EGR 125 - Introduction to Engineering Methods (C++) Due date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

File: N125-Ch10LB

**Chapter 10 Homework – Strings**

**Reading Assignment:**

Read Chapter 10, Sections 1-2 in Introduction to Programming with C++, 3rd Edition by Liang

**Problem Assignment:**

1. (12 pts) ***Check Point Exercise 10.2***:

Suppose that s1 and s2 are two strings, given as follows:

string s1("I have a dream");

string s2("Computer Programming");

Assume that each expression is independent. What are the results of the following

expressions?

­

(1) s1.append(s2) | ***I have a dreamComputer Programming***

(2) s1.append(s2, 9, 7) | ***I have a dreamProgram***

(3) s1.append("NEW", 3) | ***I have a dreamNEW***

(4) s1.append(3, 'N') | ***I have a dreamNNN***

(5) s1.assign(3, 'N') | ***NNN***

(6) s1.assign(s2, 9, 7) | ***Program***

(7) s1.assign("NEWNEW", 3) | ***NEW***

(8) s1.assign(3, 'N') | ***NNN***

(9) s1.at(0) | ***I***

(10) s1.length() | ***14***

(11) s1.size() | ***14***

(12) s1.capacity() | ***14***

(13) s1.erase(1, 2) | ***Iave a dream***

(14) s1.compare(s2) | ***1***

(15) s1.compare(0, 10, s2) | ***1***

(16) s1.c\_str() | ***I have a dream***

(17) s1.substr(4, 8) | ***ve a dre***

(18) s1.substr(4) | ***ve a dream***

(19) s1.find('A') | *18446744073709551615* (Error?)

(20) s1.find('a', 9) | ***12***

(21) s1.replace(2, 4, "NEW") | ***I NEW a dream***

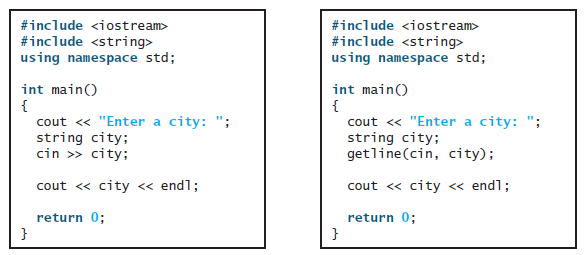
(22) s1.insert(4, "NEW") | ***I haNEWve a dream***

(23) s1.insert(6, 8, 'N') | ***I haveNNNNNNNN a dream***

(24) s1.empty() | ***0***

1. (4 pts) ***Check Point Exercise 10.5:***

Suppose you entered “New York” when running the following programs.

******

***New New York***

1. (22 pts) ***Email Address Creation***: A company named Creative Engineering develops email addresses for their employees by using the first 5 digits of their last name (or all digits if less than 5), the first digit of their first name, and the last 3 digits of their EmployeeID, followed by “@creative.com”. Write a program that will prompt the user to enter his/her last name, first name, and their EmployeeID. The program should then display their email address. Turn in a printout of the program and the results for the 3 cases below. The 3rd case uses **your** name and any 7-digit EmployeeID you pick (make one up).

#include <iostream>

#include <string>

using namespace std;

int main()

{

string Lname, Fname, EID, output;

cout << " Please enter your Last Name:";

cin >> Lname;

cout << "\n Please enter your First Name:";

cin >> Fname;

cout << "\n Please enter your Employee ID:";

cin >> EID;

output = Lname.substr(0, 4) + Fname.substr(0, 1) + EID.substr(4, 6) + "@creative.com";

cout << "\n Your Email Address is " << output;

return 0;

}

|  |  |
| --- | --- |
| **Inputs** | **Output** |
| Last Name: Stephens  First Name: Clarence  EmployeeID: 7654321 | Your email address is StephC321@creative.com |
| Last Name: Doe  First Name: John  EmployeeID: 1020030 | Your email address is DoeJ030@creative.com |
| Last Name: Vermaak  First Name: David  EmployeeID: 0076935 | Your email address is VermaD935@creative.com |

***Please enter your Last Name:Stephens***

***Please enter your First Name:Clarence***

***Please enter your Employee ID:7654321***

***Your Email Address is StepC321@creative.com***

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Please enter your Last Name:Doe***

***Please enter your First Name:John***

***Please enter your Employee ID:1020030***

***Your Email Address is DoeJ030@creative.com***

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Please enter your Last Name:Vermaak***

***Please enter your First Name:David***

***Please enter your Employee ID:0076935***

***Your Email Address is VermD935@creative.com***

4. (22 pts) ***Date Format Conversion***: Create a simple data file like the example shown below containing the 4 dates below plus 10 or more additional dates. The file should include 1 date per line and each date should have the form: **Month DayOfTheMonth, Year**. All dates should be in this century. No error checking for invalid dates is necessary. <https://www.random.org/calendar-dates>

February 19, 2009

July 4, 2017

November 4, 2018

September 30, 2019

.

.

Write a program that will read the dates in the input date file and create an output data file with the form **MonthNumber-DayOfTheMonth-Last2DigitsOfTheYear**  with no extra spaces. Example:

2-19-09

7-4-17

11-4-18

9-30-19

.

.

