GECG10069 (561085) F25: Introduction to Programming (C++)

Lab 9 :-String & String

What you will learn from Lab8

In this laboratory, you will understand how to use library `<cstring>` and `<string>`.

TASK 9-1: C-STRING

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

int main()
{
    char stringA[20] = {'C','+','+','\0'};
     char stringB[20] = "Programming";

     cout << "length of " << stringA << " is " << strlen(stringA) << endl;
     cout << "length of " << stringB << " is " << strlen(stringB) << endl;

     char stringC[20];

     // Copy stringA to stringC
     strcpy(stringC, stringA);
     cout << stringC << endl;

     // Concatenate " " and stringB to stirngC
     strcat(stringC, " ");
     strcat(stringC, stringB);
     cout << stringC << endl;

     // Clear StringC
     strcpy(stringC, "");

     //compare characters of two strings
     cout << "Enter a new word :";
     cin.getline (stringC, 20);
     if(strcmp(stringC, stringB)==0)
         cout << "The two strings are the same!" << endl;
     else
         cout << "The two strings are different!" << endl;
      cout << endl << endl;

      //compare length and size of string
     char strA[50] = "Total length of string.";

     cout << "size of \"" << strA << "\" is " << sizeof(strA) << endl;
     cout << "length of \"" << strA << "\" is " << strlen(strA) << endl;

     char *pch = strchr(strA, 'l');
     cout << pch << endl;
```

```
    char *pch2 = strstr(strA, "length");
    cout << pch2 << endl;

    return 0;
}
```

TASK 9-2 : CLASS STRING

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

int main()
{
    string heading = "Hello";
    string ending("Welcome to my school!!");
    string name;

    cout << "Enter your name: ";
    getline(cin, name);

    string sentence = heading + ", " + name + "!!" + ending;
    cout << sentence << endl;

    string str;
    cout << "Enter a sentence:" << endl;
    getline(cin,str);

    int pos = str.find("nycu");
    if (pos == string::npos)
    {
        cout << "nycu is not found !!" << endl;
    }
    else
    {
       cout << "nycu is found at pos: " << pos << endl;
    }
   cout << "Substring from str[2]to[4] is " << str.substr(2,3) << endl;

    return 0;
}
```

- `str.find(str1)` returns index of the first occurrence of `str1` in `str`
- `string::npos`, `npos` indicates the end of the string. (`npos = -1`)
- `str.substr(pos,length)` returns the sub-string of `str` from index `pos` to index `pos + length`.

EXERCISE 9-1 : STRING LENGTH AND CHARACTER OUTPUT

Description -

Read a single word (**no spaces**) into a character array **`char s[51]`**.
Then print two lines:

1. The length of the string (excluding the null terminator **`\0`**).
2. Each character of the string is separated by a single space.

Input :

- One line containing a word **with no spaces** (`length ≤ 50`).

Output Format :

- Line 1: len=*length*
- Line 2: `s[0] s[1] s[2]` … `s[`*`length-1`*`]`

Sample Test Cases -

Sample Input - 1
Hello
Sample Output - 1
len=5 H e l l o

Sample Input - 2
PassTheDemo
Sample Output - 2
len=11 P a s s T h e D e m o

EXERCISE 9-2 : LEXICOGRAPHICALLY SMALLEST NAME

Description -

Write a program that reads several names,

and prints the one that comes first in lexicographical (dictionary) order.

Please use a fixed-size 2D C-string array (`char names[50][31]`).

You **must not** use `std::string`.

Input Format :

```
N M
<name1>
<name2>
...
<nameN>
```

- `N`: the number of names (`1 ≤ N ≤ 50`)
- `M`: the maximum length of each name (`1 ≤ M ≤ 30`)

Output Format :

- Print one line — the name that comes first lexicographically.

Hints :

- Use **`strcmp()`** from **`<cstring>`** for comparison
  - Returns < 0 if a < b
  - Returns > 0 if a > b
  - Returns 0 if the strings are equal

- By default, comparisons are **case-sensitive (`'A' < 'a'`, `'A' < 'B'`, `'a' < 'b'`)**.

Sample Testcases -

Sample Input - 1
6 20 Zoe alex Bob carol Amy banana
Sample Output - 1
Amy

Lexicographical order compares strings by ASCII values, character by character.
Since 'A' (65) < 'B' (66) < 'Z' (90) < 'a' (97) < 'b' (98) < 'c' (99),
the smallest name is “Amy”, which starts with 'A', the smallest letter among all.

Sample Input - 2
5 10 Apple Apex brian Ant Almond
Sample Output - 2
Almond

The four names start with 'A', so comparison continues with the second letter:
'l' (108) < 'n' (110) < 'p' (112) → therefore “Almond” comes first.