Algoritmos y Estructuras de Datos

Clase de Práctica 17 - Strings

Comisión A y F

Docente de práctica:

Tomás Assenza

Universidad Tecnológica Nacional - Facultad Regional Santa Fe

Ejercicios - Guía práctica

Strings - Ejercicio 2

Diseñar una función que permita registrar un nuevo usuario al sistema. Los identificadores deben tener 8 caracteres, comenzar con una letra y los restantes pueden ser letras o números. La función debe recibir un string y comprobar que la misma sea un identificador de usuario válido.

Ejemplos:

registrar("jperez12") --> true

registrar("1jperez123") --> false

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

bool registrar(string);

int main(int argc, char *argv[]) {
    cout << boolalpha;
    cout << registrar("jperez12") << endl;
    cout << registrar("1jperez123") << endl;
    return 0;
}

bool registrar(string identificador) {
    unsigned i = 1;
    bool hayOtroDigito = false;
    while (i < identificador.length() and !hayOtroDigito) {
        if (!isalnum(identificador[i])) hayOtroDigito = true;
        i++;
    }
    return !hayOtroDigito and
          identificador.length() == 8 and
          isalpha(identificador[0]);
}
```

Ejercicios - Guía práctica

Strings - Ejercicio 3

Realizar un programa que pida una frase y nos indique si es o no palíndroma (se lee igual de izquierda a derecha que de derecha a izquierda).

Ejemplo:

esPalindromo(“dabale arroz a la zorra el abad”) → True

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

bool esPalindromo(string);

int main(int argc, char *argv[]) {
    cout << boolalpha;
    cout << esPalindromo("dabale arroz a la zorra el abad") << endl;
    cout << esPalindromo("dabale arros a la zorra el abad") << endl;
    return 0;
}

bool esPalindromo(string cadenaAAnalizar) {
    int indiceIzquierdo = 0, indiceDerecho = cadenaAAnalizar.length() - 1;
    if (cadenaAAnalizar.length() > 0) {
        bool cumpleCondicionPalindroma = true;

        while (indiceIzquierdo < indiceDerecho and cumpleCondicionPalindroma) {

            while (cadenaAAnalizar[indiceIzquierdo] == ' ') indiceIzquierdo++;
            while (cadenaAAnalizar[indiceDerecho] == ' ') indiceDerecho--;

            if (cadenaAAnalizar[indiceIzquierdo]!=cadenaAAnalizar[indiceDerecho])
              cumpleCondicionPalindroma = false;

            indiceIzquierdo++;
            indiceDerecho--;
        }
    }
    return cadenaAAnalizar.length() == 0 or indiceIzquierdo >= indiceDerecho;
}
```

Ejercicios - Guía práctica

Strings - Ejercicio 1

Diseñar un programa que permita generar direcciones de correo electrónico. El programa recibe el apellido y nombre de un usuario de la facultad (apellido y nombre se asignan a una sola variable) y debe retornar la dirección de correo electrónico (e-mail) generada. El dominio asignado a la Facultad para el e-mail es: frsf.utn.edu.ar, y el nombre de usuario se forma con la inicial del nombre y a continuación el apellido.

Ejemplo:

Ingrese apellido y luego el nombre del profesor: Perez Juan

Email: jperez@frsf.utn.edu.ar

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

string armaMail(string);

int main(int argc, char *argv[]) {
    cout << armaMail("Assenza Tomas") << endl;
    return 0;
}

string armaMail(string nombreApellido) {
    string mail = "";

    if (nombreApellido.length() > 0) {
        mail += tolower(nombreApellido[nombreApellido.rfind(' ') + 1]);
        mail += tolower(nombreApellido[0]);

        mail += nombreApellido.substr(1, nombreApellido.find(' ') - 1);

        mail += "@frsf.utn.edu.ar";

    }

    return mail;

}
```

Ejercicios - Guía práctica

Strings - Ejercicio 7

Se dice que la palabra α es un anagrama de la palabra β si es posible obtener α cambiando el orden de las letras de β. Por ejemplo, “MORA” y “ROMA” son anagramas de RAMO. Realizar una función que compruebe si dos palabras son anagramas entre sí. Las palabras pueden contener letras mayúsculas y minúsculas.

```cpp
#include <iostream>
using namespace std;
bool sonAnagramas(string, string);

int main(int argc, char *argv[]) {
    cout << boolalpha;
    cout << sonAnagramas("MORA", "RAMO") << endl;
    cout << sonAnagramas("RAMO", "ROMA") << endl;
    cout << sonAnagramas("RAMO", "RoMA") << endl;
    cout << sonAnagramas("RAMO", "ROMO") << endl;
    return 0;
}

bool sonAnagramas(string primerString, string segundoString) {
    int frecuenciasPrimerString[124] {}, frecuenciasSegundoString[124] {};
    bool cumplenCondicionAnagrama = true;

    for (unsigned i = 0; i < primerString.length(); i++) {
        frecuenciasPrimerString[tolower(primerString[i])]++;
    }
    for (unsigned i = 0; i < segundoString.length(); i++) {
        frecuenciasSegundoString[tolower(segundoString[i])]++;
    }

    int i = 'a';
    while (i <= 'z' and cumplenCondicionAnagrama) {
        if (frecuenciasPrimerString[i] != frecuenciasSegundoString[i])
            cumplenCondicionAnagrama = false;
        i++;
    }
    return cumplenCondicionAnagrama;
}
```

Ejercicios - Guía práctica

Strings - Ejercicio 13

Desarrollar un programa en C++, el cual recibe un string como entrada. Este programa debe invocar a una función llamada invString_recursivo, la cual tiene como objetivo devolver el string invertido con respecto a su orden original, utilizando un enfoque recursivo para lograrlo. Asegúrese de que la función invString_recursivo sea implementada de manera recursiva.

Ejemplo:

Si se recibe la cadena “lee un texto”, se retorna “otxet nu eel”

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

string invString_recursivo(string entrada, int tamanioEntrada) {
    if (tamanioEntrada == 0) return "";
    return entrada[tamanioEntrada - 1] +
           invString_recursivo(entrada, tamanioEntrada - 1);

}

int main(int argc, char *argv[]) {
    cout << invString_recursivo("lee un texto", 12) << endl;
    return 0;
}
```