Name: Samantha Shon

UID: 506170927

Notable Obstacles:

A lot of my builds would succeed however, it is hard to find small errors in such a large and specific code.

Additionally, I had problems with which types of variables to use especially for the “for” loops. I find that

having more options in the style of coding is overwhelming. For example, for “for” loops, it was difficult to

Choose the data type between char, int, and size\_t. This was the most time consuming part of coding this

project. Smaller errors were definitely hard to fix. For some reason the usage of a variable such as “qpd”

would lead to a successful output only if in the “for” loops, I wrote “qpd + 1” instead of “qpd++.” Small

Syntax errors such as this made this project noticeably more difficult than the previous one.

Description:

Function: isValidQC

Input: string (results)

Output: bool

1. Set integers passes, defects, and tests = 0
2. If results is empty, return false
3. If string doesn’t start with ‘Q,’ return false
4. For each character at qpd in results:
   1. If ‘Q’ is the first character and is not followed by a digit, return false
   2. If ‘Q’ is followed by a 0, return false
   3. The digit that follows ‘Q’ in results is added to the numeric value of tests
   4. If ‘p’ is not followed by a digit, return false
   5. The digit that follows ‘p’ in results is added to the numeric value of passes
   6. If ‘d’ is not followed by a digit, return false
   7. The digit that follows ‘d’ in results is added to the numeric value of defects
   8. If a zero comes before another non zero value, return false
   9. If results contains characters that are not ‘Q,’ ‘p,’ or ‘d’ or a digit, return false
5. If tests equals passes added to defects, return true
6. If not, return false

Function: passQC

Input: string (results)

Output: int

1. If isValidQC(results) returns true:

a. Set passNum = 0

b. For each character at index p in results, starting from index 1:

i. If the current character is 'p' and the next character is a digit:

- Add the numeric value of the next character to passNum

c. Return passNum

2. Otherwise, return -1

Function defectQC:

Inputs: results (string)

Output: int

1. If isValidQC(results) returns true:

a. Set defectNum = 0

b. For each character at index d in results:

i. If the current character is 'd' and the next character is a digit:

- Add the numeric value of the next character to defectNum

c. Return defectNum

2. Otherwise, return -1

Function totalQC:

Inputs: results (string)

Output: int

1. If isValidQC(results) returns true:

a. Set testNum = 0

b. For each character at index t in results:

i. If the current character is 'Q' and the next character is a digit:

- Add the numeric value of the next character to testNum

c. Return testNum

2. Otherwise, return -1

Function batches:

Inputs: results (string)

Output: int

1. If isValidQC(results) returns true:

a. Set batchNum = 0

b. For each character at index b in results:

i. If the current character is 'Q', increment batchNum

c. Return batchNum

2. Otherwise, return -1

Test Data:

* Q2p1d1
  + One valid statement with one batch to test if the function will return true
* “Q2d1p1Q5p3d2
  + Valid statement with two batches to test if the code could return correct output (true) (including the new number of batches)
* Q2d1p0
  + Make sure that the code could find that defects and passes don’t add to correct test number
* Q2d1p1 asdf
  + No excess characters
* “ “
  + No empty strings
* q1p0d1
  + No uncapitalized ‘Q’
* Q3pd3
  + Check for number after ‘p’
* Q2p2d
  + Check for number after ‘d’
* Q5p00000002d0000003
  + No leading zeros
* Q0p0d0
  + ‘Q’ cannot be zero
* Q22p11d11
  + Make sure that code can handle two digit numbers
* Q20p10d10
  + Code could not handle two digit numbers with zeros at the end