CPSC 2120/2121
Lab 07

Numerical Integration and Fibonacci
Due date/time are on the CPSC 2121 Canvas website

Learning Objectives

- To improve our ability to implement algorithms and data structures in C++
- To implement dynamic programming in C++
- To implement an iterative mathematical solver in C++
- To use simple data structures to implement algorithms
- To become proficient in fundamental data structures used throughout computer science
- To improve our analytical skills while gaining familiarity with mathematical tools used in the analysis of algorithms
- To implement an algorithm and understand its intricacies

Problem/Exercise

For this lab, you will complete two independent programs. For the first, you will implement functions which calculate the n^{th} Fibonacci number. You must write two functions: one which simply uses recursion, and one which makes use of dynamic programming/memoization. Run both programs with the provided lab07.cpp file and submit a written report which includes your pseudocode, the runtimes for each, and a **brief** discussion of the difference in runtimes.

The second assignment is to perform numerical integration. For this assignment, there will not be any user interface or reading in. You will be given a vector of ordered pairs (coefficient, power), and two doubles for upper and lower bounds of integration. From this information you must construct a polynomial function and numerically integrate it between the bounds given using the Midpoint Rule. Your program should iteratively improve its estimate until the difference between iterates is less than 0.0001.

For both parts of this assignment, you are allowed to use any C++ STL data structures. Your final code will be in two files, memoization.h and integration.h. Both files must compile without errors using the provided Makefile. In order to thoroughly test your program, you will also submit two input files named lab07_memo.txt and lab07_int.txt which have test cases for each program respectively.

Additional Requirements

- Your memoization.h and integration.h files must compile and run with unmodified versions of the provided lab07.txt and Makefile to produce the same output as our examples. If your class does not, then you'll need to fix it. After you get your class working, you should create several examples on your own to further test your program to make sure it works with any valid set of test cases.
- As a hint, you should run your fibonacci program on a range of values (e.g. 5, 10, 30, 100) to see how the runtime changes between memoized and normal recursive versions.
- For numerical integration, use the midpoint rule with initial step size = 1. At each step reduce the step size by half (1 to ½, ¼, etc.)
- Stop when the difference between the current and previous iteration is less than 0.0001.

Examples

Your C++ source code should be in a file called memoization.h and integration.h, and it should be compiled into an executable called lab07.out using our provided Makefile. The output of memoization.h and integration.h must look exactly like the examples below when run on the command line on our Unix machines.

```
./lab07.out
x^2+1 0 1
1.3333

x^4-x^3+x^2-1 -3 4
232.983

x^0.5 0 .5
0.2357

fib(5)
5

fib(1)
1

fib(10)
55
```

Source Code Requirements

- Put a comment at the top of your source code file with your name (first and last), the date of your submission, your lab section, and the assignment's name.
- All functions should be commented. Use inline documentation as needed to explain ambiguous, tricky, or important parts of your code.
- All source code must follow good programming style standards such as properly indenting source code; and all variables, functions, and classes should be well-named.
- Your class must use dynamic memory allocation and deallocation properly, which means your class cannot contain a memory leak. You may use valgrind to check for memory leaks. Refer to the Unix manual and valgrind's online documentation for more information on valgrind.
- Your integration program must take input in the form of ordered pairs. The first number is the coefficient, the second is the exponent.

Submission

Before the date/time stated on the CPSC 2121 Canvas webpage, you need to submit your code to handin under the correct lab assignment. Make sure to submit all of the following.

1. All source files required for this lab (memoization.h and integration.h)
2. Your testing file (lab07_memo.txt and lab07_int.txt)

After you submit, always double check that the file(s) you submitted were the correct version. To double check, download the submitted file(s), put them on one of our Unix machines, and make sure they compile and run correctly.

Grading

If your class does not compile on our Unix machines or your assignment was not submitted on time, you will receive a grade of 0 on this assignment. Otherwise, your class will be graded using the criteria below.

Your class works correctly with our testing program(s)	6 points
Proper variable names, documentation, and code organization	2 points
Your class uses dynamic memory allocation/deallocation properly (no memory leaks)	1 point
Alternate testing file containing map and query data you created for testing which is distinct from that provided.	1 point

You must test, test, and retest your code to make sure it compiles and runs correctly and efficiently on our Unix machines with any given set of valid inputs. This means that you need to create many examples on your own (that are different from the provided lab07_memo.txt and lab07_int.txt) to ensure you have a correctly working class. We will only test your program with valid sets of inputs.