***Program 10\_4***

#include <fstream> //This header allows the use of functions that edit files

using namespace std; //introduces namespace std

//Function declarations

//Function to return the number that corresponds to the month

int month (string s);

int main ( )

{

string Month, syear, output;

int Day, Year;

char c;

vector <string> outputV;

int N1;

//open file 1

ifstream infile("F:\\Date\_1.txt");

for(int i=0; i<14; i++)

{

infile >> Month >> Day >> Year;

N1 = month(Month);

syear = to\_string(Year);

output = to\_string(N1) + "-" + to\_string(Day) + "-" + syear.erase(0, 2);

outputV.push\_back(output);

}

infile.close();

//open file 2

ofstream outfile("F:\\Date\_2.txt");

for(int i=0; i<=14; i++)

{

outfile << outputV[i] <<"\n";

}

outfile.close();

return 0;

}

//Function

int month (string s)

{

if (s == "January") return 1;

else if (s == "February") return 2;

else if (s == "March") return 3;

else if (s == "April") return 4;

else if (s == "May") return 5;

else if (s == "June") return 6;

else if (s == "July") return 7;

else if (s == "August") return 8;

else if (s == "September") return 9;

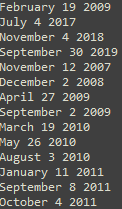
else if (s == "October") return 10;

else if (s == "November") return 11;

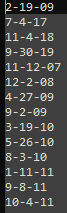
else if (s == "December") return 12;

}

***Input File: Data\_1.txt***



***Output File: Data\_2.txt***

******

5. (22 pts) ***Searching a Dictionary***: Download the file **USDictionary.txt** from the course Blackboard site that contains words in the US dictionary (about 118,000 words – all in lower case). Write a C++ program that will determine and display the following items:

* The total number of words in the dictionary.
* The total number of characters in the dictionary (not including white spaces)
* The total number of characters in the dictionary (including white spaces) – Hint: Use get( ).
* The total number of words ending in the letter b.
* The total number of words ending in the letter t.
* The total number of 5 letter words.
* The total number of words ending with a vowel.
* The total number of words containing the substring “est”.
* The total number of occurances of the letter t.
* The total number of words containing at least two occurances of the letter t.
* The total number of words containing 3 or more vowels.

Turn in a printout of the program and the results.

Name: \_\_\_David Vermaak\_\_\_\_\_

6. (18 pts) Determine the output of the program below **by hand**. This is good test practice.

|  |  |
| --- | --- |
| // Project: StringsHW  #include <iostream>  #include <string>  using namespace std;  int main()  {string S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11,S12,S13 = "aeiou";  int I1,I2,I3,I4,I5;  int Count1 = 0, Count2 = 0;  S1 = "Never memorize something that you can look up.";  // A. Einstein  S2 = S1.substr(0,5);  I1 = S1.length();  S3 = S1.substr(I1-8,7);  S4 = S2 + " " + S3;  S5 = S1.substr(6,2);  I2 = S1.find(S5);  I3 = S1.rfind(S5,I1-1);  S6 = S1.substr(I2,8);  S7 = S1.substr(I3-2,9);  S8 = S4;  S8.append(" " + S2,0,6);  S9 = S6 + " " + S7;  I4 = S1.find\_first\_of(S13);  I5 = S1.find\_last\_of(S13);  S10 = S1;  S10.erase(0,6);  S10.insert(0,"Always",7);  S11 = S1;  S11.replace(37,8,"'t look up");  S12 = S6;  for (int i = 0; i < I1; i++)S12[i]=toupper(S12[i]);  for (int i = 0; i < I1; i++)  { if(S1[i] == 'e') Count1++;  if(S1[i] == 32) Count2++; }  cout << "S2 = " << S2 << endl;  cout << "I1 = " << I1 << endl;  cout << "S3 = " << S3 << endl;  cout << "S4 = " << S4 << endl;  cout << "S5 = " << S5 << endl;  cout << "I2 = " << I2 << endl;  cout << "I3 = " << I3 << endl;  cout << "S6 = " << S6 << endl;  cout << "S7 = " << S7 << endl;  cout << "S8 = " << S8 << endl;  cout << "S9 = " << S9 << endl;  cout << "I4 = " << I4 << endl;  cout << "I5 = " << I5 << endl;  cout << "S10 = " << S10 << endl;  cout << "S11 = " << S11 << endl;  cout << "S12 = " << S12 << endl;  cout << "Count1 = " << Count1 << endl;  cout << "Count2 = " << Count2 << endl;  return 0; } | Program Output:  S2 = \_\_\_Never\_\_\_  I1 = \_\_\_46\_\_\_  S3 = \_\_\_look up\_\_\_  S4 = \_\_\_Never look up\_\_\_  S5 = \_\_\_me\_\_\_  I2 =\_\_\_ 6\_\_\_  I3 = \_\_\_17\_\_\_  S6 = \_\_\_memorize\_\_\_  S7 = \_\_\_something\_\_\_  S8 = \_\_\_Never look up Never\_\_\_  S9 = \_\_\_memorize something\_\_\_  I4 = \_\_\_1\_\_\_  I5 = \_\_\_43\_\_\_  S10 = \_\_\_Always memorize something  that you can look up. \_\_\_  S11 = \_\_\_Never memorize something that you can't look up.\_\_\_  S12 = \_\_\_MEMORIZE\_\_\_  Count1 = \_\_\_5\_\_\_  Count2 = \_\_\_7\_\_\_ |