Testing Feedback

1.) How helpful were the test suites in guiding development?

The Verification test suites were very helpful in terms of helping me figure out what needed to be done with the File Input/Output methods. They clearly laid out what each file should contain in them and what they should return. The only thing I noticed is that there wasn't any sort of test cases related to the encryption handling in the Verification suites. I feel like there could have been some included, like maybe one to verify that the String inputted for Caesar encryption is a valid value.

The Functionality test suites had a pretty good combination of test cases to check both the file input/output methods, as well as the important encryption methods. The comments were generally helpful in telling me what the test case was testing for and the format for the file input/output methods was helpful.

2.) Did the test suites clearly define and build on the project?

Yes, but there were a few minor things that I had issues with that weren't so clearly defined and required some research. I felt as though all of the test cases relating to the File methods were clearly defined and laid out well in the comments for me to understand. The only confusion I had was pertaining to the amount of data read in, there was no clear explanation of what to do if the file had more than 1 line, hence it wouldn't fit into the array of size 3 very well.

The encryption tests were all useful and were necessary to make sure the methods worked alright. The problem I had with all these is that I had to do "research" on Google to figure out what they were. I had no idea what "Caesar" encryption or "Inverse Cipher" encryption were. In fact, I had a lot of trouble finding the term "inverse cipher". Also, all of these encryptions usually require the user to choose a "shift" value and none of that was provided or included as a parameter which I felt it should have.

3.) Were some test suites unnecessary? Were some missing?

All of the 4 Encryption/Decryption test cases were definitely important and it's great that they were included. I felt like maybe there could have been some test cases added that tested what happened if a number or symbol was included in the string to decrypt .

I think most of the File functions that were tested were spot on. I don't think that an "empty file" test is necessarily important, mainly because the other tests checked the values to make sure they had been properly imported from the file. While the writeMessageData method is a Boolean value, I still think there should be a test to make sure that the new file that's been outputted contains all the right data somehow.

4.) Overall Feedback:

Overall I feel as though the tester gave a sufficient amount of test cases to help me know what I needed to do to implement the code. I kind of wish there were at least a few comments in the skeleton code to help make things clearer. The test suites did do a good job of making up for that though by generally providing a good idea of what should happen in each method.

In the real world, I think I would have had a pretty good understanding of this program by the test cases. While I may have had to do some research on the "encryption" methods, I feel as though in a company setting I would have had that information provided to me more than likely in the overall project description. I did learn a lot about encryption stuff from this project and felt a sense of accomplishment as I passed each test case and had no major issues passing any of the ones provided.

Additional Test Cases

Syntax Coverage

Class: EncryptionHandling
Method Chosen: String caesarMessageEncryption(String decryptedMessage)

- **Mutant 1:** else if(alphabet <= 'A' && alphabet <= 'Z');
  [Changed alphabet >= 'A' to <= 'A']

  Description: This method works by checking to see if the value is a lowercase or uppercase letter or a blank space. In this part of the method it evaluates if it's a capital letter, so I changed this operator because the value must be between A and Z to be a valid capital letter. A lower value than A would make it a symbol character which is not valid input for this encryption method!

- **Mutant 2:** else if(alphabet >= 'A' || alphabet <= 'Z');
  [Changed alphabet >= 'A' && to ||]

