National Research University Higher School of Economics
Faculty of Computer Science
Bachelor's Program "HSE University and University of London Double Degree Program in Data Science and Business Analytics"

Introduction to Programming

Workshop #3

Wed 20.01.2021

Julio Carrasquel

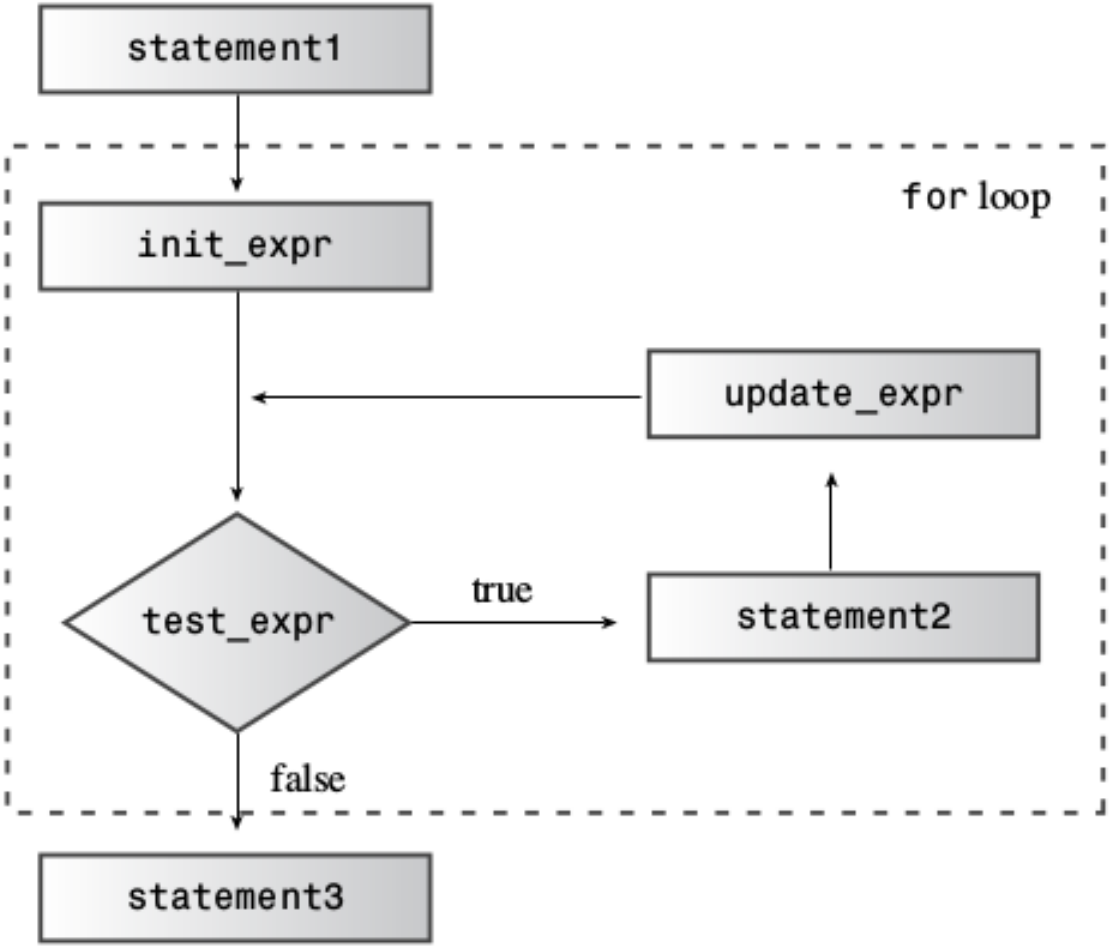
NATIONAL RESEARCH UNIVERSITY

Outline

- Loops
- Conditionals
- Control-flow statements

Statement *for*

```
statement1
for (int_expr; test_expr; update_expr)
     statement2
statement3
```



Operators ++ and --

- Increase and decrease of the operand by one step.

  Postfix version: `a++`

  `a++` means *"use the current value of* `a` *in evaluating an expression, and then increment the value of* `a`*."*

  Prefix version: `++b`

  `++b` means *"increment the value of* `b`*, and then use the current value of and then use the new value of* `b` *in evaluating the expression."*

Operators ++ and --

- Increase and decrease of the operand by one step.

```
int x = 5;
int y = ++x;  // change x, then assign to y
              // y is 6, x is 6

int z = 5;    // assign to y, then change z
int y = z++;  // y is 5, z is 6
```

Operators ++ and --

- Do the following pieces of code have the same effect?.

```
for (n = lim; n > 0; --n)
```

```
for (n = lim; n > 0; n--)
```

Operators ++ and --

- Do the following pieces of code have the same effect?.

