for loop

Today

- do while loop
- for loop
- for vs. while loop

Due this week

- **Homework 3**
  - Start-early due today
  - Write solutions in VSCode and paste in Autograder, **Homework 3 CodeRunner**.
  - Zip your .cpp files and submit on canvas **Homework 3**.
- Start going through the textbook readings and watch the videos
  - Take **Quiz 4**.
- Check the due date! **No late submissions!!**
- Start practicum prep

do loop

The `do{ } while()` Loop

- The while() loop's condition test is the first thing that occurs in its execution.
- The do loop (or do-while loop) has its condition tested only after at least one execution of the statements. The test is at the bottom of the loop:

```
do
{
   statements
}
while (condition);
```

The `do` Loop

- This means that the `do` loop should be used only when the statements must be executed before there is any knowledge of the condition.

- This also means that the `do` loop is the least used loop.

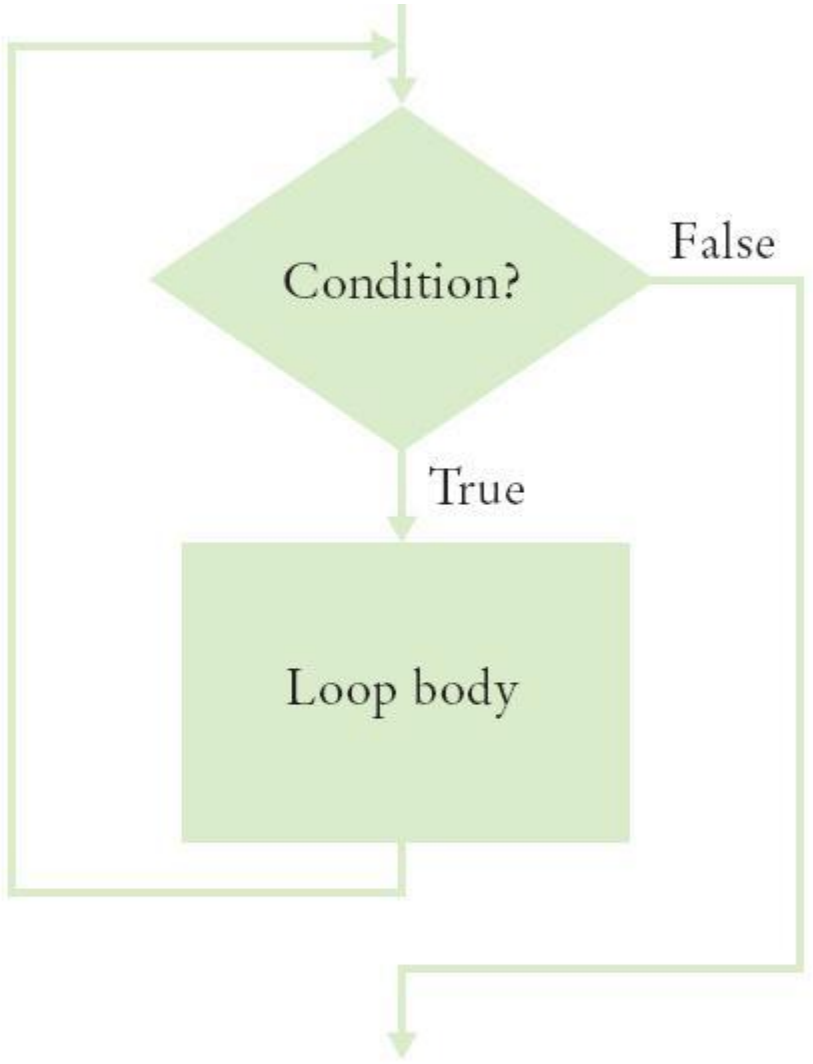
`do{ }` Loop Code: getting user input Repeatedly

