Jennifer Mince, Quality Assurance

**Project 2 Algorithm Test Procedures**

**Brute Force/General Checks**

**Test Number:** TTT-01

**Author:** Jennifer Mince

**Date:** 4/8/19

**Revision:** 1.1

**Test Purpose:** This test is intended to check if the Brute Force Algorithm runs to completion.

**Test Procedures:**

1. Run the file
2. Use the prompt to input one file
3. Repeat to input the second file
4. Click go

**Measure of Success:**

That the code runs without errors and gives output based off the sequences provided.

**Test Number:** TTT-02

**Author:** Jennifer Mince

**Date:** 4/8/19

**Revision:** 1.1

**Test Purpose:** This is intended to test if the input files are parsed and handled correctly by the program.

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file
3. Repeat to input the second file
4. Click go

**Measure of Success:**

The program runs to completion without errors and uses the correct variables to come up with its output. Input taken in by GUI, parsed and stored, backend code is able to use parsed variables to send an output back to GUI.

**Test Number:** TTT-03

**Author:** Jennifer Mince

**Date:** 4/10/19

**Revision:** 1.1

**Test Purpose:** This is intended to test if the input files are parsed and handled correctly by the program.

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off the edge case input small file 1
3. Repeat to input the second file based off the edge case input small file 2
4. Click go

**Measure of Success:**

The program runs with small sequence input files.

**Test Number:** TTT-04 (input handling)

**Author:** Jennifer Mince

**Date:** 4/24/2019

**Revision:** 1.1

**Test Purpose:** Is input file formatted correctly? No strange characters or letters inside?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off the edge case input small file 1
3. Repeat to input the second file based off the edge case input small file 2
4. Click go

**Measure of Success:**

The program runs with input files and checks for anything that should not be in them.

**Test Number:** TTT-05

**Author:** Jennifer Mince

**Date:** 4/24/2019

**Revision:** 1.1

**Test Purpose:** Does the code run with larger sequence, such as the example file from project 1?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off the edge case input large file 1
3. Repeat to input the second file based off the edge case input large file 2
4. Click go

**Measure of Success:**

The program runs with input files and comes up with a compatible score, as well as does not time out.

**Test Number:** TTT-06

**Author:** Jennifer Mince

**Date:** 4/8/19

**Revision:** 1.1

**Test Purpose:** Does the code run and output the sequence, second sequence aligned to the first, and the score?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file
3. Repeat to input the second file
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations.

**Test Number:** TTT-07 (runtime of brute force)

**Author:** Jennifer Mince

**Date:** 4/24/19

**Revision:** 1.1

**Test Purpose:** Discover the runtime of the Brute Force algorithm

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file
3. Repeat to input the second file
4. Click go

**Measure of Success:**

The program runs and outputs a timestamp of how long the program took to run and the group will evaluate the algorithm for its runtime.

**Test Number:** TTT-08

**Author:** Jennifer Mince

**Date:** 4/8/19

**Revision:** 1.1

**Test Purpose:** Does the code run correctly with different lengths of input?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off different lengths example
3. Repeat to input the second file off different lengths example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations.

**Test Number:** TTT-09 (valid character check)

**Author:** Jennifer Mince

**Date:**

**Revision:** 1.1

**Test Purpose:** Does the code run correctly with a valid, but different character than A, G, C, or T?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off valid character example
3. Repeat to input the second file off valid character example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations. Handles character issues.

**Test Number:** TTT-010 (invalid character check)

**Author:** Jennifer Mince

**Date:** 4/24/2019

**Revision:** 1.1

**Test Purpose:** Does the code run correctly with an invalid character?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off invalid character example
3. Repeat to input the second file off invalid character example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations. Handles character issues.

**Test Number:** TTT-011

**Author:** Jennifer Mince

**Date:** 4/10/19

**Revision:** 1.1

**Test Purpose:** Do the GUI and backend connect correctly?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file
3. Repeat to input the second file
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations.

**Test Number:** TTT-012

**Author:** Jennifer Mince

**Date:** 4/1/19

**Revision:** 1.1

**Test Purpose:** This test is a check to make sure the GUI only allows txt files to be inputted into the backend code.

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input a file that is other than a txt file
3. Repeat to input the second file
4. Click go

**Measure of Success:**

The program gives an error message and does not proceed if anything other than a txt file is inputted.

