Final Project for CS 372

***Erase this*** *before you submit: You are more than welcome to bullet point or tabulate everything like in the first heading and built-in tests. I’m going to be “checking” off what I can from the report, and the easier it is to find, the less likely a requirement will be missed. Team only parts are in green.*

*[Your name]*

# Algorithm, Application, Language Choice

* *[Your algorithm choice]*
* *[Your application choice]*
* *[Your language choice]*

# Where It Is Used

*[Brief description of the class of problems your algorithm(s) solves]*

## Other applications

* *[at least 2 other possible applications for your algorithm]*

## Alternative algorithms

* *[and at least 2 other possible algorithms for your application]*

*[ Compare and contrast the algorithms]*

## Reason for choice

*[explain why you picked the algorithm you did]*

*[1/2 page is typically sufficient for this section]*

# How Your Project Works

*[How it works. Pretend you are explaining how your project works to a Programming II freshman. 1-3 paragraphs will likely be sufficient, but ask if you are unsure This is mostly to make sure you understand what you did and you didn’t just copy and paste code from somewhere.]*

*(team projects are expected to have more detail. You must “chunk” the algorithm and explain each part.)*

## Correctness (team only)

*[Formally, justify why your algorithm is correct or why it works. You must formally use loop invariant, pre-and post conditions, etc. Any and all properties for a given algorithm, if applicable, are required. Assume you are speaking to someone who has completed this course]*

# Run time

*[Name and explain why the project has this big-O run time as the theoretical run time* **for your implementation***]*

*[A* ***fully*** *labeled graph* runtime graph for varying n with a minimum of 10 points goes here. It **must clearly show** the run time. If you have more than 1 value that affects input, you may just vary one for the graph, but be clear which one you used. Two graphs would be ideal, and you may use a minimum of 5 points for each in this case)

(team only): Formally prove the run time through instruction counting, probability, or recursion analysis depending on your problem. You **will need** to use psudeocode to prove this.

# Built-in Code Correctness Tests

(team only, you must have a minimum of 6 rather than 3 built-in tests)

|  |  |  |  |
| --- | --- | --- | --- |
| ***Test Case*** | ***Description*** | ***Input*** | ***Actual output*** |
| *[test 1]* |  |  |  |
| *[test 2]* |  |  |  |
| … |  |  |  |

Alternatively, you may have this in the following format:

## Test 1; name

Description

### Input

The input

### Output

The output

# Program usage or README

[OPTIONAL, and only used if needed]

# References

[Where did you find the explanation of your code]

[Tutorials]

[Etc.]