Chapter 8: The class string & File input output Handling

Laboratory Exercises (5)

EXAMPLE 8-14 (clear, empty, erase, length, AND size FUNCTIONS)

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
string firstName = "Elizabeth";
string name = firstName + " Jones";
string str1 = "It is sunny.";
string str2 = "";
string str3 = "computer science";
string str4 = "C++ programming.";
string str5 = firstName + " is taking " + str4;
string::size_type len;

cout << "Line 9: str3: " << str3 << endl;
str3.clear();
cout << "Line 11: After clear, str3: " << str3<< endl;
cout << "Line 12: str1.empty(): " << str1.empty()<< endl;
cout << "Line 13: str2.empty(): " << str2.empty() << endl;
cout << "Line 14: str4: " << str4 << endl;
str4.erase(11, 4);
cout << "Line 16: After erase(11, 4), str4: "<< str4 << endl;
```

```
cout << "Line 17: Length of \"" << firstName << "\" = " << static_cast<unsigned int>
(firstName.length())<< endl;

cout << "Line 18: Length of \"" << name << "\" = "<< static_cast<unsigned int> (name.length())<< endl;

cout << "Line 19: Length of \"" << str1 << "\" = "<< static_cast<unsigned int> (str1.length())<< endl;

cout << "Line 20: Size of \"" << str5 << "\" = "<< static_cast<unsigned int> (str5.size()) << endl;

len = name.length();

cout << "Line 22: len = "<< static_cast<unsigned int> (len) << endl; return 0;


return 0;

}
```

EXAMPLE 8-15 (find FUNCTION)

```
#include <iostream>

#include <string>

using namespace std;

int main()

{

string sentence = "Outside it is cloudy and warm.";

string str = "cloudy";

string::size_type position;


cout << "Line 4: sentence = \"" << sentence << "\"" << endl;

cout << "Line 5: The position of \"is\" in sentence = "<< static_cast<unsigned int> (sentence.find("is"))<<
endl;

cout << "Line 6: The position of 's' in sentence = "<< static_cast<unsigned int> (sentence.find('s'))<<
endl;

cout << "Line 7: The position of \"" << str<< "\" in sentence = "<< static_cast<unsigned int>
(sentence.find(str))<< endl;

cout << "Line 8: The position of \"the\" in sentence = "<< static_cast<unsigned int>
(sentence.find("the"))<< endl;
```

```
cout << "Line 9: The first occurrence of \'i\' in "<< "sentence \n after position 6 = "<<
static_cast<unsigned int> (sentence.find('i', 8))<<endl;

position = sentence.find("warm");

cout << "Line 11: " << "Position = "<< position << endl;


return 0;

}
```

EXAMPLE 8-16 (insert AND replace FUNCTIONS)

```
#include <iostream>

#include <string>

using namespace std;

int main()

{

   string firstString = "Cloudy and warm.";

   string secondString = "Hello there";

   string thirdString = "Henry is taking programming I.";

   string str1 = " very ";

   string str2 = "Lisa";


cout << "Line 6: firstString = " << firstString<< endl;

firstString.insert(10, str1);

cout << "Line 8: After insert; firstString = "<< firstString << endl;


cout << "Line 9: secondString = " << secondString << endl;

secondString.insert(11, 5, '!');

cout << "Line 11: After insert; secondString = "<< secondString << endl;


cout << "Line 12: thirdString = " << thirdString << endl;
```

```
thirdString.replace(0, 5, str2);
cout << "Line 14: After replace, thirdString = "<< thirdString << endl;
return 0;
}
```

EXAMPLE 8-17 (substr FUNCTION)

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
string sentence;
string str;
sentence = "It is cloudy and warm.";

cout << "Line 4: substr(0, 5) in \""<< sentence << "\" = \""<< sentence.substr(0, 5) << "\"" << endl;
cout << "Line 5: substr(6, 6) in \""<< sentence << "\" = \""<< sentence.substr(6, 6) << "\"" << endl;
cout << "Line 6: substr(6, 16) in \""<< sentence << "\" = " << endl<<" \"" << sentence.substr(6, 16) <<
"\"" << endl;
cout << "Line 7: substr(17, 10) in \""<< sentence << "\" = \""<< sentence.substr(17, 10) << "\"" << endl;
cout << "Line 8: substr(3, 6) in \""<< sentence << "\" = \""<< sentence.substr(3, 6) << "\"" << endl;

str = sentence.substr(0, 8);
cout << "Line 10: " << "str = \"" << str<< "\"" << endl;

str = sentence.substr(2, 10);
cout << "Line 12: " << "str = \"" << str<< "\"" << endl;
return 0;
```

```
}
```

Challenge Program!

Write a program that read your full name (first and last) in one string using *getline()* function, and then prints back your last name.

```
#include <iostream>

#include<string>

using namespace std;


int main()

{

int Pos, lnamePos, lnameLength;

string name;

string lname;


cout<<"Enter you first name and last name: ";

getline(cin, name);

Pos=name.find(' ');

lnamePos= Pos+1;

lnameLength=name.length()-lnamePos;


lname=name.substr(lnamePos, lnameLength);


cout<<"\n Your last name is: "<<lname<<endl;


return 0;

}
```

