CS 202 Iditarod - Rainy Pass

Bryan Beus

April 7, 2020

Source Code Link: `https://github.com/siddhartha-crypto/cs202/tree/master/iditarod/rainy-pass`

1 Design

1.1 Ackerman's Function

The design provided in the homework assignment is sufficient. Ackerman's function is already provided. As with the Anchorage assignment, I will keep this simple.

1.2 Part 2 of Ackerman's Function

I intend to output the values to a local file and display the values using either LibreOffice or any easily availble online resource via Google.

2 Post Mortem

2.1 Ackerman's Function

The Wikipedia article was not very helpful for understanding Ackerman's function, beyond basic concepts. But that was okay. I was able to get the basic idea and quickly create a logic mockup.

2.2 Part 2 of Ackerman's Function

The only difficult thing here was to create a set of `m,n` values that maxed out each respective value. This took about 20 minutes of trial and error.

In retrospective, I think if I had used `try, catch` statements, instead of hard-coding in the `m,n` values, I would have saved myself an hour.

3 Commit History

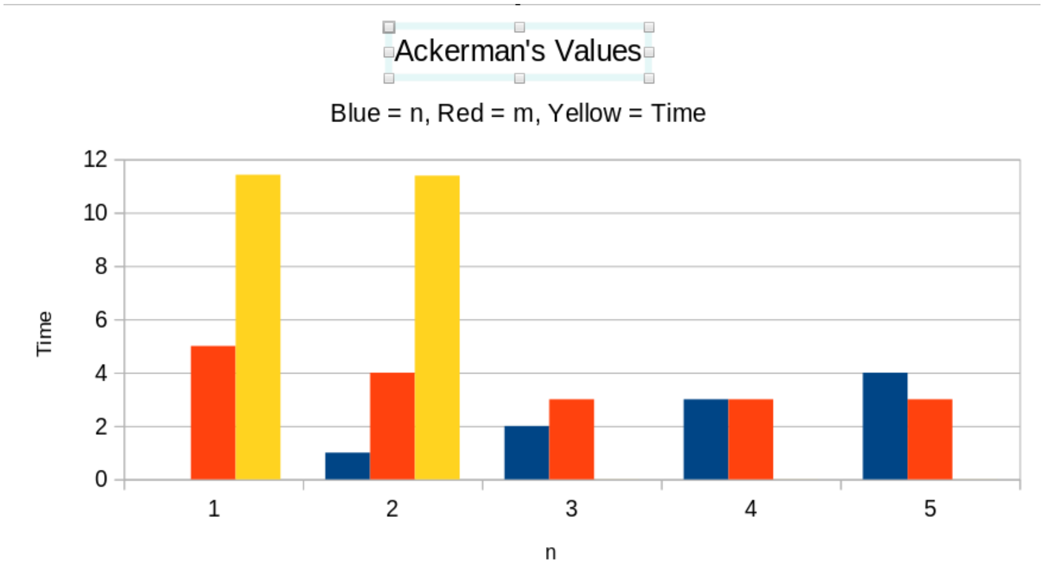
3.1 Both Parts 1 2

2020-04-07 Write an initial pass for Ackerman's Function
2020-04-07 Create iterative process for testing Ackerman's values
2020-04-07 Capture individual StopWatch times
2020-04-07 set up switch statement to measure each value
2020-04-07 Calculate largest Ackerman's value my machine can compute

4 Sample Output

4.1 Ackerman's Output Values (to values.txt)

```
0, 0, 1, 2.08e-07,
1, 0, 2, 1.85e-07,
2, 0, 3, 3.09e-07,
3, 0, 5, 6.74e-07,
4, 0, 13, 2.024e-06,
5, 0, 65533, 11.4208,
0, 1, 2, 5.5e-08,
1, 1, 3, 1e-07,
2, 1, 5, 1.83e-07,
3, 1, 13, 5.79e-07,
4, 1, 65533, 11.3859,
0, 2, 3, 5.8e-08,
1, 2, 4, 1.13e-07,
2, 2, 7, 2.43e-07,
3, 2, 29, 2.254e-06,
0, 3, 4, 4.6e-08,
1, 3, 5, 8.6e-08,
2, 3, 9, 2e-07,
3, 3, 61, 8.83e-06,
0, 4, 5, 4e-08,
1, 4, 6, 9e-08,
2, 4, 11, 2.95e-07,
3, 4, 125, 3.5402e-05,
```



5 My Programs

5.1 Main.cpp (Includes Both)

```
/*
 * main.cpp
 * CS202
 * April 7, 2020
 * Bryan Beus
 * Rainy Pass station for Iditarod Challenge
 */

#include <iomanip>
#include <vector>
#include <string>
#include <iostream>
#include <fstream>

#include "StopWatch.hpp"

using std::cout;
using std::cin;
using std::endl;
using std::string;
using std::vector;
using std::ofstream;

```

```cpp
// Calculate Ackerman's value
long long int ack(const long long int& m, const long long int& n) {
    if (m == 0) {
        return n + 1;
    } else if (m > 0 && n == 0) {
        return ack(m - 1, 1);
    } else if (m > 0 && n > 0) {
        return ack(m - 1, ack(m, n - 1));
    } else {
        return 0;
    }
}

int main() {

    // Save output to local file
    ofstream fout("./values.txt”);

    if (!fout) {
        cout << "Error loading output file” << endl;
    }

    // Initiate StopWatch instance
    StopWatch totalTime;

    long long int val;

    cout << "Calculating Ackerman's Values” << endl;
    int max = 0;

    // Iterations for n
    for (long long int n = 0; n < 5; n++) {

        // Set highest level for m at each level n
        switch (n) {
            case 0:
                max = 5;
                break;
            case 1:
                max = 4;
                break;
            case 2:
                max = 3;
                break;
            case 3:
                max = 3;
                break;
            case 4:
                max = 3;
                break;
            default:
                break;
        }

        // Iteration level for m
        for (long long int m = 0; m <= max; m++) {

            cout << endl;
            cout << "m: " << m << endl;
            cout << "n: " << n << endl;
            StopWatch currSw;
            val = ack(m, n);
            currSw.captureFinishTime();
            cout << "v: " << val << endl;
            cout << "t: " << currSw.reportFinishTime() << endl;
            fout << m << ", " << n << ", " << val << ", " << currSw.reportFinishTime() << "," << endl;
            cout << endl;
            cout << endl;
        }
    }

    cout << "Final Time: ";
    totalTime.captureFinishTime();
    cout << totalTime.reportFinishTime() << endl << endl;


    return 0;
}
```