|  |  |  |
| --- | --- | --- |
| **CARRERA:**  Ingeniería De Software | **GUÍA**  No. 01 | **TIEMPO ESTIMADO:**  - |
| **ASIGNATURA:**  Estructura de Datos NRC: 2967 | **FECHA DE ELABORACION:** 27-01-2020  **SEMESTRE**: Septiembre 2019 – Febrero 2020 | |
| **TÍTULO:**  Proyecto Parcial 1   * Jorge Galarza * Kevin Zurita | **DOCENTE:** Ing. Fernando Solís. MsC. | |

**OBJETIVO**

* Realizar un programa que contenga lo aprendido en primer parcial y resuelva un problema utilizando backtracking

**INSTRUCCIONES**

1. Utilice como material principal, aquel indicado en clase por el docente.
2. Utilice información consultada en Internet y conocimiento adquirido en clase.

**ACTIVIDADES**

1. **Ubicación de recursos**

**p.e.**

1. Formar grupos de máximo 2 personas por computador
2. Instalar la herramienta Tiberius
3. **Planteamiento del problema**

**p.e.**

Desarrollar un programa en C++ que contenga todo el contenido aprendido en primer parcial de la materia Estructura de Datos, aplicando backtracking para la resolución del problema.

El programa consiste en buscar una secuencia de palabras tanto en vertical, horizontal y diagonal en una matriz de 10x10 usando backtraking.

**Marco Teórico**

**Backtracking:** Estrategia para encontrar soluciones a problemas que satisfacen restricciones. El término "backtrack" fue acuñado por primera vez por el matemático estadounidense D. H. Lehmer en los años 1950. El objetivo del recorrido es encontrar soluciones para algún problema. Esto se consigue construyendo soluciones parciales a medida que progresa el recorrido; estas soluciones parciales limitan las regiones en las que se puede encontrar una solución completa. El recorrido tiene éxito si, procediendo de esta forma, se puede definir por completo una solución. En este caso el algoritmo puede bien detenerse (si lo único que se necesita es una solución del problema) o bien seguir buscando soluciones alternativas (si deseamos examinarlas todas). Por otra parte, el recorrido no tiene éxito si en alguna etapa la solución parcial construida hasta el momento no se puede completar. En tal caso, el recorrido vuelve atrás exactamente igual que en un recorrido en profundidad, eliminando sobre la marcha los elementos que se hubieran añadido en cada fase. Cuando vuelve a un nodo que tiene uno o más vecinos sin explorar, prosigue el recorrido de una solución.

**Matriz:** Las matrices o como algunos las llaman "arreglos multidimensionales" son una estructura de datos bastante similar a los vectores o arreglos. De hecho, una matriz no es más que una serie de vectores contenidos uno en el otro (u otros), es decir, una matriz es un vector cuyas posiciones son otros vectores.

**Conclusión:**

* Es posible realizar un programa que utilice backtracking para la resolución de un problema en C++, con memoria dinámica se ha realizado la resolución de este problema de una matriz.

**Código:**

//CLASE MAIN

/\*\* **@brief** Proyecto Estructura de Datos Primer Parcial\*/

/\*\* UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE \*

\* Ingenieria de Sofware \*

\* Estructura de Datos \*

\* \*

\* **@author** Jorge Luis Galarza - Kevin Zurita \*

\* NRC:2967 \*

\* **@date** 07/11/2019 \*

\* Ingeniero Fernando Solis \*

\* Version: 1.0

\*/

#include <iostream>

#include "WordMatrix.h"

#include <conio.h>

#include <string.h>

#include <stdio.h>

#include <stdlib.h>

#include <windows.h>

#include <winbgim.h>

#include "Ingresar.h"

void**\*** marquesina**(**void**\*);**

void gotoxy**(**int**,**int**);**

void**\*** proceso**(**void**\*);**

**using** **namespace** std**;**

int main**()**

**{**

//Creacion de hilos

void **\***status**;**

pthread\_t thread1**;**

pthread\_t thread2**;**

pthread\_create **(&**thread1 **,** **NULL** **,** marquesina **,** **(** void **\*)** **&** thread2**);**

pthread\_create **(&**thread2 **,** **NULL** **,** proceso **,** **(** void **\*)** **&** thread1**);**

pthread\_join**(**thread1**,** **&**status**);**

pthread\_join**(**thread2**,** **&**status**);**

system**(**"pause"**);**

**return** 0**;**

**}**

// Proceso

/\*\*

\* **@brief** Funcion en donde se encuentra todo el proceso del proyecto

\* **@param** Void puntero para la creacion de hilos

\* **@return** Void puntero para la creacion de hilos

\*/

void**\*** proceso**(**void **\***data**)**

**{**

Ingreso ing**;**

bool bandera **=** **true;**

int dimension**;**

string palabra**;**

char tecla**;**

system**(**"cls"**);**

cout**<<**endl**;**

cout**<<**endl**;**

cout**<<**endl**;**

cout**<<**"\t Seleccione lo que desea realizar \t\t\t\t Ingrese 5 para ayuda"**<<**endl**;**

cout**<<**"1. Ejercicio Backtracking (Encontrar una palabra dentro de una matriz)"**<<**endl**;**

cout**<<**"2. Codigo QR \t\t\t 3. Codigo de Barras \t\t\t 4. Salir"**<<**endl**;**

