Project 3 Report

1. Notable obstacles
   1. Checking for uppercase R
      1. First, I tried looking exactly for R by manually indexing through the string, but this could not work because some batches are smaller and larger than others due to the number of digits (Ex: R2+1-1 vs R200+100-100). Instead, I created a string that takes all alphabet characters from the given string and check only it for exactly R.
   2. Checking for both + and – in each batch
      1. First, I thought I had to differentiate batches from each other. I could not do it by manually indexing through the string because some batches are smaller and larger than others due to the number of digits (Ex: R2+1-1 vs R200+100-100). Instead, I found the total number of totals, +, and –. The total number of + signs and – signs need to match and their average should equal the total number of totals for to verify that there is both + and – in each batch.
2. Description of the design of the program
   1. Include needed libraries (iostream, string, cctype, cassert)
   2. Declarations of functions
   3. Main (included assert here to check program)
   4. int extractNumber function (given by Professor Howard)
   5. bool isValidResultString function
      1. Check for empty string “”
      2. Check for an uppercase R as only valid letter of the alphabet
         1. Index through string and create new string of found alphabet characters
            1. Index through new string and check for exactly “R”
      3. Check for a total after R
         1. Index through string and look for “R”
         2. Create new index that starts at the position after “R”
         3. Use new index and find the total using extractNumber
         4. Check if the total is less than or equal to 0
         5. Update total number of tests (true value)
      4. Check for leading 0s before numeric values
         1. Index through string and look if the position before a digit is a 0
      5. Check if the last character in the string is a digit
         1. Define the last position as the size of the string minus 1
         2. Check if the character at the last position is a digit
      6. Check if the total number of tests equals the total positive and negative tests
         1. Index through string and search for “+” or “-“
         2. Create new index that starts in position after “+” or “-“
            1. Check if digit at new position
            2. Get value at new position using extractNumber
            3. Update variable used to check total
         3. Check if the check total variable equals the true total value
      7. Check for “+” and “-“ in each batch
         1. Index through string
            1. Look for “R”, check if character at position after if a digit 1-9

Update counter for number of totals

* + - * 1. Look for “+” and update counter for number of positives
        2. Look for “-“ and update counter for number of negatives
      1. Compare number of positive and negatives and if the average equals the number of totals
  1. int postiveTests function
     1. If isValidResultString is true, calculate total number of positive tests
        1. Index through string and find “+”
           1. Use extractNumber to get number at position after “+”
           2. Update total number of positive tests by adding to it the extracted number
        2. Return total number of positive tests
     2. Otherwise, return -1
  2. negativeTests function
     1. If isValidResultString is true, calculate total number of negative tests
        1. Index through string and find “-”
           1. Use extractNumber to get number at position after “-”
           2. Update total number of negative tests by adding to it the extracted number
        2. Return total number of negative tests
     2. Otherwise, return -1
  3. totalTests function
     1. If isValidResultString is true, calculate total number of tests
        1. Index through string and find “R”
           1. Use extractNumber to get number at position after “R”
           2. Update total number of tests by adding to it the extracted number
        2. Return total number of tests
     2. Otherwise, return -1
  4. batches function
     1. f isValidResultString is true, calculate total number of positive tests
        1. Index through string and find “R”
           1. Update batches counter by 1
        2. Return batches counter
     2. Otherwise, return -1

1. List of test data

assert(isValidResultString("") == **false**); //empty string

assert(isValidResultString(" ") == **false**); //string with just whitespace

assert(isValidResultString("R+1-1") == **false**); //need number ot total tests

assert(isValidResultString("R3+1-2") == **true**); //valid string

assert(isValidResultString("R0+0-0") == **false**); //must have a number of cases to report

assert(isValidResultString("R0-0+0") == **false**); //must have a number of cases to report

assert(isValidResultString("R5+0-0") == **false**); //total must equal pos and neg tests

assert(isValidResultString("r1+0-1") == **false**); //must have capital R

assert(isValidResultString("R1+-1") == **false**); //need number of positive tests

assert(isValidResultString("R1+1-") == **false**); //need number of negative tests

assert(isValidResultString("R360-300+60") == **true**); //int values are not limited to a single digit

assert(isValidResultString("R1+0-1 asdfR") == **false**); //no extra characters

assert(isValidResultString("R3-1+2 ") == **false**); //no extr characters, spec, spaces

assert(isValidResultString("R5+00003-0002") == **false**); //no leading zeros

assert(isValidResultString("R5-3-2") == **false**); // + and - required every batch

assert(isValidResultString("R5+3+2") == **false**); // + and - required every batch

assert(isValidResultString("R2-1+1R5+3-2") == **true**); //can take in multiple batches

assert(positiveTests("R3+2-1") == 2); //2 positive results

assert(positiveTests("R1+1-0R1-1+0R1-1+0R1+1-0") == 2); //positive results from more than 1 batch

assert(positiveTests("R5+00003-0002") == -1); //invalid string (no leading 0s)

assert(positiveTests("R3+2+1") == -1); //invalid string (need + and - each batch)

assert(negativeTests("R3+2-1") == 1); //1 negative result

assert(negativeTests("R3+2-1R9+5-4") == 5); //negative results from more than 1 batch

assert(negativeTests("R3-2-1") == -1); //return -1 for invalid string

assert(totalTests("R3+2-1") == 3); //3 total tests

assert(totalTests("R3+2-1R9+5-4") == 12); //total tests from more than 1 batch

assert(totalTests("R3-2-1") == -1); //return -1 for invalid string

assert(batches("R3+2-1") == 1); //1 batch

assert(batches("R3+2-1R9+5-4") == 2); //more than 1 batch

assert(batches("R3-2-1") == -1); //return -1 for invalid string