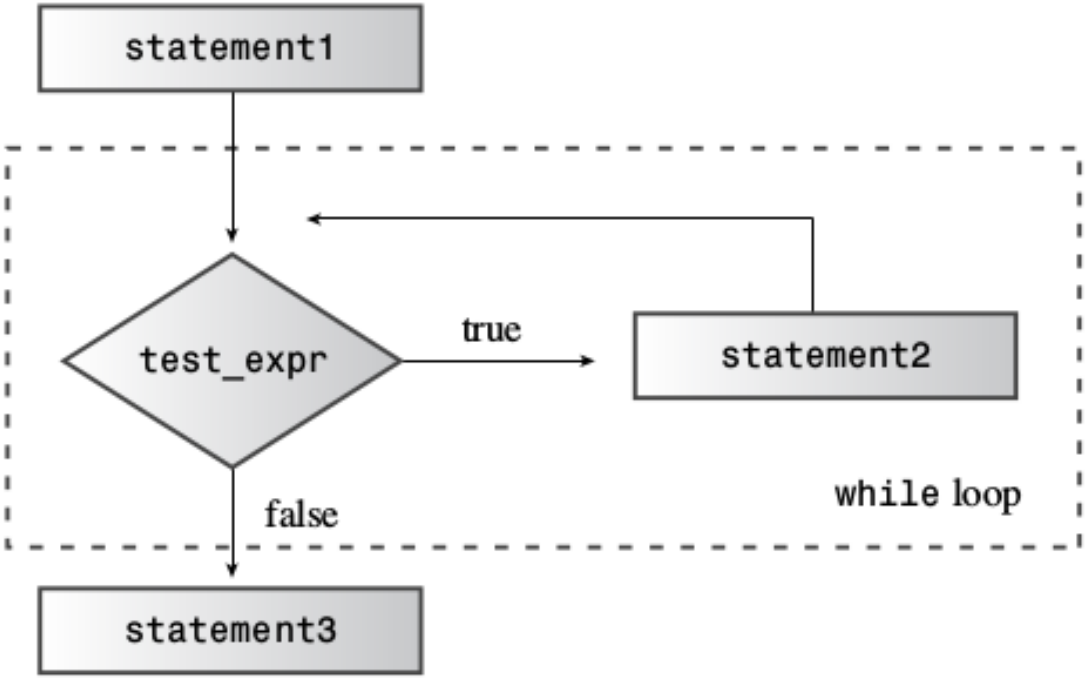
```
for (n = lim; n > 0; --n)        for (n = lim; n > 0; n--)
```

Yes! Value of **n** is not used exactly when updating the variable.

- For built-in types (int, float, etc) no difference.
- For user-defined types having user-defined operators **--** or **++**, use the prefix form **--n** or **++n**. More efficient!

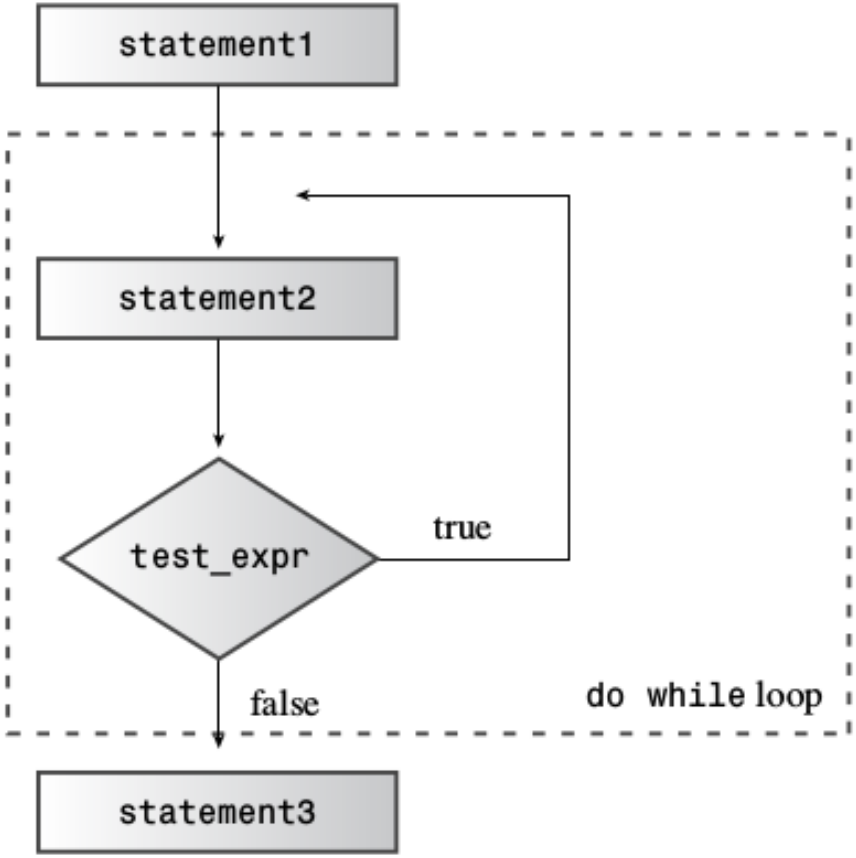
Statement *while*

```
statement1
while (test_expr)
      statement2
statement3
```



Statement *do-while*

```
statement1
do
      statement2
while (test_expr);
statement3
```



