CS 202 Iditarod Challenge 1

Kelby Hubbard

March 24, 2020

- Repository Link: `https://github.com/krhubbard2/CS202/tree/master/Iditarod1`
- Git Commits: `https://github.com/krhubbard2/CS202/commits`
- This homework took approximately 1 hours to complete.

1 Design

The design I took for this program was to follow Catch2 layout. By implementing Catch2 I was able to test my functions as I wrote them allowing me to quickly know what is going wrong and what is going right. It allowed for easier tracking and monitoring of my functions and their outputs.

2 Post Mortem

The program wrote very easily after implementing Catch2. The biggest "catch" was realizing how to write the two functions differently– one using recursion and one without. After figuring out the "math" behind the functions it was fairly straight forward and Catch2 allowed me to find my issues and fix them quickly.

3 Recursion Problems

3.1 Sample Output

```
===============================================
    All tests passed (20 assertions in 1 test case)
```

3.2 Git Commit Messages

Date	Message
2020-03-24	Write fib(n)s
2020-03-24	Write fibloop(n)n
2020-03-24	Write factorial(n).
2020-03-24	Write factorialloop(n)

3.3 Source Code

```
1  // Kelby Hubbard
2  // CS202
3  // March 24, 2020
4  // Iditarod Challenge 1
5
6  #define CATCH_CONFIG_MAIN
7  #include <catch2/catch.hpp>
8
9  //Design
10 // [X] fib(n)
11 // [X] fib_loop(n)
12 // [X] factorial(n)
13 // [X] factorial_loop(n)
14
15 //Fibonacci with recursion
16 unsigned int fib(unsigned int n)
17  {
18    if (n <= 1)
19     {
20       return n;
21     }
22   return fib(n-1) + fib(n-2);
23 }
24
25 //Fibonacci without recursion
26 int fib_loop(int n)
27 {
28   if (n <= 1)
29   {
30     return n;
31   }
32   int a = 1;
33   int b = 1;
```

```
34    for (int i = 2; i < n; ++i)
35    {
36      int temp = a;
37      a += b;
38      b = temp;
39    }
40    return a;
41  }
42
43  //Factorial with recursion
44  int factorial(int n)
45  {
46    if (n > 1)
47    {
48      return n * factorial (n-1);
49    }
50    else
51    {
52      return 1;
53    }
54
55  }
56
57  //Factorial non recursion
58  int factorial_loop(int n)
59  {
60    if (n > 1)
61    {
62      int a = 1, i;
63      for (i = 1; i <= n; i++)
64      {
65        a = a * i;
66      }
67      return a;
68    }
69    else
70    {
71      return 1;
72    }
73  }
74
75  TEST_CASE( "Fibonacci Sequence", "[fibonacci]" )
76  {
77    SECTION("FIBONACCI RECURSIVE")
78    {
79      REQUIRE( fib(0) == 0 );
80      REQUIRE( fib(1) == 1 );
81      REQUIRE( fib(9) == 34 );
82      REQUIRE( fib(14) == 377 );
83    }
84
85    SECTION ("FIBONACCI NON RECURSIVE")
86    {
87      REQUIRE( fib_loop(0) == 0);
88      REQUIRE( fib_loop(1) == 1 );
89      REQUIRE( fib_loop(9) == 34 );
90      REQUIRE( fib_loop(14) == 377 );
91    }
92
93    SECTION ("FACTORIAL RECURSIVE")
94    {
95      REQUIRE (factorial(0) == 1);
```

```
96      REQUIRE (factorial(1) == 1);
97      REQUIRE (factorial(2) == 2);
98      REQUIRE (factorial(4) == 24);
99      REQUIRE (factorial(6) == 720);
100     REQUIRE (factorial(12) == 479001600);
101   }
102
103   SECTION ("FACTORIAL NON REDCURSIVE")
104   {
105     REQUIRE (factorial_loop(0) == 1);
106     REQUIRE (factorial_loop(1) == 1);
107     REQUIRE (factorial_loop(2) == 2);
108     REQUIRE (factorial_loop(4) == 24);
109     REQUIRE (factorial_loop(6) == 720);
110     REQUIRE (factorial_loop(12) == 479001600);
111   }
112 }
```
