CS 150 Programming Exam 3

For this exam, you'll complete the processing portion of two short looping functions using a Windows toolchain that is simpler than the Cloud9 IDE. You may not access any notes or files, other than the C++ quick reference I've supplied, or communicate with anyone in any way. You must use only the tools provided; do not open up any other software (such as Visual Studio). This exam is worth 50 points and *you will have 45 minutes*.

You can check each problem by using `Tools->Compile` to see if your code is ***syntactically correct***. To test your code, use `Tools->Go`. Code you wish to see printed (instead of just tested) should be placed in the `studentTests()` function.

Strings and Iteration

Each problem appears in its own file, `p1.cpp` and `p2.cpp`. Each file contains a single function with the header already written. Finish each function.

- P1 – ***simple iteration*** can be solved with a single loop (usually a "for" loop).
- P2 – ***intermediate iteration*** will require one or more loops in combination with several selection statements.

A good strategy is to spend 15-20 minutes on Problem 1 and then tackle Problem 2. Make sure you at least look at each problem. Do not spend all of your time stuck on one of them.

CS 150 Quick-Reference

```
int main() { return 0; } // basic C++ structure
```

Standard Library Headers

```
#include <iostream>    // input-output streams
#include <iomanip>     // stream manipulators
#include <cmath>       // all math functions
#include <string>      // c++ style strings
#include <cctype>      // character classification
#include <cstdlib>     // exit codes
using namespace std;   // make sure standard namespace is used
```

Comments

```
/* inline or multi-line */   // end of line or single-line
```

Input/Output

```
cout << anything << endl;    // only need iostream
cout << fixed << setprecision(2) << setw(12) << value; // need iomanip
cin >> anyVar;               // skips whitespace, reads one token (word)
cin >> noskipws >> ch;       // reads a character including whitespace
getline(cin, stringVar);     // needs <string> include, reads line
```

// Escape sequences

```
\t = tab, \n = newline, \" = quote, \\ = backslash \' = single-quote
```

// Selection—simple if (independent decisions)

```
if (condition)       // boolean or numeric (non-zero) value
   statement;        // multi-line? enclose in { } block
else
   statement;
```

// Selection—nested if (leveled decisions)

```
if (conditionA)
   if (conditionB)
      statement-if A and B
   else
      statement-if A and not B
else
   if (conditionB)
      statement-if not A and B
```

```
   else
      statement-if not A and not B
```

// Selection—sequential if (dependent decisions)

```
if (conditionA)
   statement-if A
else if (conditionB)
   statement-if B
else if (conditionC)
   statement-if C
else
   statement-if not A or B or C
```

// Selection—numbered decisions (single test against a constant)

```
switch (integer-expression-test)
{                              // braces required
   case 1:                     // case block for integer-expression == 1
      statement;
      statement;
      break;                   // needed to end block; fall-through otherwise
   case 5:                     // case block for integer-expression == 5
      statement;
      statement;
      break;                   // needed to end block; fall-through otherwise
   default:                    // optional block (else for switch)
      statement;               // if expression != 1 and != 5
}
```

// While Loops

```
int num, sum = 0;   // Sentinel summing loop, primed variety
cin >> num;
while (cin && num >= 0)  // test both cin and num; exit on negative number
{
   sum += num;
   cin >> num;    // read number again to go to next iteration
}

int num, sum = 0;                  // Sentinel summing loop with inline-test
while ((cin >> num) && num >= 0)  // test both cin and num; exit on negative number
{
   sum += num;
}
```

```
string str;                 // Assume these three statements in front of rest of loops
getline(cin, str);
int len = str.size();       // or str.length()

int i = 0, vowels = 0;      // Counted summing loop (process string or array)
while (i < len)             // always make sure you go < len
{
   char c = str.at(i);   // less safe: str[i]; as string: str.substr(i, 1)
   if (c == 'a' || c == 'e' || str == 'i' || str == 'o' || str == 'u')
      vowels++;

   i++;    // don't forget to update the counter
}
```

// For Loops

```
string str;                 // Assume these three statements in front of rest of loops
getline(cin, str);
int len = str.size();       // or str.length()

int digits = 0;   // using a for loop instead
for (int i = 0; i < len; i++)
   if (isdigit(str.at(i))) digits++;

//Other cctype functions: ispunct(), isspace(), isupper(), islower(), isalpha(), toupper(), tolower()

string result = "";                    // remove all "dog"s from str
for (int i = 0; i < len - 2; i++)   // note condition is < len – (3 – 1)
{
   string subs = str.substr(i, 3);   // note difference from Java; second is length
   if (subs == "dog")
      i += 2;  // skip over dog in output
   else if (i == len - 3)   // last three characters not dog
   {
      result += subs;
      i += 2;
   }
   else
      result += subs.at(0);   // put next character in output
}
```