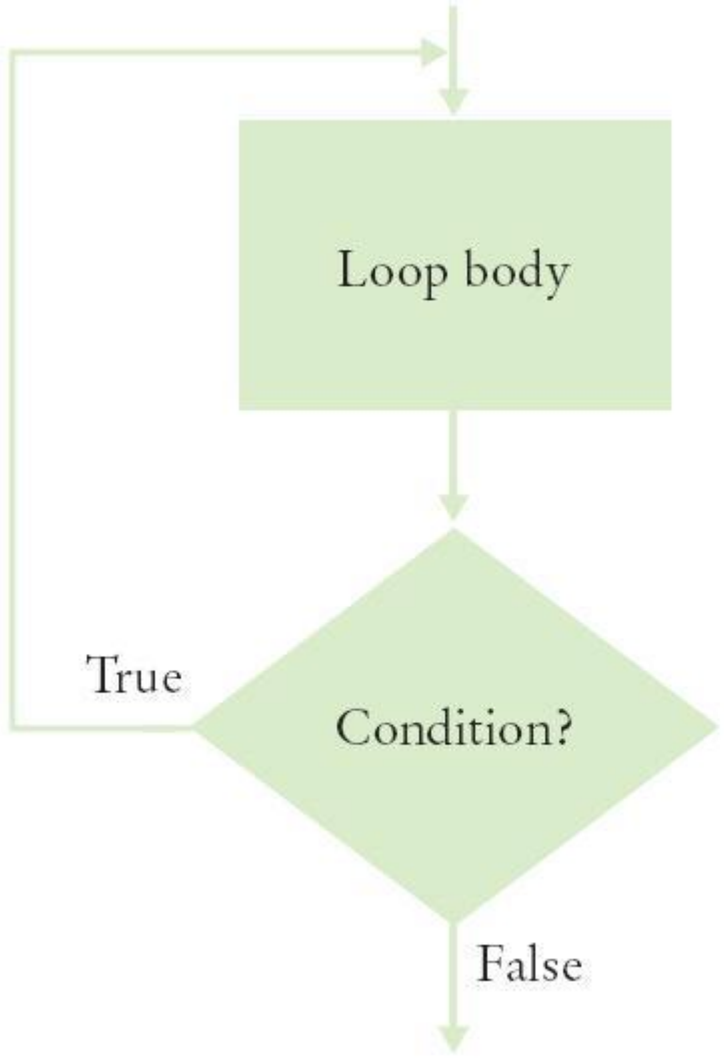
- Code to keep asking a user for input until it satisfies a condition, such as non-negative for applying the sqrt():

```
double value;
do
{
 cout << "Enter a number >= 0: ";
 cin >> value;
}
while (value < 0);

cout << "The square root is " << sqrt(value) << endl;
```

Flowcharts for the `while` Loop and the `do` Loop





for vs. while loop

The `for` Loop vs. the `while` loop

- Often you will need to execute a sequence of statements a given number of times.

  You could use a `while` loop:

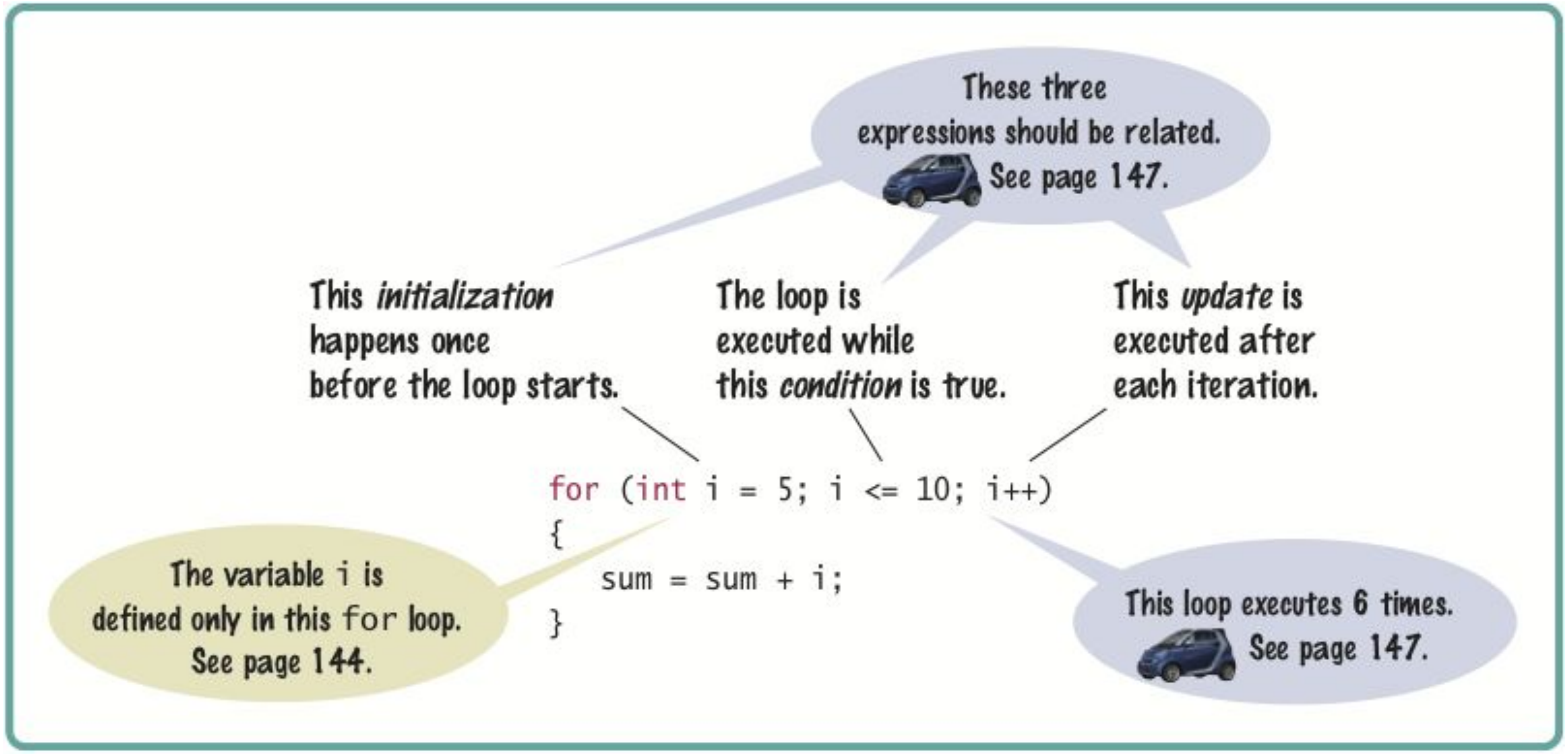
```
num = 1; // Initialize the variable
while (num <= 10) // Check the variable
{
    cout << num << endl;
    num++; // Update the variable
}
```

The `for` Loop

- C++ has a statement custom made ***for*** this sort of processing: the **`for`** loop.

```
for (num = 1; num <= 10; num++)
{
   cout << num << endl;
}
```

The `for` Loop Syntax

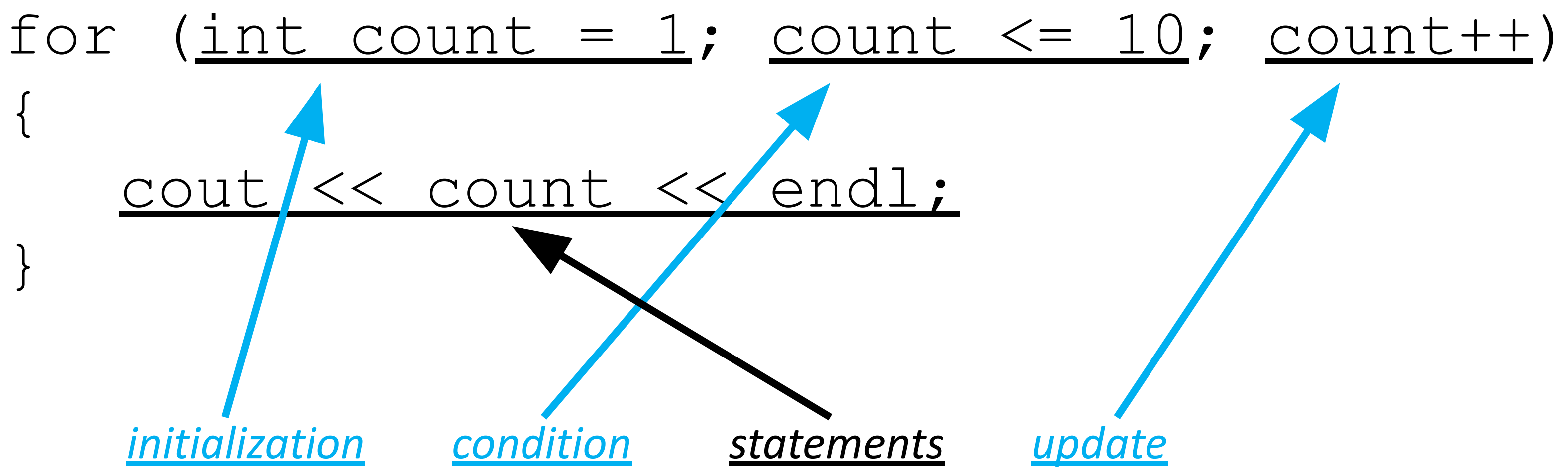


```
for (int i = 5; i <= 10; i++)
{
   sum = sum + i;
}
```