**do**

**{**

cin**>>**tecla**;**

**switch(**tecla**){**

**break;**

**case** 49**:** //1

**{**

system**(**"cls"**);**

cout **<<** "\n\n"**<<**endl**;**

cout**<<**"La dimension predeterminada de la matriz es de 10x10"**<<**endl**;**

dimension **=** 10**;**

WordMatrix w **=** WordMatrix**(**dimension**);**

char **\*\*** m**;**

m **=(**char **\*\*)**malloc**(**dimension**\*sizeof(**char **\*));**

**for(**int j**=**0**;**j**<**dimension**;**j**++)**

**\*(**m**+**j**)=(**char **\*)**malloc**(**dimension**\*sizeof(**char**));**

srand**(**time**(NULL));**

**for(**int i**=**0**;**i**<**dimension**;**i**++)**

**for(**int j**=**0**;**j**<**dimension**;**j**++){**

int x **=** **(**rand**()%**26**)+**97**;**

char c**=(**char**)**x**;**

**\*(\*(**m**+**i**)+**j**)=**c**;**

**}**

initwindow**(**650**,**650**);**

settextstyle**(**8**,** 0**,** 4**);**

int x**=**0**;**

**for(**int i**=**55**;** i**<=**600**;**i**+=**60**){**

int y**=**0**;**

**for(**int j**=**55**;**j**<=**600**;**j**+=**60**){**

char**\*** output**;**

output**=(**char **\*)**malloc**(**1**\*sizeof(**char**));**

sprintf**(**output**,**"%c"**,\*(\*(**m**+**x**)+**y**));**

outtextxy**(**j**,**i**,**output**);**

free**(**output**);**

y**++;**

**}**

x**++;**

**}**

rectangle**(**25**,**25**,**625**,**625**);**

**for(**int i**=**85**;**i**<=**625**;**i**+=**60**){**

line**(**i**,**25**,**i**,**625**);**

line**(**25**,**i**,**625**,**i**);**

**}**

palabra**=**ing**.**ingresarString**(**"Ingrese la palabra que desea buscar en la matriz"**);**

closegraph**();**

system**(**"cls"**);**

cout **<<** "\n\n\n"**<<**endl**;**

gotoxy**(**0**,**15**);**

**if** **(**w**.**searchWord**(**m**,** palabra**))** **{**

cout**<<**"Intento acertado (Se saldrá del programa automaticamente)"**<<**endl**;**

exit**(-**1**);**

**}** **else** **{**

cout**<<**"Fallo"**<<**endl**;**

exit**(-**1**);**

**}**

bandera **=** **false;**

**break;**

**}**

**case** 50**:** //2

**{**

system**(**"CodigoQR.png"**);**

**break;**

**}**

**case** 51**:** //3

**{**

system**(**"CodigoDeBarras.png"**);**

**break;**

**}**

**case** 52**:** //4

**{**

system**(**"cls"**);**

cout **<<** "\n"**<<**endl**;**

cout **<<**"Gracias por usar el programa, hasta luego"**<<**endl**;**

exit**(-**1**);**

**break;**

**}**

**case** 53**:**

system**(**"AyudaProyecto1.chm"**);**

**break;**

**default:**

**{**

cout **<<**"\n\n\n\n"**<<**endl**;**

cout **<<**"Opcion no valida"**<<**endl**;**

Beep**(**500**,**100**);**

**break;**

**}**

**}**

**}while(**bandera**);**

**}**

// Marquesina

/\*\*

\* **@brief** Funcion para el desplazamiento de la marquesina

\* **@param** Void puntero para la creacion de hilos

\* **@return** Void puntero para la creacion de hilos

\*/

void**\*** marquesina**(**void **\***data**){**

int a**,** b**,** c**,** n**=**0**,**letra**=**39**,**pos**=**1**,**cont**=**39**,**aux**,**cont1**=**39**,**auxg**;**

char t**[**50**]** **=**"Bienvenidos Proyecto Estructura de Datos"**,**auxt**[**39**]=**" "**;**

