AEDD - Guía Práctica 13: String

Ejercicios propuestos:

Ejercicios propuestos:

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

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

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:

   Entrada “dabale arroz a la zorra el abad” → True

4. Dado 2 string, a y b, retornar otro string con la forma corto+largo+corto, de manera questring más corto esté en los extremos y el más largo en el medio. Los string no serán de la misma longitud pero sí pueden estar vacías (longitud 0).

   Ejemplos:

   comboString("Hello", "hi") → "hiHellohi"

   comboString("hi", "Hello") → "hiHellohi"

   comboString("aaa", "b") → "baaab"

5. Escribir un programa que dada un string indique el número total de caracteres y de palabras que la conforman. Tener en cuenta que los espacios también son caracteres. Se asume que la oración está bien conformada, teniendo un único espacio entre palabras, los signos de puntuación van inmediatamente después de la última letra de cada palabra, etc.

   Ejemplo:

   "El perro de San Roque no tiene rabo, Porque Ramón Ramírez se lo ha cortado."

   Palabras: 15

   Caracteres: 75

6. Realizar un programa que busque todas las ocurrencias de la palabra prohibida, que introduzca el usuario, en una frase leída y las reemplace por la cadena "XXX" (donde el número de "X" que contiene la cadena deberá ser igual al número de caracteres que tiene la palabra prohibida).

   Ejemplo:

   Introduce una frase: "Esto es una ese escondida y la he escrito yo"

   Introduce la palabra prohibida: "es"

   La frase resultante es: "Esto XX una XXe XXcondida y la he XXcrito yo

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.

8. Escribir un programa que limpie de ruidos una señal de entrada. La señal de entrada será una cadena con letras y números y la salida será la misma cadena eliminando los números.

   Ejemplo:

   Entrada: "Es2to0 3es u9na se88ñal c0on ru1id2os"

   Salida: "Esto es una señal con ruidos".

9. Escribir un programa que convierta una fecha en formato "MMDDYYYY" al formato "DD de mes del YYYY".

   Por ejemplo, para "12072006" debería devolver "7 de diciembre del 2019".

10. Dado un archivo de texto que contiene un telegrama que termina en punto, desarrollar una función que permita la lectura del texto desde el archivo y que:

    - informe la cantidad de veces que aparece cada vocal.
    - informe el porcentaje de espacios en blanco.
    - cuente y muestre la cantidad de palabras que posean más de 7 letras.
    - informe el costo del telegrama sabiendo que cada palabra cuesta $5,2.

    Ejemplo:

    Leyendo el telegrama ...

    En breve evaluación de Algoritmos a estudiar.

    El telegrama ingresado consta de:

    3 palabras con más de 7 letras

    Ocurrencias de vocales: 19

    - a: 5
    - e: 6
    - i: 3
    - o: 3
    - u: 2

    13.33% de espacios en blanco sobre el total de caracteres.

    Costo del telegrama: $36,40.

    Nota: Considerar que las palabras están separadas por un único espacio blanco. Los números se consideran como palabra.

11. Un texto se ha guardado en un string, donde cada componente tiene un carácter. Hay frases que terminan con punto y párrafos que terminan con ´#´.

    Ejemplo:

    El cielo está oscuro.Llega la noche.#Serenidad absoluta.Silencio profundo.#estoy aquí.

    - Se desea formatear el texto, modificando la lista de manera que luego de cada salto de línea se agreguen 3 “blancos”.
    - También se debe controlar que después de un punto el primer carácter sea mayúscula.
    - Se debe informar cuantas frases hay en todo el texto.

12. El cifrado por desplazamiento, es una de las técnicas de codificación de textos más simples y usadas. Es un tipo de cifrado por sustitución en el que una letra en el texto original es reemplazada por otra letra que se encuentra un número fijo de posiciones más adelante en el alfabeto. Por ejemplo, con un desplazamiento de 3 posiciones la A sería sustituida por la D (situada 3 lugares a la derecha de la A), la B sería reemplazada por la E, etc. Se supone que el alfabeto es circular de modo que a continuación de la Z comienza de nuevo la letra A.

    Se propone que programe una función que reciba como parámetros un string y el desplazamiento a realizar (cantidad de posiciones) y devuelva el texto codificado. Para simplificar la solución, previo a realizar la sustitución la función deberá pasar a mayúsculas el string recibido como argumento.

    Ejemplo:

    Si se recibe la cadena "UN TEXTO" y el desplazamiento 1, se retorna "VO UFYUP"

    Nota: Tener en cuenta que sólo se codifican los caracteres correspondientes a letras mayúsculas del alfabeto. Los espacios en blanco que aparecen en la cadena deben mantenerse.

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"

Anexo: Funciones de Manejo de Cadenas tipo String

Prototipo	Descripción
`string(const char *s);`	Devuelve la cadena tipo C *s* como una cadena tipo string.
`const char *c_str();`	Devuelve la cadena tipo string como una cadena tipo C terminada en `'\0'`.
`size_t length()` `size_t size()`	Devuelve la longitud de la cadena tipo string.
`istream &getline(istream &is, string &str);`	Extrae caracteres desde *is* y los guarda en *str* hasta que se encuentra el caracter `'\n'`.
`size_t find(const string &str)`	Devuelve el índice de la primer ocurrencia de *str* en la cadena. Si no se encuentra, devuelve `-1`.
`string &insert(size_t pos, const string &str);`	Inserta la cadena *str* dentro de la cadena original a partir de la posición *pos*.
`char &at(size_t pos);`	Devuelve el carácter situado en la posición *pos*. Alternativa: operador `[]`.
`string substr(size_t pos, size_t len);`	Devuelve la cadena que resulta de la composición de *len* caracteres a partir de la posición *pos*.
`void swap(string &str);`	Realiza un intercambio entre los caracteres de la cadena origen y los caracteres de *str*.
`int compare(const string& str);`	Funcionamiento equivalente a `strcmp()` de cadenas tipo C. Alternativa: operadores `==, !=, <, <=, >, >=`.
`string &append(const string& str);`	Concatena *str* al final de la cadena. Alternativa: operador `+`.