These three expressions should be related. See page 147.
This *initialization* happens once before the loop starts.
The loop is executed while this *condition* is true.
This *update* is executed after each iteration.
The variable i is defined only in this for loop. See page 144.
This loop executes 6 times. See page 147.

The `for` Loop Is Better than `while` for Certain Things

- Doing something a known number of times or causing a variable to take on a sequence of values is so common, C++ has a statement just for that:



```
for (int count = 1; count <= 10; count++)
{
    cout << count << endl;
}
```

`for()` loop execution

```
for (initialization; condition; update)
   {
      statements;
   }
```

- The ***initialization*** is code that happens once, before the check is made, to set up counting how many times the *statements* will happen. The loop variable may be created here, or before the `for()` statement.
- The ***condition*** is a comparison to test if the loop is done. When this test is false, we skip out of the `for()`, going on to the next statement.
- The ***update*** is code that is executed at the bottom of each iteration of the loop, immediate before re-testing the condition. Usually it is a counter increment or decrement.
- The ***statements*** are repeatedly executed until the condition is false. These also are known as the "loop body".

The `for` Can Count Up or Down

- A `for` loop can count down instead of up:

```
    for (int counter = 10; counter >= 0; counter--)…
```

- Notice that in this examples, the loop variable is defined **in** the *initialization* (where it really should be!).

`for` Loop Examples, Index Values

Loop	Values of i	Comment
for (i = 0; i <= 5; i++)	0 1 2 3 4 5	Note that the loop is executed 6 times. (See Programming Tip 4.3)
for (i = 5; i >= 0; i--)	5 4 3 2 1 0	Use i-- for decreasing values.
for (i = 0; i < 9; i = i + 2)	0 2 4 6 8	Use i = i + 2 for a step size of 2.
for (i = 0; i != 9; i += 2)	0 2 4 6 8 10 ... (infinite loop)	You can use < or <= instead of != to avoid this problem.
for (i = 1; i <= 20; i = i * 2)	1 2 4 8 16	You can specify any rule for modifying i, such as doubling it in every step.
for (i = 0; i < str.length(); i++)	0 1 2 … until the last valid index of the string str	In the loop body, use the expression str.substr(i, 1)to get a string containing the ith character.

Converting from a *while* loop to a *for* loop

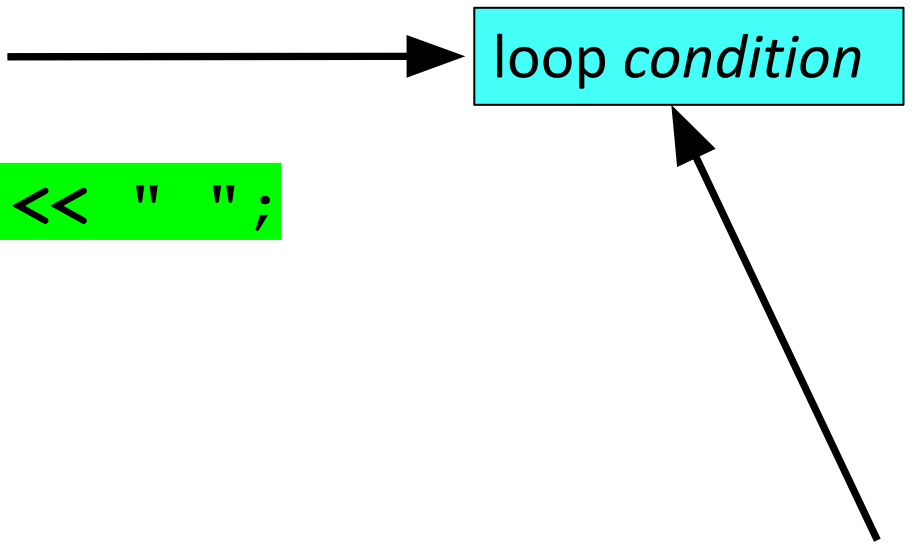
```
int i = 0;
while (i < 5)
{
    cout << i << " ";
    i++;
}
```

initialize loop variable *i:*
ONLY ONCE!

```
for (int i = 0; i < 5; i++)
{
    cout << i << " ";
}
```

