# Group 3: Nguyễn Đức Duy Nguyễn Công Hoàng Trần Quốc Bảo

Project 3

## Programming Challenges

#### 1. String Length (1)

#### Write a function that returns an integer and accepts a pointer to a C-string as an argument. The function should count the number of characters in the string and return that number. Demonstrate the function in a simple program that asks the user to input a string, passes it to the function, and then displays the function’s return value.

**#include <iostream>**

**using namespace** std;

**int** getLength(**char** \*str) {

**int** r = 0;

**while** (str[r] != '\0') {

r++;

}

**return** r;

}

**int** main() {

**char** str[100];

cout << "Enter a string:\t";

cin.getline(str, 100);

cout << "Your string has " << getLength(str) << " characters." << endl;

**return** 0;

}

#### 2. Backward String (2)

#### Write a function that accepts a pointer to a C-string as an argument and displays its contents backward. For instance, if the string argument is “ Gravity ” the function should display “ ytivarG ”. Demonstrate the function in a program that asks the user to input a string and then passes it to the function.

#include <iostream>  
#include <cstring>

**using namespace** std;

**void** reverse(**char** \*str) {

size\_t l = strlen(str);

**for** (**int** i = 0; i < l / 2; ++i) {

**char** temp = str[i];

str[i] = str[l - i - 1];

str[l - i - 1] = temp;

}

cout << str << endl;

}

**int** main() {

**char** str[100];

cout << "Enter a string to reverse:\t";

cin.getline(str, 100);

cout << "\t-> ";

reverse(str);

**return** 0;

}

#### 3. Word Counter

#### Write a function that accepts a pointer to a C-string as an argument and returns the number of words contained in the string. For instance, if the string argument is “Four score and seven years ago” the function should return the number 6. Demonstrate the function in a program that asks the user to input a string and then passes it to the function. The number of words in the string should be displayed on the screen.

#### Optional Exercise: Write an overloaded version of this function that accepts a string class object as its argument.

#include <iostream>  
#include <cstring>

#include <cctype>

#include <string>

**using namespace** std;

**int** countWord(**char** \*str) {

size\_t len = strlen(str);

**int** count = 0;

**bool** isCurrentWord = **false**;

**for** (**int** i = 0; i < len; ++i) {

**if** (isalnum(str[i]) != 0) {

**if** (!isCurrentWord) {

isCurrentWord = **true**;

count++;

}

} **else** isCurrentWord = **false**;

}

**return** count;

}

**int** countWord(string str) {

**int** count = 0;

**bool** isCurrentWord = **false**;

**for** (**unsigned long** i = 0; i < str.length(); ++i) {

**char** c = str.at(i);

**if** ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z')) {

**if** (!isCurrentWord) {

isCurrentWord = **true**;

count++;

}

} **else** isCurrentWord = **false**;

}

**return** count;

}

**int** main() {

**char** s[256] = {0};

cout << "Enter a string:\t";

cin.getline(s, 256);

cout << "There are " << countWord(s) << " words using c-style string algorithm." << endl;

string s2(s);

cout << "There are " << countWord(s2) << " words using string object algorithm." << endl;

**return** 0;

}

#### 4. Average Number of Letters (4)

#### Modify the program you wrote for Problem 3 (Word Counter), so it also displays the average number of letters in each word.

#include <iostream>  
#include <cstring>

#include <cctype>

**using namespace** std;

**double** avgLetters(**char** \*str) {

size\_t len = strlen(str);

**int** count = 0;

**double** letters = 0.0;

**bool** isCurrentWord = **false**;

**for** (**int** i = 0; i < len; ++i) {

**if** (isalnum(str[i]) != 0) {

**if** (!isCurrentWord) {

isCurrentWord = **true**;

count++;

}

letters += 1;

} **else** isCurrentWord = **false**;

}

**return** letters / count;

}

**int** main() {

**char** str[256];

cout << "Enter a string:\t";

cin.getline(str, 256);

cout << "The string has an average of " << avgLetters(str)

<< " letters per word" << endl;

**return** 0;

}

#### 5. Most Frequent Character (9)

#### Write a function that accepts either a pointer to a C-string, or a string object, as its argument. The function should return the character that appears most frequently in the string. Demonstrate the function in a complete program.

#include <iostream>  
#include <string>

#include <cstring>

**using namespace** std;

**int** findMaxIndex(**int** l[], **int** size) {

**int** max = l[0], maxIndex = 0;

**for** (**int** i = 1; i < size; i++) {

**if** (max < l[i]) {

max = l[i];

maxIndex = i;

}

}

**return** maxIndex;

}

**char** mostFreqChar(**char** \*arr) {

**int** counter[95] = {0};

**for** (**int** i = 0; i < strlen(arr); i++) {

**if** (isprint(arr[i])) {

counter[(arr[i]) - 32]++;

}

}

**return static\_cast**<**char**>(findMaxIndex(counter, 95) + 32);

}

**char** mostFreqChar(string arr) {

**int** counter[95] = {0};

**for** (**int** i = 0; i < arr.length(); i++) {

**if** (isprint(arr[i])) {

counter[(arr[i]) - 32]++;

}

}

**return static\_cast**<**char**>(findMaxIndex(counter, 95) + 32);

}

**int** main() {

string arr;

cout << "Enter a string: ";

getline(cin, arr);

cout << "The character " << "'" << mostFreqChar(arr) << "'" << " appears most frequent" << endl;

**return** 0;

}