**do{**

**for** **(**a**=**0**;**a**<**42**;**a**++){**

aux**=**pos**;**

**for(**b**=**39**;**b**>**cont**;**b**--){**

gotoxy**(**pos**,**1**);**

cout**<<**t**[**b**];**

pos**--;**

**}**

aux**++;**

pos**=**aux**;**

cont**--;**

Sleep **(**75**);**

**if(**a**==**40**){**

**break;**

**}**

**}**

**for(**a**=**3**;**a**<**70**;**a**++){**

gotoxy**(**a**-**1**,**1**);**

cout**<<**" "**;**

gotoxy**(**a**,**1**);**

cout**<<**t**;**

Sleep **(**75**);**

**}**

pos**=**70**;**

auxg**=**69**;**

**for** **(**a**=**0**;**a**<**41**;**a**++){**

gotoxy**(**auxg**,**1**);**

cout**<<**" "**;**

aux**=**pos**;**

**for(**b**=**0**;**b**<=**cont1**;**b**++){**

gotoxy**(**pos**,**1**);**

cout**<<**t**[**b**];**

pos**++;**

**}**

cont1**--;**

aux**++;**

pos**=**aux**;**

auxg**++;**

Sleep **(**75**);**

**}**

cont1**=**39**;**

letra**=**39**;**

pos**=**1**;**

cont**=**39**;**

c**=**0**;**

**}while** **(**c**=**1**);**

**}**

// Funcion gotoxy

/\*\*

\* **@brief** Funcion gotoxy, usado en el desplazamiento de la marquesina

\* **@param** Enteros x e y, que indican la posicion horizontal y vertical de la consola.

\* **@return** Void

\*/

void gotoxy**(**int x**,**int y**){**

HANDLE hcon**;**

hcon **=** GetStdHandle**(**STD\_OUTPUT\_HANDLE**);**

COORD dwPos**;**

dwPos**.**X **=** x**;**

dwPos**.**Y**=** y**;**

SetConsoleCursorPosition**(**hcon**,**dwPos**);**

**}**

//CLASE EXCLUIR

#include <iostream>

#include <string>

#include <bits/stdc++.h>

#include <stdlib.h>

**using** **namespace** std**;**

class Excluir**{**

public**:**

bool control**(**string s**){**

int d**=**0**;**

**for(**int i**=**0**;**i**<**s**.**size**();**i**++)**

**{**

**if(!**isdigit**(**s**[**i**])){**

cout**<<**"Error"**<<**endl**;**

**return** **false;**

**}**

**}**

**return** **true;**

**}**

bool controlS**(**string s**){**

int d**=**0**;**

**for(**int i**=**0**;**i**<**s**.**size**();**i**++)**

**{**

**if(**isdigit**(**s**[**i**])){**

cout**<<**"Error"**<<**endl**;**

**return** **false;**

**}**

**}**

**return** **true;**

**}**

bool controlSN**(**string s**){**

int d**=**0**;**

**for(**int i**=**0**;**i**<**s**.**size**();**i**++)**

**{**

**if(!**isdigit**(**s**[**i**])){**

cout**<<**"Error"**<<**endl**;**

**return** **false;**

**}**

**}**

**return** **true;**

**}**

**};**

// CLASE INGRESAR

#include <iostream>

#include "Excluir.h"

#include <string>

**using** **namespace** std**;**

//Universidad de las Fuerzas Armdas

//Nombre: Kevin Zurita

//NRC: 2967

//Docente: Ing. Edgar Solis

//Fecha de creacion: 23/09/2019

//FecHa de ultima modificacion: 1/10/2019

class Ingreso**{**

public**:**

string ingresarString**(**char**\*** **);**

int ingresar**(**char**\*** **);**

string ingresarStringN**(**char**\*** **);**

**};**

string Ingreso**::**ingresarString**(**char **\***msg**){**

string valor**;**

bool flag**;**

Excluir comprobante**;**

cout**<<**msg**<<**endl**;**

**do{**

cin**>>**valor**;**

flag**=**comprobante**.**controlS**(**valor**);**

**}while(**flag**==false);**

**return** valor**;**

**}**

string Ingreso**::**ingresarStringN**(**char **\***msg**){**

string valor**;**

bool flag**;**

Excluir comprobante**;**

cout**<<**msg**<<**endl**;**

**do{**

cin**>>**valor**;**

flag**=**comprobante**.**controlSN**(**valor**);**

**}while(**flag**==false);**

**return** valor**;**

**}**

int Ingreso**::**ingresar**(**char **\***msg**){**

string valor**;**

bool flag**;**

Excluir comprobante**;**

cout**<<**msg**<<**endl**;**

**do{**

cin**>>**valor**;**

flag**=**comprobante**.**control**(**valor**);**

**}while(**flag**==false);**

int num**;**

num **=** strtof**((**valor**).**c\_str**(),**0**);**

**return** num**;**

**}**