**Práctica: Clase String**

**Resolver usando las funciones de la clase string, así como otras funciones de la librería estándar de c++.**

1. **Convertir un objeto string a un array de tipo char, y viceversa.**

void CopiaStringAChar(string& cadena) {

const char\* ptr1 = 0;

ptr1 = cadena.data();

cout << cadena << endl;

}

void CopiaCharAString(char\* ptr1) {

string cadena;

cadena = ptr1;

cout << cadena << endl;

}

1. **¿Cuál es la diferencia entre la función capacity y la función size de la clase string?**

**Size:** Nos muestra el tamaño utilizado de la cadena, es decir, nos dice el tamaño que contiene caracteres.

**Capacity:** Nos muestra el tamaño asignado de la cadena, puede que Capacity sea mayor que Size ya que se le asigna un tamaño adicional para otras operaciones.

Probe en Code Blocks y en Visual, inicialmente cuando la cadena tiene contenido se le asigna Capacity de tamano 15 y cuando pasa eso sube al doble

1. **Convertir las letras de un string de minúsculas a mayúsculas, y las mayúsculas a minúsculas.**

void mayusculas(string a) {

for (int i = 0; i < a.size(); i++) {

a[i] = toupper(a[i]);

}

cout << a << endl;

}

void minusculas(string a) {

for (int i = 0; i < a.size(); i++) {

a[i] = tolower(a[i]);

}

cout << a << endl;

}

1. **Recibir dos números por consola en variables tipo entero, luego construir un objeto string que concatene el primer número seguido del segundo número. Imprimir la cadena resultante, y en una nueva línea su valor multiplicado por 2. Usar funciones para convertir de string a entero y visceversa.**

Ejemplo:

**Input:**

15

135

**Output:**

15135

30270

void enterosAString(int a, int b) {

string c = "";

string d = "";

int e = 0;

c = to\_string(a);

d = to\_string(b);

c += d;

cout << c << endl;

e = atoi(c.c\_str());

e = e \* 2;

cout << e << endl;

}

1. **Crear un objeto string que esté formado por 1000 caracteres ‘a’, escribiendo una sola línea de código.**

void cienString() {

string a(100, 'a');

cout << a << endl;

}

1. **Convertir la cadena: “Yo desaprobe el curso de Algebra Abstracta” a la cadena: “Yo aprobe el curso de Algebra Abstracta”.**

void convertir() {

string a = "Yo desaprobe el curso de Albegra Abstracta";

string b = "";

cout << a << endl;

b = a.replace(3, 9, "aprobe");

cout << b << endl;

}

1. **Recibir dos strings por entrada estándar, una frase y una palabra. Contar cuántas veces aparece la palabra en la frase.**

Ejemplo:

**Input:**

Como poco coco como, poco coco compro.

coco

**Output:**

2

void seRepite(string& frase, string& palabra) {

int posicion = frase.find(palabra);

int i = 0;

while (posicion != string::npos) {

posicion = frase.find(palabra, posicion + 1);

i++;

}

cout << i << endl;

}

1. **Recibir un string de longitud N e imprimir cada letra del string en una nueva línea usando iteradores.**

void iteradores(string& a) {

string::const\_iterator iterador1 = a.begin();

while (iterador1 != a.end()){

cout << \*iterador1;

iterador1++;

}

cout << endl;

}

1. **Recibir dos strings por consola e imprimir un string que tenga al segundo string en la mitad el primer string. El primer string siempre tiene tamaño par.**

Ejemplo:

**Input:**

gato

ti

**Output:**

gatito

void stringEnMedio() {

string a, b;

cout << "Ingrese la primera palabra: ";

getline(cin,a);

cout << "Ingrese la segunda palabra: ";

getline(cin, b);

a.insert((a.size())/2, b);

cout << "La nueva cadena es: " << a << endl;

}

1. **Recibir tres strings por consola, buscar todas las apariciones del segundo string en el primero y reemplazarlas por el tercer string recibido.**

Ejemplo:

**Input:**

Cuando te diga que te creo, no me creas, porque ya no creo lo que creo.

creo

temo

**Output:**

Cuando te diga que te temo, no me creas, porque ya no temo lo que temo.

void stringReemplazo() {

string a, b, c;

cout << "Ingrese la primera palabra: ";

getline(cin, a);

cout << "Ingrese la segunda palabra: ";

getline(cin, b);

cout << "Ingrese la tercer palabra: ";

getline(cin, c);

int posicion = a.find(b);

while (posicion != string::npos) {

a.replace(posicion, 4, c);

posicion = a.find(b, posicion + 1);

}

cout << "Nueva frase: " << a << endl;

}

1. **Dadas dos palabras, imprimir la que es lexicográficamente mayor.**

Ejemplo:

**Input:**

Algebra

Abstracta

**Output:**

Algebra

void lexicograMayor(string& a, string& b) {

if (a > b)

cout << "Es Lexicograficamente mayor: " << a << endl;

if (a < b)

cout << "Es Lexicograficamente mayor: " << b << endl;

else

cout << "Son iguales: " << a << endl;

}

1. **Crear una función que reciba un número N (N va de 1 a 10), y que escriba N números con valores que van de 1 a 1000 en un archivo llamado “numeros.txt”.**

void archivos() {

string target = "F:\\San Pablo\\Algebra Abstracta\\Practica String1\\numeros.txt";

ofstream numeros(target.c\_str());

if (!numeros.is\_open()) {

cout << "No se pudo abrir el archivo " << target << endl;

}

int x;

cout << "ingrese un numero del 1 al 10: ";

cin >> x;

while (x < 1 || x > 10) {

cout << "te dije del 1 al 10: ";

cin >> x;

}

srand((unsigned)time(0));

while (x>0) {

int y = 1 + (rand() % 1000);

numeros << y << endl;

x--;

}

numeros.close();

}

1. **Leer los números en el archivo “numeros.txt” de la pregunta 10 e imprimir la sumatoria.**

void archivosSumatoria() {

string source = "F:\\San Pablo\\Algebra Abstracta\\Practica String1\\numeros.txt";

ifstream numeros(source.c\_str());

if (!numeros.is\_open()) {

cout << "No se pudo abrir el archivo " << source << endl;

}

int x = 0;

int y = 0;

while (!numeros.eof()) {

numeros >> x;

y += x;

}

y -= x;

cout << "La sumatoria es: " << y << endl;

numeros.close();

}

**18.7** (*Cifrado simple*) Cierta información en Internet se puede cifrar con un algoritmo simple conocido como “rot13”, el

cual rota cada carácter 13 posiciones en el alfabeto. Así, 'a' corresponde a 'n', y 'x' corresponde a 'k'. rot13 es un ejemplo

del **cifrado de clave simétrica**. Con este tipo de cifrado, tanto el que cifra como el que descifra utilizan la misma clave.

a) Escriba un programa que cifre un mensaje usando rot13.

b) Escriba un mensaje que descifre el mensaje codifi cado usando 13 como la clave.

c) Después de escribir los programas de los incisos (a) y (b), responda brevemente a la siguiente pregunta: si no conociera

la clave para el inciso (b), ¿qué tan difícil cree usted que sería quebrantar el código? ¿Qué pasaría si tuviera

acceso a un poder de cómputo considerable ( por ejemplo, supercomputadoras)?

string alfabeto = "abcdefghijklmnopqrstuvwxyz ";

int clave = 13;

int tam = alfabeto.size();

string cifrarRot13(string& mensaje) {

int pos\_letra, n;

string result;

for (int i = 0; i < mensaje.size(); i++) {

pos\_letra = alfabeto.find(mensaje[i]);

n = pos\_letra + clave;

while (n >= tam) {

n -= tam;

}

result += alfabeto[n];

}

return result;

}

string decifrarRot13(string& mensaje) {

int pos\_letra, n;

string result;

for (int i = 0; i < mensaje.size(); i++) {

pos\_letra = alfabeto.find(mensaje[i]);

n = pos\_letra - clave;

while (n < 0) {

n += tam;

}

result += alfabeto[n];

}

return result;

}

Dada la poca complejidad del programa es seria muy fácil poder descifrarlo, y más aun teniendo una súper computadora.

**18.11 Escriba un programa que introduzca por separado un primer nombre y un apellido, y que concatene los dos en un nuevo objeto string.**

string nombreApellido() {

string a, b, c;

cout << "Ingrese el nombre: ";

cin >> a;

cout << "Ingrese el apellido: ";

cin >> b;

c += a;

c += ' ';

c += b;

return c;

}

**18.12** Juego del ahorcado

Mis funciona muy bien solo que nadie puede ganar.

void dibujarahorcado(int intentos) {

if (intentos >= 1)

cout << setw(10) << "O" << endl;

else

cout << setw(10) << "" << endl;

if (intentos == 2)

cout << setw(9) << "/" << endl;

if (intentos == 3)

cout << setw(10) << "/|" << endl;

if (intentos >= 4)

cout << setw(11) << "/|\\" << endl;

if (intentos >= 5)

cout << setw(10) << "|" << endl;

if (intentos == 6)

cout << setw(9) << "/" << endl;

if (intentos >= 7)

cout << setw(11) << "/ \\" << endl;

}

bool palabraAdivinar(string adivina, string pais) {

bool ganaste = true;

string s;

for (int i = 0; i < adivina.size(); i++) {

if (pais.find(adivina[i]) == string::npos) {

ganaste = false;

s += "x ";

}

else {

s += adivina[i];

s += " ";

}

}

cout << setw(15) << s;

return ganaste;