**Test Number:** TTT-013 (perfect match input)

**Author:** Jennifer Mince

**Date:** 4/1/19

**Revision:** 1.1

**Test Purpose:** Does the code run correctly when both inputs are exactly the same?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off exact match character example
3. Repeat to input the second file off exact match character example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations. Score should be as expected, 1 point per each matched character.

**Test Number:** TTT-013.2 (completely mismatch input)

**Author:** Jennifer Mince

**Date:** 4/24/19

**Revision:** 1.1

**Test Purpose:** Does the code run correctly when both inputs are completely different?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off mismatch character example
3. Repeat to input the second file off mismatch character example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations. Score should be as expected, 1 point per each matched character.

**Needleman-Wunsch Algorithm Tests**

**Test Number:** TTT-014

**Author:** Jennifer Mince

**Date:** 4/8/19

**Revision:** 1.1

**Test Purpose:** This test is intended to check if the Needleman-Wunsch Algorithm runs to completion.

**Test Procedures:**

1. Run the file
2. Use the prompt to input one file
3. Repeat to input the second file
4. Click go

**Measure of Success:**

That the code runs without errors and gives output based off the sequences provided.

**Test Number:** TTT-015

**Author:** Jennifer Mince

**Date:** 4/24/2019

**Revision:** 1.1

**Test Purpose:** Does the code run with larger sequence, such as the example file from project 1?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off the edge case input large file 1
3. Repeat to input the second file based off the edge case input large file 2
4. Click go

**Measure of Success:**

The program runs with input files and comes up with a compatible score, as well as does not time out.

**Test Number:** TTT-016 (runtime of Needleman-Wunsch)

**Author:** Jennifer Mince

**Date:** 4/24/19

**Revision:** 1.1

**Test Purpose:** Discover the runtime of the Needleman-Wunsch algorithm

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file
3. Repeat to input the second file
4. Click go

**Measure of Success:**

The program runs and outputs a timestamp of how long the program took to run and the group will evaluate the algorithm for its runtime.

**Test Number:** TTT-017

**Author:** Jennifer Mince

**Date:** 4/8/19

**Revision:** 1.1

**Test Purpose:** Does the code run correctly with different lengths of input?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off different lengths example
3. Repeat to input the second file off different lengths example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations.

**Test Number:** TTT-018 (perfect match input)

**Author:** Jennifer Mince

**Date:** 4/1/19

**Revision:** 1.1

**Test Purpose:** Does the code run correctly when both inputs are exactly the same?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off exact match character example
3. Repeat to input the second file off exact match character example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations. Score should be as expected, 1 point per each matched character.

**Test Number:** TTT-019

**Author:** Jennifer Mince

**Date:** 4/10/19

**Revision:** 1.1

**Test Purpose:** This is intended to test if the input files are parsed and handled correctly by the program.

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off the edge case input small file 1
3. Repeat to input the second file based off the edge case input small file 2
4. Click go

**Measure of Success:**

The program runs with small sequence input files.

**Test Number:** TTT-013.2 (completely mismatch input)

**Author:** Jennifer Mince

**Date:** 4/24/19

**Revision:** 1.1

**Test Purpose:** Does the code run correctly when both inputs are completely different?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off mismatch character example
3. Repeat to input the second file off mismatch character example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations. Score should be as expected, 1 point per each matched character.

**Greedy Algorithm Tests**

**Test Number:** TTT-020

**Author:** Jennifer Mince

**Date:** 4/8/19

**Revision:** 1.1

**Test Purpose:** This test is intended to check if the Greedy Algorithm runs to completion.

**Test Procedures:**

1. Run the file
2. Use the prompt to input one file
3. Repeat to input the second file
4. Click go

**Measure of Success:**

That the code runs without errors and gives output based off the sequences provided.

**Test Number:** TTT-021

**Author:** Jennifer Mince

**Date:** 4/24/2019

**Revision:** 1.1

**Test Purpose:** Does the code run with larger sequence, such as the example file from project 1?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off the edge case input large file 1
3. Repeat to input the second file based off the edge case input large file 2
4. Click go

**Measure of Success:**

The program runs with input files and comes up with a compatible score, as well as does not time out.

