CS 202 Iditarod Challenge: Anchorage

Millard A. Arnold V

March 27, 2020

- Repository Link: `https://github.com/imthegngrbrdmn/cs-202/tree/master/iditarod-anchorage`
- Project Link: `https://github.com/imthegngrbrdmn/cs-202/projects/1`
- Git Commits: `https://github.com/imthegngrbrdmn/cs-202/commits/master`
- This homework took approximately 04 hours to complete.

1 Design

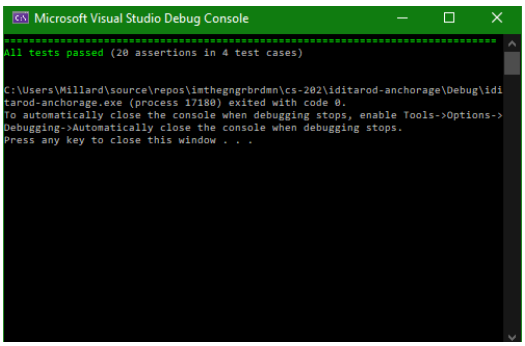
The design I took for this program is simple. I wrote the basic functionality for each program file with the loop first, and recursion second. I used Catch2 to test each function after writing it. I wrote the test cases in order to ensure that each function is working well, and I contained all of those in the test file.

2 Post Mortem

Using the GitHub Project cards was actually really useful and allowed me to organise the program beforehand so that the actual program writing phase went very quickly.

3 Program 1

3.1 Sample Output Screenshot



3.2 Git Commit Messages

Date	Message
2020-03-26	Set Up Project In Visual Studio
2020-03-26	Integrate Catch2 Library
2020-03-26	Integrate Catch2 Library
2020-03-26	write factorial_loop(n) using a loop to find factorials
2020-03-26	Use Catch2 to test factorial_loop(n)
2020-03-26	write factorial(n) using recursion
2020-03-26	Use Catch2 to test factorial(n)
2020-03-26	write fib_loop(n) using a loop to find Fibonacci sequence numbers
2020-03-26	Use Catch2 to test fib_loop(n)
2020-03-26	Write fib(n) through recursion
2020-03-26	Use Catch2 to test fib(n)

3.3 Source Code

3.4 Factorial

Header

```c
#ifndef FACTORIAL_H
#define FACTORIAL_H

int factorial_loop(int n);
int factorial(int n);

#endif
```

Source

3.5 Factorial Source

```c
int factorial_loop(int n)
{
  int result = 1;
  while (n>0)
  {
    result *= n;
    n--;
  }
  return result;
}

int factorial(int n)
{
  if (n > 0)
  {
    return n * factorial(n - 1);
  }
  else if (n == 0)
  {
    return 1;
  }
}
```

3.6 Fibonacci

Header

```c
#ifndef FIB_H
#define FIB_H

int fib_loop(int n);
int fib(int n);

#endif
```

Source

```
#include "fib.h"

int fib_loop(int n)
{
  int n2 = 1;
  int n1 = 0;
  int sum = 0;
  if (n == 0)
  {
    return 0;
  }
  if (n == 1)
  {
    return 1;
  }
  for (int i = 2; i <= n; i++)
  {
    sum = n1 + n2;
    n1 = n2;
    n2 = sum;
  }
  return sum;
}

int fib(int n)
{
  if (n == 0)
  {
    return 0;
  }
  else if (n == 1)
  {
    return 1;
  }
  else if (n > 1)
  {
    return fib(n - 1) + fib(n - 2);
  }
}
```

3.7 Test File

```
#define CATCH_CONFIG_MAIN
#include "../catch.hpp"
#include "fib.h"
#include "factorial.h"

TEST_CASE("Factorials are computed through a loop", "[factorial_loop]")
{
  REQUIRE(factorial_loop(0) == 1);
  REQUIRE(factorial_loop(1) == 1);
  REQUIRE(factorial_loop(5) == 120);
  REQUIRE(factorial_loop(9) == 362880);
}

TEST_CASE("Factorial are computed through recursion", "[factorial]")
```

```
{
  REQUIRE(factorial(0) == 1);
  REQUIRE(factorial(1) == 1);
  REQUIRE(factorial(5) == 120);
  REQUIRE(factorial(9) == 362880);
}

TEST_CASE("Fibonacci sequence numbers are computed through a loop", "[fib_loop}")
{
  REQUIRE(fib_loop(0) == 0);
  REQUIRE(fib_loop(1) == 1);
  REQUIRE(fib_loop(2) == 1);
  REQUIRE(fib_loop(3) == 2);
  REQUIRE(fib_loop(7) == 13);
  REQUIRE(fib_loop(30) == 832040);
}

TEST_CASE("Fibonacci sequence numbers are computed through recurstion", "[fib]")
{
  REQUIRE(fib(0) == 0);
  REQUIRE(fib(1) == 1);
  REQUIRE(fib(2) == 1);
  REQUIRE(fib(3) == 2);
  REQUIRE(fib(7) == 13);
  REQUIRE(fib(30) == 832040);
}
```