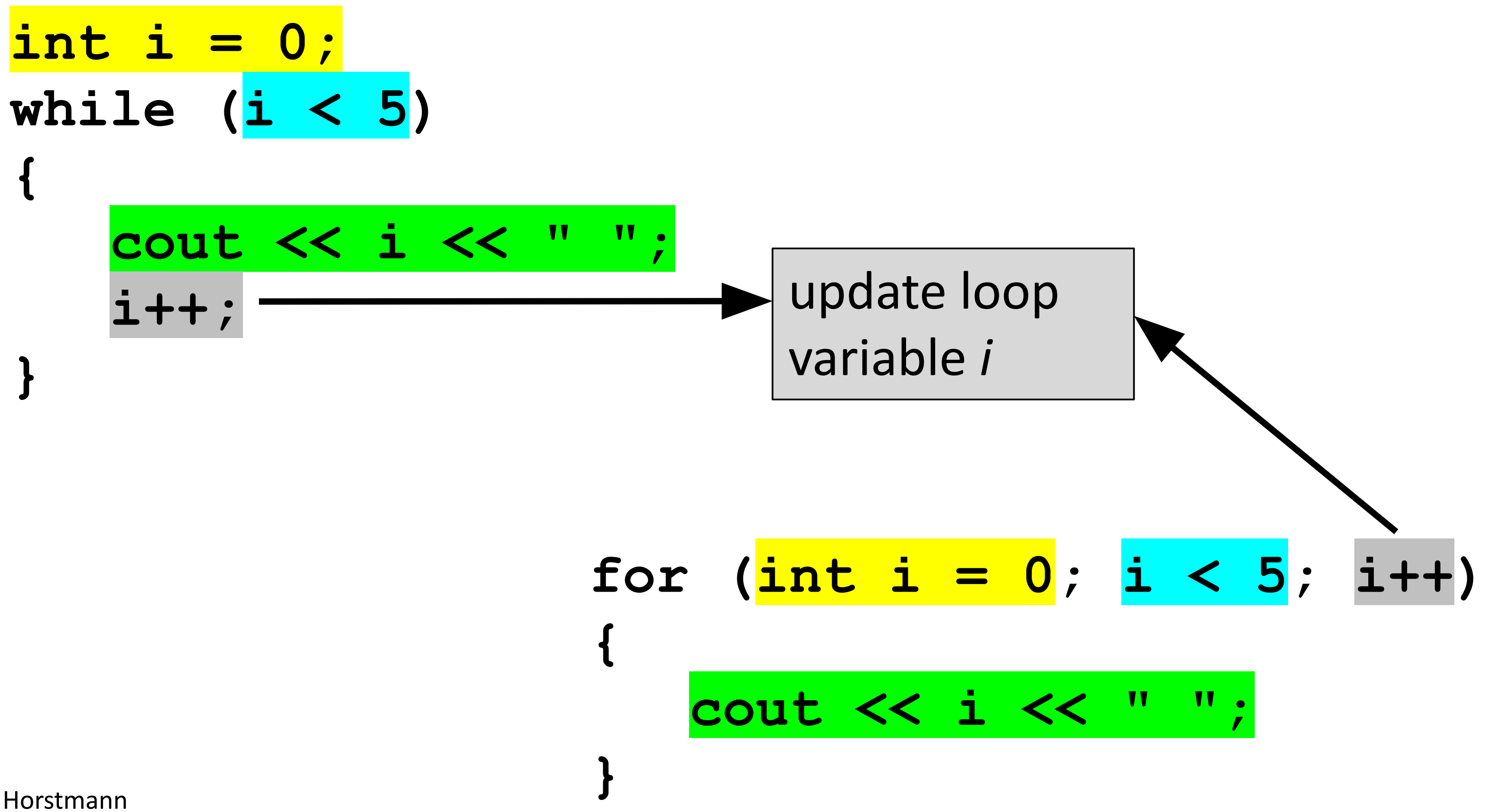
Converting from a *while* loop to a *for* loop

```
int i = 0;
while (i < 5)
{
    cout << i << " ";
    i++;
}
```



```
for (int i = 0; i < 5; i++)
{
    cout << i << " ";
}
```