**Test Number:** TTT-022 (runtime of greedy)

**Author:** Jennifer Mince

**Date:** 4/24/19

**Revision:** 1.1

**Test Purpose:** Discover the runtime of the Greedy algorithm

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file
3. Repeat to input the second file
4. Click go

**Measure of Success:**

The program runs and outputs a timestamp of how long the program took to run and the group will evaluate the algorithm for its runtime.

**Test Number:** TTT-023

**Author:** Jennifer Mince

**Date:** 4/8/19

**Revision:** 1.1

**Test Purpose:** Does the code run correctly with different lengths of input?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off different lengths example
3. Repeat to input the second file off different lengths example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations.

**Test Number:** TTT-024 (perfect match input)

**Author:** Jennifer Mince

**Date:** 4/1/19

**Revision:** 1.1

**Test Purpose:** Does the code run correctly when both inputs are exactly the same?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off exact match character example
3. Repeat to input the second file off exact match character example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations. Score should be as expected, 1 point per each matched character.

**Test Number:** TTT-025

**Author:** Jennifer Mince

**Date:** 4/10/19

**Revision:** 1.1

**Test Purpose:** This is intended to test if the input files are parsed and handled correctly by the program.

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off the edge case input small file 1
3. Repeat to input the second file based off the edge case input small file 2
4. Click go

**Measure of Success:**

The program runs with small sequence input files.

**Test Number:** TTT-013.2 (completely mismatch input)

**Author:** Jennifer Mince

**Date:** 4/24/19

**Revision:** 1.1

**Test Purpose:** Does the code run correctly when both inputs are completely different?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off mismatch character example
3. Repeat to input the second file off mismatch character example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations. Score should be as expected, 1 point per each matched character.

**Random Algorithm Tests**

**Test Number:** TTT-026

**Author:** Jennifer Mince

**Date:** 4/8/19

**Revision:** 1.1

**Test Purpose:** This test is intended to check if the Random Algorithm runs to completion.

**Test Procedures:**

1. Run the file
2. Use the prompt to input one file
3. Repeat to input the second file
4. Click go

**Measure of Success:**

That the code runs without errors and gives output based off the sequences provided.

**Test Number:** TTT-027

**Author:** Jennifer Mince

**Date:** 4/24/2019

**Revision:** 1.1

**Test Purpose:** Does the code run with larger sequence, such as the example file from project 1?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off the edge case input large file 1
3. Repeat to input the second file based off the edge case input large file 2
4. Click go

**Measure of Success:**

The program runs with input files and comes up with a compatible score, as well as does not time out.

**Test Number:** TTT-028 (runtime of random)

**Author:** Jennifer Mince

**Date:** 4/24/19

**Revision:** 1.1

**Test Purpose:** Discover the runtime of the Random algorithm

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file
3. Repeat to input the second file
4. Click go

**Measure of Success:**

The program runs and outputs a timestamp of how long the program took to run and the group will evaluate the algorithm for its runtime.

**Test Number:** TTT-029

**Author:** Jennifer Mince

**Date:** 4/8/19

**Revision:** 1.1

**Test Purpose:** Does the code run correctly with different lengths of input?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off different lengths example
3. Repeat to input the second file off different lengths example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations.

**Test Number:** TTT-030 (perfect match input)

**Author:** Jennifer Mince

**Date:** 4/1/19

**Revision:** 1.1

**Test Purpose:** Does the code run correctly when both inputs are exactly the same?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off exact match character example
3. Repeat to input the second file off exact match character example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations. Score should be as expected, 1 point per each matched character.

**Test Number:** TTT-031

**Author:** Jennifer Mince

**Date:** 4/10/19

**Revision:** 1.1

**Test Purpose:** This is intended to test if the input files are parsed and handled correctly by the program.

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off the edge case input small file 1
3. Repeat to input the second file based off the edge case input small file 2
4. Click go

**Measure of Success:**

The program runs with small sequence input files.

**Test Number:** TTT-013.2 (completely mismatch input)

**Author:** Jennifer Mince

**Date:** 4/24/19

**Revision:** 1.1

**Test Purpose:** Does the code run correctly when both inputs are completely different?

**Test Procedures:**

1. Run the GUI file
2. Use the prompt to input one file based off mismatch character example
3. Repeat to input the second file off mismatch character example
4. Click go

**Measure of Success:**

The program runs and outputs correct output based off developer and researcher’s expectations. Score should be as expected, 1 point per each matched character.