Lab 4

Diana Gage

Loops: for and while

Exam practice

Loops exercise

New topic: LOOPS

- What is a loop?

New topic: LOOPS

- What is a loop?
  - A block of code that is executed **repeatedly**
  - Until a certain condition is met

- We need to be familiar with a few more operators for loops…

++i and i++

- These are called the increment operators
- They increment i by one

- i++ happens after i does its job
  - This is called the post-increment operator

- ++i happens before i does its job
  - This is called the pre-increment operator

--i and i--

- These are called the decrement operators
- They decrement i by one

- i-- happens after i does its job
  - This is called the post-decrement operator

- --i happens before i does its job
  - This is called the pre-increment operator

Different Kinds of Loops

- Count-controlled
  - Ends after a counter reaches a certain value
  - For Example: until count becomes > 35
- Event-Controlled
  - Ends when a certain event occurs, not based on a count
  - Example: as long as the input was valid

Different kinds of Loops

- for loop
- while loop
- Anything you can do with a for loop, you *can* do with a while loop, and vice versa
  - **But there are guidelines for which is better in what scenario**

Different kinds of Loops

- for loop
- while loop
- **If the loop will be count-controlled:**
  - **A *for* loop is usually best**
- **If the loop will be event-controlled:**
  - **Use a *while* loop**

While loops

- How would we write a while loop that prints the integers from 0 to 4?

While loops

```
int x = 0;

while (x < 5){
    cout << x << endl;
    ++x;
}
```

While loops

```
while (x < 5){
    cout << x << endl;
    ++x;
}
```

- **The (x < 5) is the…**

While loops

```
while (x < 5){
     cout << x << endl;
     ++x;
}
```

- **The (x < 5) is the…** condition

While loops

```
while (x < 5){
    cout << x << endl;
    ++x;
}
```

- The (x < 5) is the… condition
- **The cout << x << endl is the…**

While loops

```
while (x < 5){
    cout << x << endl;
    ++x;
}
```

- The (x < 5) is the… condition
- **The cout << x << endl is the…** loop body

While loops

```
while (x < 5){
    cout << x << endl;
    ++x;
}
```

- The (x < 5) is the… condition
- The cout << x << endl is the… loop body
- **The ++x is the…**

While loops

```
while (x < 5){
    cout << x << endl;
    ++x;
}
```

- The (x < 5) is the… condition
- The cout << x << endl is the… loop body
- **The ++x is the…** update (part of loop body)

While loops

```
int x = 0;

while (x < 5){
    cout << ++x << endl;
    ~~++x;~~
}
```

What would happen if the increment happened as above?

While loops

```
int x = 0;


while (x < 5){
    cout << ++x << endl;
    ~~++x;~~
}
```

What would happen if the increment happened as above?

increment x, and then print this new value of x

While loops

```
int x = 0;

while (x < 5){
    cout << ++x << endl;
    ~~++x;~~
}
```

What would happen if the increment happened this way instead?

print x is it is, and then **increment** x

For loops

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

What are the differences here?

For loops

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

What are the differences here?

- Everything happens inside parentheses
  - Initialization (int i = 0)
  - Condition (i < 5)
  - Update (++i)

  - **But its all still there!**

Difference in scope

```
int x = 0;
while (x < 5){
    cout << x << endl;
    ++x;
}
```
	```
	for (int i = 0; i < 5; ++i){
	    cout << i << endl;
	}
	```

- **What is the scope of x?**

Difference in scope

```
int x = 0;
while (x < 5){
    cout << x << endl;
    ++x;
}
```

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

- **What is the scope of x?**
  - whatever function the while loop is in (could be main)

Difference in scope

`int x = 0;`
`while (x < 5){`
`    cout << x << endl;`
`    ++x;`
`}`
	`for (int i = 0; i < 5; ++i){`
	`    cout << i << endl;`
	`}`

- What is the scope of x?
  - whatever function the while loop is in (could be main)
- **What is the scope of i?**

Difference in scope

```
int x = 0;
while (x < 5){
    cout << x << endl;
    ++x;
}
```

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

- What is the scope of x?
  - whatever function the while loop is in (could be main)
- **What is the scope of i?**
  - Only exists inside the for loop

Difference in scope

```
int x = 0;
while (x < 5){
    cout << x << endl;
    ++x;
}
```

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

- What is the scope of x?
  - whatever function the while loop is in (could be main)
- What is the scope of i **now?**

Difference in scope

```
int x = 0;
while (x < 5){
    cout << x << endl;
    ++x;
}
```

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

- What is the scope of x?
  - whatever function the while loop is in (could be main)
- What is the scope of i **now?**
  - **Now i exists in whatever function the for loop is in**

Nested Loops Example

```
for (int i = 0; i < 3; i++) {
    for (int j = 2; j >= 0; j--) {
        cout << 3 * i + j << " ";
    }
}
```

What prints?