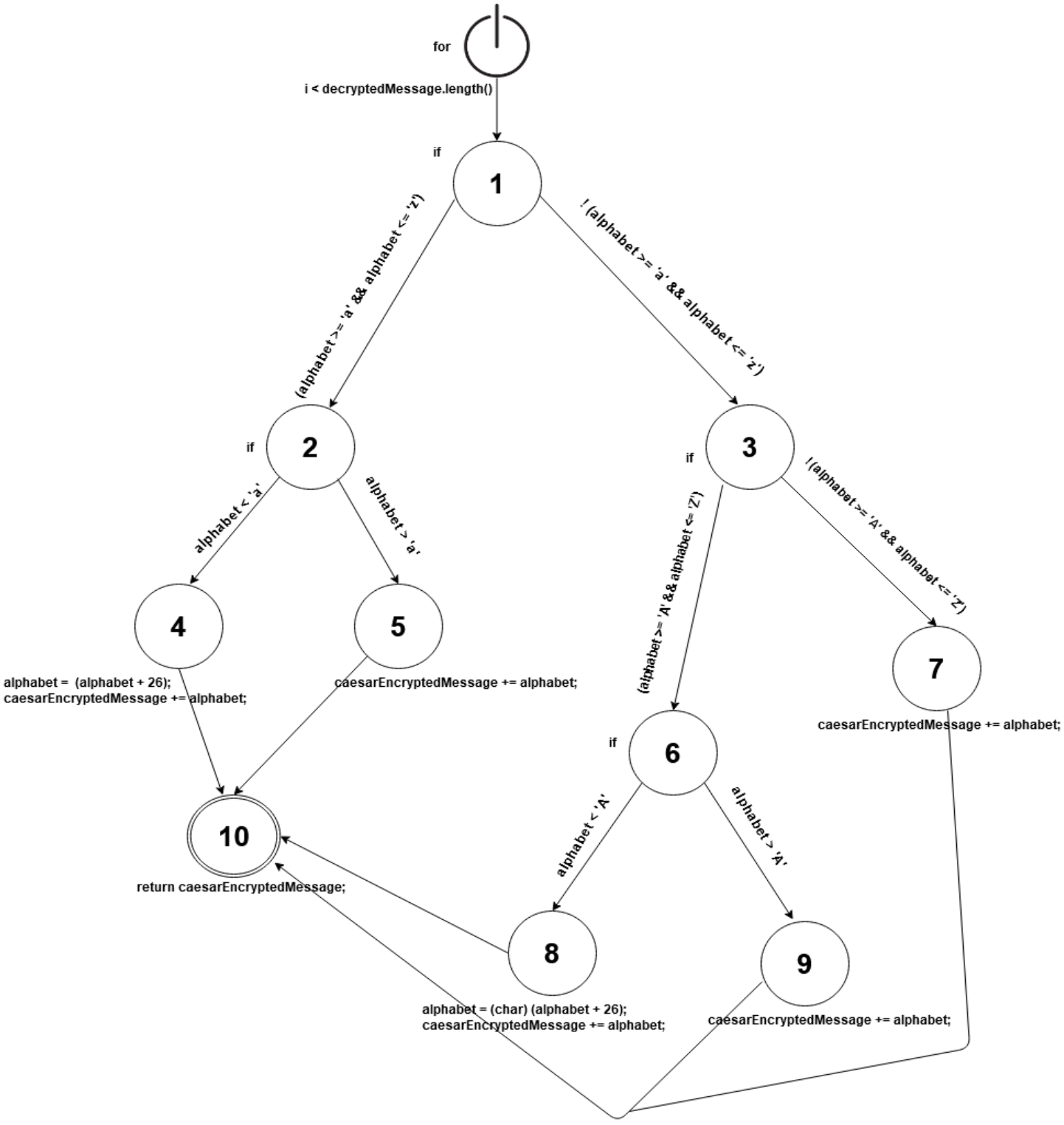
  Description: As mentioned above, this method works by checking to see if the value is a lowercase or uppercase letter or a blank space. In this part of the method it checks if it's a valid capital letter between the range of A to Z. It uses the && operator to make sure the value is BOTH in the range of A or higher and Z or lower. By changing the && to || then that removes that check and could allow a character value higher than Z to be accepted in the if statement. In other words, for example the character ~ may be allowed to be processed as it has a higher value than A!

- **Mutant 3:** alphabet = (char) (alphabet + shift);
  [Changed alphabet – shift to alphabet + shift]

  Description: In this part of the method, it takes the value of alphabet and sets it to a new value set by the value of the shift variable. In the case of the Caesar encryption method that is being implemented in this program, we want to shift the values "left" and not right. For this reason, we need to subtract the shift value from alphabet and not add it. If we were to add it then value of alphabet will be moved right. So, if we take the letter H and shift it left 3 times then we would be at E, but if we use "+" then that would cause it to shift right 3 times to the value K which would be incorrect for this program.

Graph Coverage

Class: EncryptionHandling
Method Chosen: String caesarMessageEncryption(String decryptedMessage)



Test Paths that satisfy Prime Path Coverage

1.) N1 -> N3 -> N6 -> N9 - > N10
*Input value: alphabet = ‘K’

2.) N1 -> N2 -> N5 -> N10
*Input value: alphabet = ‘k’

3.) N1 -> N3 -> N6 -> N8 -> N10
*Input value: alphabet = ‘A’

4.) N1 -> N2 -> N4 -> N10
*Input value: alphabet = ‘a’

5.) N1 -> N3 -> N7 -> N10
*Input value: alphabet = ‘ ‘ (blank space)