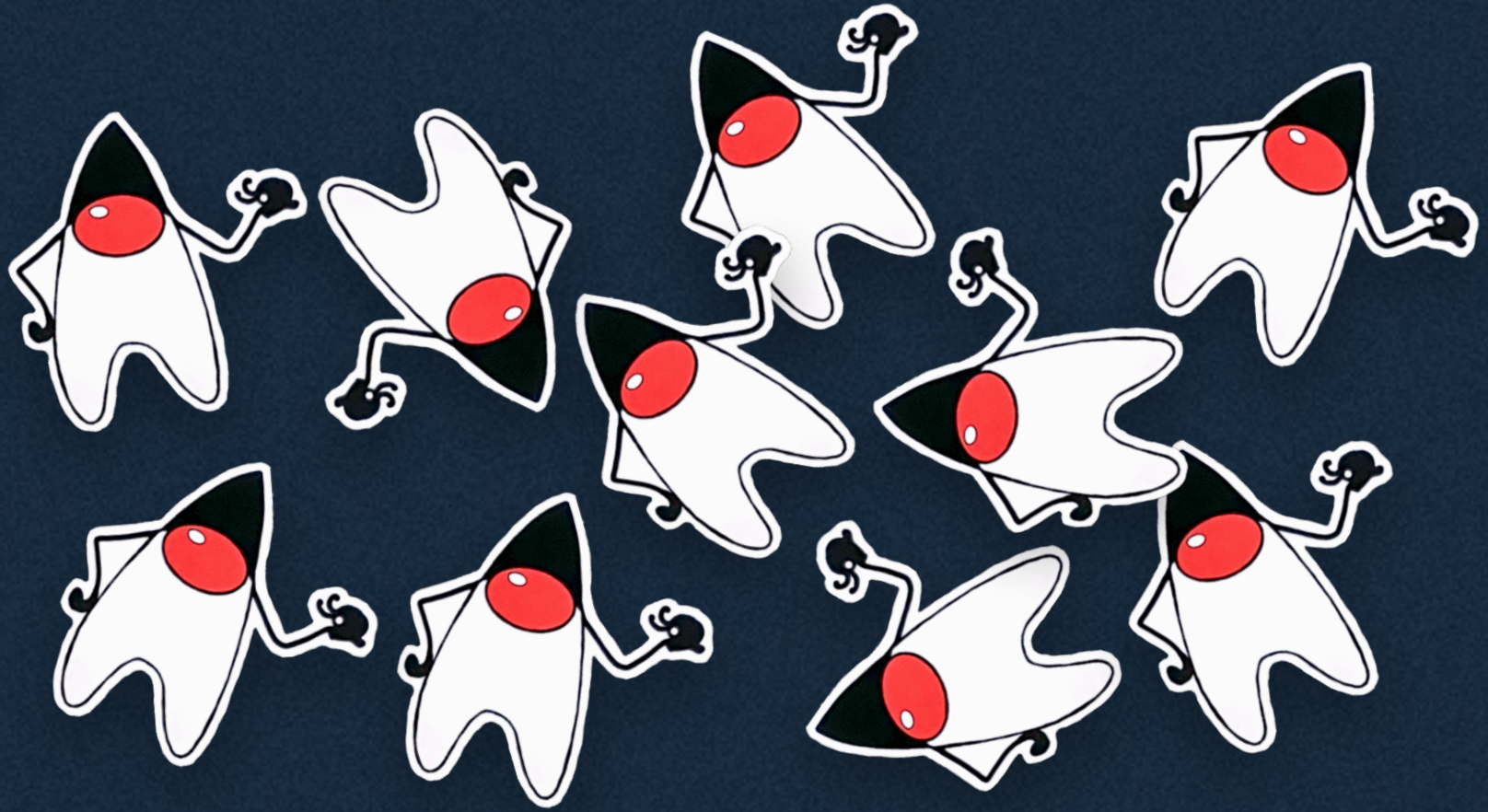
- PUT THE SIDE EFFECTS ... ASIDE
- HOW TO THINK LIKE A COMPILER:
  - TWO AT A TIME
  - HIGHEST PRIORITY FIRST
  - TYPE BATTLE!
- COMPILERS LIE... YET, TRUST THEM
- APPRETIATE WHEN COMPILER COMPLAINS
- HELP YOUR COMPILER OPTIMIZE

CONCLUSIONS & TIPS

- PUT THE SIDE EFFECTS ... ASIDE
- HOW TO THINK LIKE A COMPILER:
  - TWO AT A TIME
  - HIGHEST PRIORITY FIRST
  - TYPE BATTLE!
- COMPILERS LIE... YET, TRUST THEM
- APPRETIATE WHEN COMPILER COMPLAINS
- HELP YOUR COMPILER OPTIMIZE
- BUT WRITE CODE FOR DEVS

CONCLUSIONS & TIPS

- PUT THE SIDE EFFECTS ... ASIDE
- HOW TO THINK LIKE A COMPILER:
  - TWO AT A TIME
  - HIGHEST PRIORITY FIRST
  - TYPE BATTLE!
- COMPILERS LIE... YET, TRUST THEM
- APPRETIATE WHEN COMPILER COMPLAINS
- HELP YOUR COMPILER OPTIMIZE
- BUT WRITE CODE FOR DEVS



@ WORTH_EXPLORING

THANKS FOR YOUR

THANKS FOR YOUR ATTENTION

THANKS FOR YOUR

PLEASE RATE MY SESSION
IN THE NLJUG EVENT APP