Philip Pesic

Week 3

September 4 2022

Week 3 Lecture 3 Notes

Sequences and Alternative Sequences

A sequence is a set series of commands that a program executes. An alternative sequence is one with branching paths and variation. Alternative sequences are the basis for widely-effective programming. Branching paths with varying outcomes make user interaction with a program possible.

Ex: Multiple varying outcomes must be possible for a program to produce if someone asks for something to be converted to another unit

Criteria and Relational Criteria

Criteria is the measurement by which something is judged. They are conditions that have to be met, or guidelines. Criteria is used in everyday life. Relational criteria is the means by which something is compared to something else. The symbolic denotations of relational criteria are:

>: Greater than

<: Less than

>=: Greater than or equal to

<=: Less than or equal to

==: Equal

!=: Unequal

Philip Pesic

Week 3

September 4 2022

Week 3 Lecture 3 Notes

IF Statement

IF statements are functions that will execute certain code if criteria is met. IF statements are used in every facet of software development, and serve as the basis of alternative sequences and user interaction in computer programming. (The example below is simply a representation, not real code)

Ex: if (var == 1) {print “hello”}; //This means that “hello” will only be printed if var == 1

If/Else Statements

If/Else statements are the basis of double selection, or multiple branching pathways. If/else statements are used to execute certain code if criteria is met, AND different code if it is not. If/else statements are often used when dealing with a wide range of possible user inputs, and expand upon the IF statements basis for alternative sequence programming. (The example below is simply a representation, not real code)

Ex: if (var == 1) {print “Hello”}; Else {print “Bye”};

//This prints “Hello” if var == 1, and “Bye” if not

Philip Pesic

Week 3

September 4 2022

Week 3 Lecture 3 Notes

Cascading and Nested If/Else statements

Cascading and nested If/Else statements are combinations of multiple if/else statements used to create even more branching paths. Cascading statements build upon criteria and will cascade downward into a new condition until criteria is met. Nested If/Else statements are If/Else statements within one another. These are used to test a combination of criteria before producing an output. (The example below is simply a representation, not real code)

Ex: if (var > 1) {if (var < 2) {varValue = 4}} else {varValue = 5};

/*Here, varValue will only equal 4 if var is greater than 1 but less than 2. Otherwise varValue will equal 5*/

Loop

A loop, or repeating sequence, is used to save time and provide an easier to read syntax when programming. Each loop has 3 parts: A variable, a condition, and a modification. Each of these must be implemented somewhere above or in the loop in order for it to properly function.

Ex: for (int i = 1; i < 5; i++) {cout << i;}

Types of loops

Two types of loops: Pretest and Posttest

Pretest: Conditions are checked every time before the loop is run

Posttest: Conditions are checked every time after the loop is run

Philip Pesic

Week 3

September 4 2022

Week 3 Lecture 3 Notes

For loop

For loops are pretest loops. Their syntax looks like this:

For (variable; condition; modification) {code}

Loop

A repeating sequence

Ex: for (int i = 1; i < 10; i++) {cout << i;}

Types of loops

Pretest: Condition is checked before each loop

Posttest: Condition is checked after each loop

While loop

Pretest loop with the following syntax: while (condition) {output; modifier;}

Ex: int i = 1; while (i < 5) {i++; cout << i;}

Nested while loop

Works the same way as a nested for loop, with the same syntax

Ex: while (condition) {modifier; while (condition) {modifier; output;}}

Philip Pesic

Week 3

September 4 2022

Week 3 Lecture 3 Notes

Do While loop

Posttest loop. Completes an iteration, then checks the condition.

Ex: do {output; modifier;} while (condition);

Functions

Functions are a form of non-contiguous code reuse. A set of commands is tied to a statement, which can be executed as a placeholder for the set.

Ex: void func() {cout << “Hello” << endl;}

Prewritten and Programmer written functions

Prewritten functions are ones provided by a library, such as cmath, with a set of commands built in. Programmer written functions are completely written by the programmer in order to serve any purpose.

Ex: pow(2, 4);

The four functions:

void/noValue

void/value

return/noValue

return/value