Objects and Classes in C++ Programming

```
// Program to illustrate the working of
// objects and class in C++ Programming

#include <iostream>
using namespace std;

// create a class
class Room {

  public:
   double length;
   double breadth;
   double height;



   double calculateArea() {
      return length * breadth;
   }

   double calculateVolume() {
      return length * breadth * height;
   }
};

int main() {
   // create object of Room class
   Room room1, room2;
```

```
    // assign values to data members for room1
    room1.length = 2.5;
    room1.breadth = 3.8;
    room1.height = 2.2;
    // calculate and display the area and volume of room2
    cout << "Area of Room =  " << room1.calculateArea() << endl;
    cout << "Volume of Room =  " << room1.calculateVolume() << endl;

    // assign values to data members for room2
    cout<<"Enter the dimissions of room 2:"<<endl;
    cin>>room2.length;
    cin>>room2.breadth;
    cin>>room2.height;

    // calculate and display the area and volume of room2
    cout << "Area of Room =  " << room2.calculateArea() << endl;
    cout << "Volume of Room =  " << room2.calculateVolume() << endl;
    return 0;
}
```

Using public and private in C++ Class

```
// Program to illustrate the working of
// public and private in C++ Class

#include <iostream>
using namespace std;

class Room {

  private:
   double length;
   double breadth;
   double height;

  public:
   // function to initialize private variables
   void initData(double len, double brth, double hgt) {
      length = len;
      breadth = brth;
      height = hgt;
   }

   double calculateArea() {
      return length * breadth;
   }

   double calculateVolume() {
      return length * breadth * height;
```

```
    }
};

int main() {

    // create object of Room class
    Room room1;

    double l, b, h;
    cin>>l;
    cin>>b;
    cin>>h;
    // pass the values of private variables as arguments
    room1.initData(l, b, h);

    cout << "Area of Room =  " << room1.calculateArea() << endl;
    cout << "Volume of Room =  " << room1.calculateVolume() << endl;

    return 0;
}
```

Strings

KEYWORDS: Char array, Pass by reference.

Program 1: Write a C++ program to count all the vowels in a string.

```
#include <iostream>
#include<string>
using namespace std;

int Vowel_Count(string a, int size)
{
  int ctr = 0;
  for(int i = 0; i < size ; i++){
    if (a[i] == 'a' || a[i] == 'e' || a[i] == 'i' || a[i] == 'o' || a[i] == 'u')
      ctr++;
    if (a[i] == 'A' || a[i] == 'E' || a[i] == 'I' || a[i] == 'O' || a[i] == 'U')
      ctr++;
  }
  return ctr;

}

int main() {
	string str;
	cout << "Enter a string:"<< endl;
	getline(cin, str);
	int s = static_cast<unsigned int> (str.length());
			cout << "number of vowels is "<<Vowel_Count(str, s) << endl;
	return 0;
}
```

Program 2: Write a program that prompts the user to input a string. The program then uses the function substr to remove all the vowels from the string. For example, if str = "There", then after removing all the vowels, str = "Thr". After removing all the vowels, output the string. Your program must contain a function to remove all the vowels and a function to determine whether a character is a vowel.

```
#include <iostream>
#include <string>
using namespace std;
string c_vowels = "aeiou";
string s_vowels = "AEIOU";
bool is_vowel(char c)
{
   for(int i = 0; i < c_vowels.length(); i++)
   {
      if(c == c_vowels[i])
         return true;
      if(c == s_vowels[i])
         return true;
   }
   return false;
}
string remove_vowels(string str)
{
   string temp = "";
   for(int i = 0; i < str.length(); i++)
   {
      if(!is_vowel(str[i]))
      {
         temp += str.substr(i,1);
      }
```

```
    }
    str = temp;
    return str;
}
int main()
{
    string str;
    cout<<"Input string: ";
    cin>>str;
    cout<<remove_vowels(str);
    return 0;
}
```

File Input Output Handling

KEYWORDS: fstream, iomanip, inFile, outFile.

Program 1: Write a C++ program to write number 1 to 100 in a data file NOTES.TXT

```
#include<fstream>
using namespace std;

int main()
{
	ofstream fout;
	fout.open("NOTES.TXT");
	for(int i=1;i<=100;i++)
		fout<<i<<endl;
	fout.close();
	return 0;
}
```

Program 2: Write a program that reads a student name followed by five test scores from a file. The program should output the student's name, the five test scores, and the average test score to a file. Output the average test score with two decimal places. The data to be read is stored in a file called test.txt. The output should be stored in a file called testavg.out.

Input a file, test.txt, containing the student's name and the five test scores. A sample input is:

Andrew Miller 87.50 89 65.75 37 98.50

Output The student's name, the five test scores, and the average of the five test scores, saved to a file, testavg.txt.

```
#include <iostream>
#include <fstream>
#include <iomanip>
#include <string>
```

```
using namespace std;


int main()
{
   //Declare variables;  Step 1
   ifstream inFile;
   ofstream outFile;
   double test1, test2, test3, test4, test5;
   double average;
   string firstName;
   string lastName;
   inFile.open("test.txt");
   outFile.open("testavg.out");
   outFile << fixed << showpoint;
   outFile << setprecision(2);
   cout << "Processing data" << endl;
   inFile >> firstName >> lastName;
   outFile << "Student name: " << firstName<< " " << lastName << endl;
   inFile >> test1 >> test2 >> test3>> test4 >> test5;
   outFile << "Test scores: " << setw(6) << test1 << setw(6) << test2 << setw(6) << test3 << setw(6) <<
test4 << setw(6) << test5 << endl;
   average = (test1 + test2 + test3 + test4 + test5) / 5.0;
   outFile << "Average test score: " << setw(6) << average << endl;
   inFile.close();
   outFile.close();
   return 0;
}
```