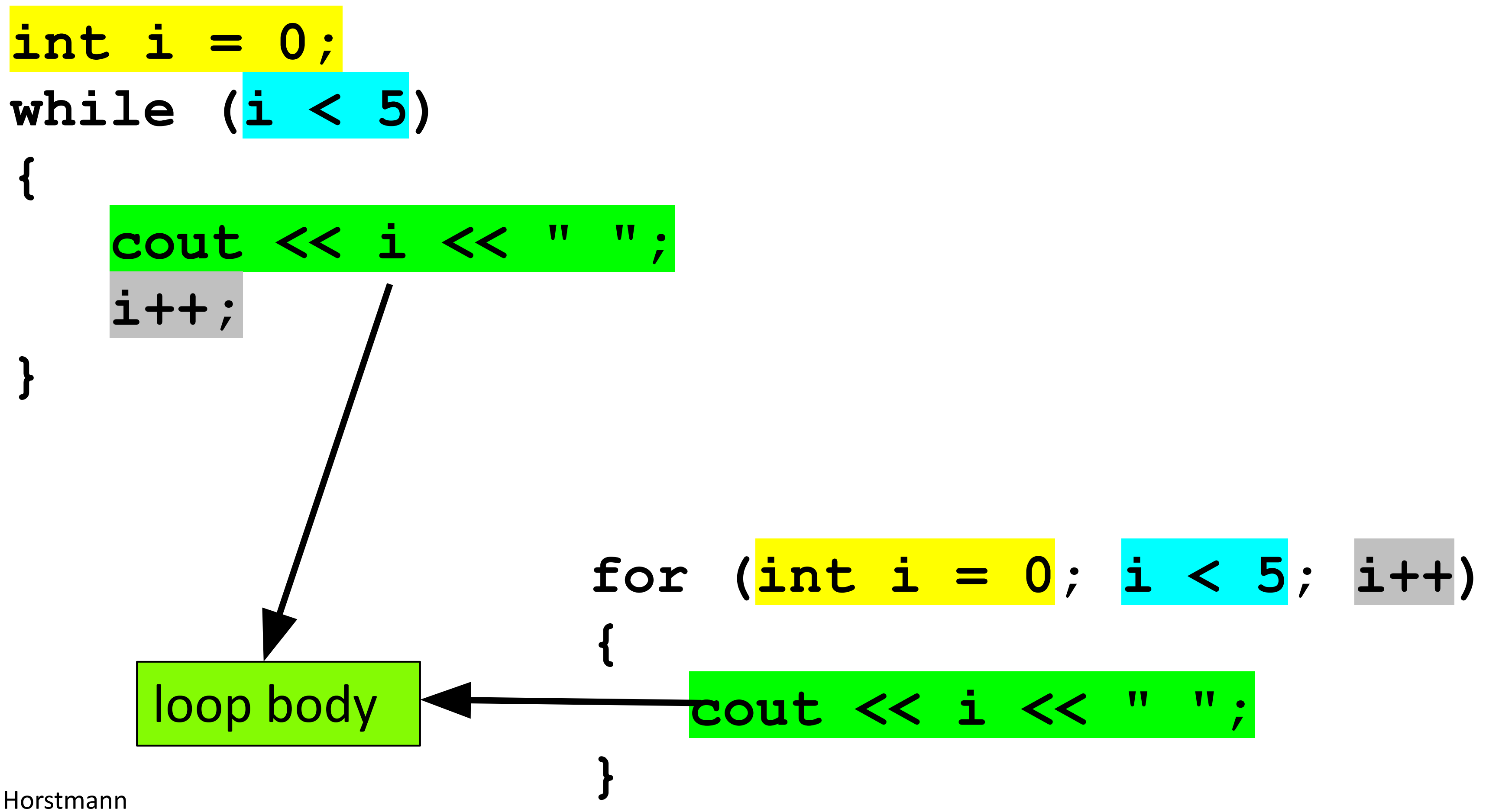
Converting from a *while* loop to a *for* loop



```
int i = 0;
while (i < 5)
{
    cout << i << " ";
    i++;
}
```

```
for (int i = 0; i < 5; i++)
{
    cout << i << " ";
}
```

