COMP 2210

January 13th, 2022

Tyler Teufel

* Following COMP 2210, COMP 2710 (Software Construction) should be taken, COMP 3270 (introduction to Algorithms) should be taken, and COMP 3240 (Discrete Math) should be taken prior to the remainder of the curriculum.
* Data Structures (Methods of storing data) + Algorithms (Methods of solving problems) = Programs

Example Problem: Patterm recognition

Given a set of two-dimensional points as input, highlight all the points that participate in a specified pattern (e.g., collinear). Assume you have a machine that executes 1 billion instructions per second.

Program input (~100k points)

COMP 2210

January 18th, 2022

Tyler Teufel

* Today’s Topics: Problem solving, solution strategies, patterns
* Before next time:
  + Correctness and Testing – videos and notes
  + M1: Hands-On: Linear Scan
  + M1: Hands-On: Testing

Problem-solving / Programming Approach

1. Develop a solution strategy
2. Develop an algorithm that implements that strategy.
3. Develop a program that implements that algorithm

Solution Patterns

* Solution patterns repeat
* A first solution pattern: Linear Scan
* Linear Scan is the fundamental solution strategy for many problems. It only has to be tailored to the specific task and expressed in an appropriate form.

for (SomeType value : Group) {

Process(value);

}

for (int I = 0; …) {

process(value);

}

While (cond) {

Process(value);

}

do {

process(value);

} while (cond)

Array Search

* Given an array of ints, return true if a given value is present and false if not.
* For each item that we have { process the current item }

Pair Sum

* Does the given array have a pair of nums that some to x? You can assume the array elements are in ascending order

How to approach developing in A1: For each method in the Selector class:

1. Develop a strategy based upon Linear Scan. (You should be able to explain this strategy to your roommate who isn’t a CS major.)
2. Write informal t

COMP 2210

January 20th, 2022

Tyler Teufel

Testing – intent should be to break your code, or to make it fail.

* Test cases should be focused and precise so that a failure gives you actionable information.
* Tests should be repeatable
* Tests should be separate from the code under development.
* Tests can be (and maybe should be?) written before the code under development.

Levels of testing

* Unit: very small scale
* Integration
* System
* Acceptance

Black Box Testing

* Only the external behavior and characteristics of the item under test are used to design test cases.
* Internal structure of the item under test is not known/used.
* Tests only the what, not the how.
* Also known as behavioral testing or specification testing.