}

string palabraAleatoria() {

int contador = 0;

string pais;

vector<string> p;

srand((unsigned)time(0));

string reader = "F:\\San Pablo\\Algebra Abstracta\\Practica String1\\ahorcado.txt";

ifstream ahorcado(reader.c\_str());

if (ahorcado.is\_open()) {

while (std::getline(ahorcado, pais))

p.push\_back(pais);

int paisAleatorio = rand() % p.size();

pais = p.at(paisAleatorio);

ahorcado.close();

}

return pais;

}

int intentosFaltantes(string palabra, string pais) {

int error = 0;

for (int i = 0; i < pais.size(); i++) {

if (palabra.find(pais[i]) == string::npos) {

error++;

}

}

return error;

}

int main()

{

string adivina;

string paisParaAdivinar;

paisParaAdivinar = palabraAleatoria();

int intentos = 0;

bool ganaste = false;

for (; intentos < 8; ) {

system("cls");

cout << paisParaAdivinar << endl;

cout << "Juego del Ahorcado de Paises" << endl;

dibujarahorcado(intentos);

palabraAdivinar(adivina, paisParaAdivinar);

if (ganaste)

break;

intentos++;

char x;

cout << "ingrese una letra: ";

cin >> x;

if (adivina.find(x) == string::npos)

adivina += x;

intentos = intentosFaltantes(paisParaAdivinar, adivina);

}

return 0;

}

**18.19 Escriba un programa que inserte los caracteres "\*\*\*\*\*\*" en la mitad exacta de un objeto string.**

void stringAsteriscosMedio() {

string a;

string b = "\*\*\*\*\*\*";

cout << "Ingrese la primera palabra: ";

getline(cin, a);

a.insert((a.size()) / 2, b);

cout << "La nueva cadena es: " << a << endl;

}

**18.21** Escriba un programa que introduzca una línea de texto, reemplace todos los signos de puntuación con espacios y utilice la función strtok de la biblioteca de cadenas estilo C para dividir el objeto string en palabras individuales (tokens).

string signos = ",./;':-`~!/\|@#$%^&\*()\_+=\*<>?";

void stringReemplazoSignos() {

string a;

cout << "Ingrese la primera palabra: ";

getline(cin, a);

for (int i = 0; i < a.size(); i++) {

for (int j = 0; j < signos.size(); j++) {

if (a[i] == signos[j]) {

a.replace(i, 1, " ");

}

}

}

cout << a << endl;

int longitud = a.size();

char\* str = new char[longitud];

char\* pch;

a.copy(str, longitud, 0);

str[longitud] = '\0';

printf("Splitting string \"%s\" into tokens:\n", str);

pch = strtok(str, " ,.-"); //Error desde aca

while (pch != NULL)

{

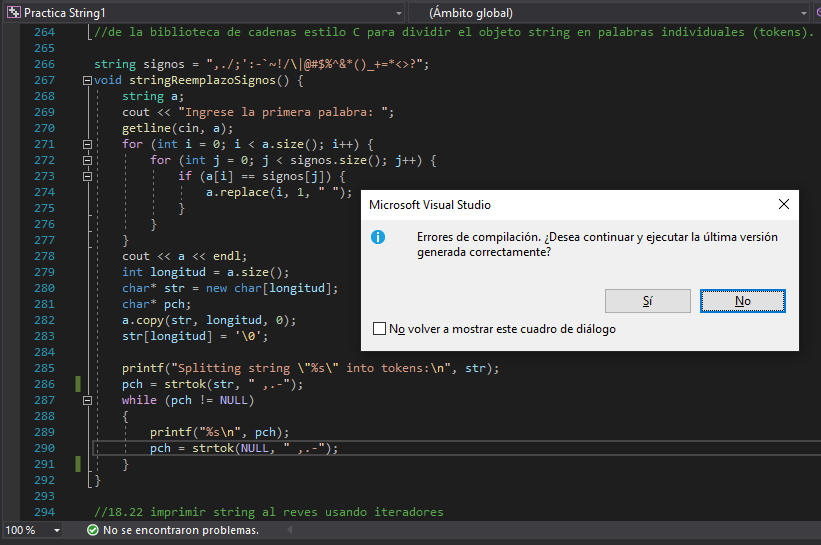
printf("%s\n", pch);

pch = strtok(NULL, " ,.-");

}

}

Miss este programa en Code Block funciona perfectamente, pero en Visual no funciona desde el mensaje de error en amarillo y rojo, como se ve en la flecha roja de la imagen “no se encontraron problemas” pero no lo deja compilar, solo es esa parte del código, pero en Code Blocks si funciona con normalidad.



**18.22** Escriba un programa que introduzca una línea de texto y la imprima al revés. Use iteradores en su solución.

void iteradoresReverso(string& a) {

string::const\_iterator iterador1 = a.end()-1;

while (iterador1 != a.begin()) {

cout << \*iterador1;

iterador1--;

}

cout << \*a.begin();

}