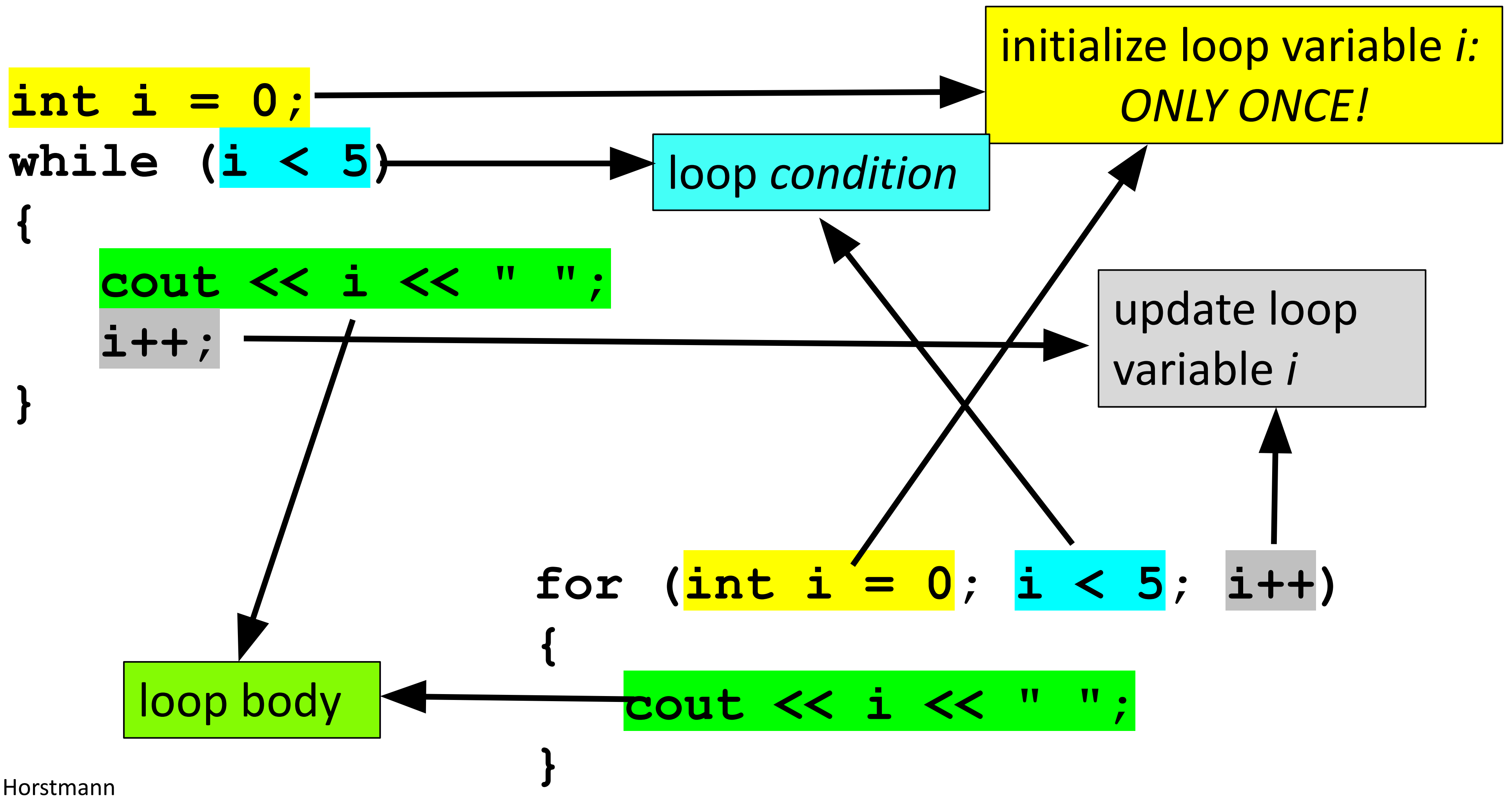
Converting from a *while* loop to a *for* loop



```
int i = 0;
while (i < 5)
{
    cout << i << " ";
    i++;
}
```

```
for (int i = 0; i < 5; i++)
{
   cout << i << " ";
}
```

Converting from a *while* loop to a *for* loop



```
int i = 0;
while (i < 5)
{
    cout << i << " ";
    i++;
}
```

```
for (int i = 0; i < 5; i++)
{
    cout << i << " ";
}
```