Nested Loops Example

```
for (int i = 0; 0 < 3; i++) {
    for (int j = 2; 2 >= 0; j--) {
        cout << 3 * 0 + 2 << " ";
    }
}
```

What prints?

2

Nested Loops Example

```
for (int i = 0; 0 < 3; i++) {
    for (int j = 2; 1 >= 0; j--) {
        cout << 3 * 0 + 1 << " ";
    }
}
```

What prints?

2 1

Nested Loops Example

```
for (int i = 0; 0 < 3; i++) {
    for (int j = 2; 0 >= 0; j--) {
        cout << 3 * 0 + 0 << " ";
    }
}
```

What prints?

2 1 0

Nested Loops Example

```
for (int i = 0; 1 < 3; i++) {
    for (int j = 2; 2 >= 0; j--) {
        cout << 3 * 1 + 2 << " ";
    }
}
```

What prints?

2 1 0 5

Nested Loops Example

```
for (int i = 0; 1 < 3; i++) {
    for (int j = 2; 1 >= 0; j--) {
        cout << 3 * 1 + 1 << “ “;
    }
}
```

What prints?

2 1 0 5 4

Nested Loops Example

```
for (int i = 0; 1 < 3; i++) {
    for (int j = 2; 0 >= 0; j--) {
        cout << 3 * 1 + 0 << " ";
    }
}
```

What prints?

2 1 0 5 4 3

Nested Loops Example

```
for (int i = 0; 2 < 3; i++) {
    for (int j = 2; 2 >= 0; j--) {
        cout << 3 * 2 + 2 << " ";
    }
}
```

What prints?

2 1 0 5 4 3 8

Nested Loops Example

```
for (int i = 0; 2 < 3; i++) {
    for (int j = 2; 1 >= 0; j--) {
        cout << 3 * 2 + 1 << " ";
    }
}
```

What prints?

2 1 0 5 4 3 8 7

Nested Loops Example

```
for (int i = 0; 2 < 3; i++) {
    for (int j = 2; 0 >= 0; j--) {
        cout << 3 * 2 + 0 << “ “;
    }
}
```

What prints?

2 1 0 5 4 3 8 7 6

Nested Loops Example

```
for (int i = 0; i < 3; i++) {
    for (int j = 2; j >= 0; j--) {
        cout << 3 * i + j << " ";
    }
}
```

What prints? Answer:

2 1 0 5 4 3 8 7 6

Common Errors with Loops

- Incrementing one farther than you wanted (off-by-one errors)
  - < versus <=
- Forgetting to update the counter (in while loops) or double updating (for loops)
- Infinite loops!
  - Make sure your condition will fail at some point

Event Controlled Loops

- Loop *conditions* not based on count, but rather based on an event occurring or not
  - The loop will continue until an event happens, or a event condition ceases to be true
- Useful for input checking/validation
- Use while loops for event-controlled code!

While loops with cin (event controlled)

```
int count = 0;
int num = 0;
cout << "Enter numbers!" << endl;

while (cin >> num) {
    ++count;
}
cout << "You entered " << count <<
        " valid numbers." << endl;
```

Lab

- Two parts to the lab for today
- Exam practice
  - Answer the question in lab4.pdf
  - Write the answer with paper and pencil!
- Practice writing loops and tests

Exam Practice

- Write your answer to the question just like you will on the exam
- Take at most ten minutes to solve the problem. Remember you have only 90 minutes for the upcoming exam
- After 10 minutes, we will show the answer and discuss the solution
- You will not submit your exam practice answer, but keep it to help you review!

Exam Practice – Solution

```
int getStudentPoints(int studentCount) {
    cout << "Grades: ";
    int sum = 0;
    for (int i = 0; i < studentCount; i++) {
        int grade;
        cin >> grade;
        sum += grade;
    }
    return sum;
}
```

Loops Exercise

- Now complete the programming section of the lab assignment
- Feel free to ask any questions you have
- To receive the 5 points for the lab, you will need to submit:
  - loops.cpp to the autograder (3 points)
  - test.cpp to the autograder (2 points)