Exercise

- Print out using a *nested loop* the multiplication table

1 2 3 4 5 6 7 8 9

2 4 6 8 10 12 14 16 18

…

9 18 …

Range-based *for* loops (from C++11)

```
double prices[5] = {4.99, 10.99, 6.87, 7.99, 8.49};


for (double x : prices)
   std :: cout << x << std :: endl;


for (double &x : prices) // & is a referencing oper. (to see later)
   x = x * 0.80; //20% off sale!


for (int x : {3, 5, 2, 8, 6})
   std :: cout << x << " ";
std :: cout << ‘\n’;
```

`if-else` forms of constructions (examples)

```
if (ch == 'A')
   a_grade++;   // alternative # 1
else
   if (ch == 'B') // alternative # 2
      b_grade++; // subalternative # 2a
   else
      soso++;    // subalternative # 2b
```

```
if (ch == 'A')
   a_grade++;   // alternative # 1
else if (ch == 'B')
   b_grade++; // alternative # 2
else
   soso++;    // alternative # 3
```

`if-else` forms of constructions (examples)

```
if (age > 17 && age < 35)
    index = 0;
else if (age >= 35 && age < 50)
    index = 1;
else if (age >= 50 && age < 65)
    index = 2;
else
    index = 3;
```

Conditional ternary operator “?”

```
expression1 ? expression2 : expression3
```

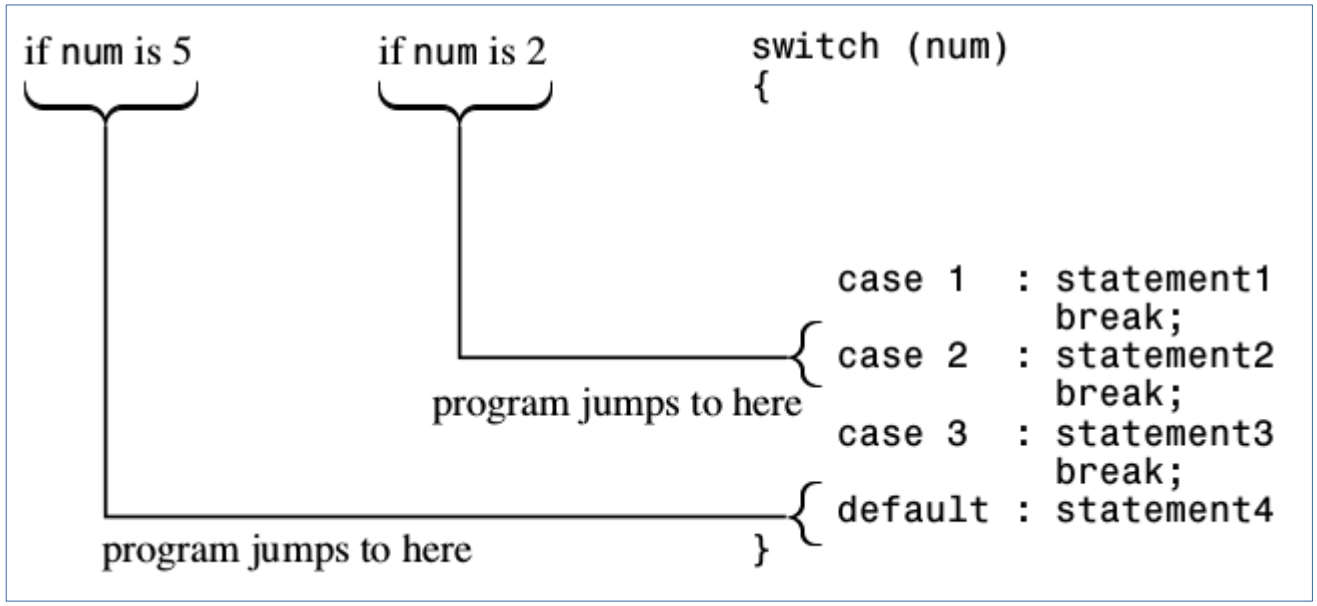
Conditional ternary operator “?”

```
expression1<boolean_expression> ? expression2<true_case> : expression3<false_case>
```

```
5 > 3 ? 10 : 12 // 5 > 3 is true, so expression value is 10
3 == 9? 25 : 18 // 3 == 9 is false, so expression value is 18
```

Statement `switch`

```
switch (integer-expression)
{
      case label1 : statement(s)
      case label2 : statement(s)
      ...
      default : statement(s)
}
```



Statement `switch` (works with chars too!)

```
switch(choice)
{
    case 'a':
    case 'A': cout << "\a\n";
              break;

    case 'r':
    case 'R': report();
              break;

    case 'l':
    case 'L': cout << "The boss was in all day.\n";
              break;

    case 'c':
    case 'C': comfort();
              break;

    default : cout << "That's not a choice.\n";
}
```

continue and break statements

```
while (cin.get(ch))
{
   statement1
   if (ch == '\n')
      continue;
   statement2
}
statement3
```

continue skips rest of loop body and starts a new cycle

```
while (cin.get(ch))
{
   statement1
   if (ch == '\n')
      break;
   statement2
}
statement3
```

break skips rest of loop and goes to following statement