|  |  |  |
| --- | --- | --- |
| **CARRERA:**  Ingeniería de Software | **GUÍA**  No. 02 | **TIEMPO ESTIMADO:**  1h y 20 min. |
| **ASIGNATURA:**  Estructura de Datos  NRC: 2967 | **FECHA DE ELABORACION:** 26/01/2020  **SEMESTRE**: septiembre 2019 – febrero 2020 | |
| **TÍTULO:**  Tetris (Listas) | **DOCENTE:** Ing. Fernando Solís | |

**OBJETIVO**

Creación e implementación de listas circulares doblemente enlazadas en un programa C++ a través del juego TETRIS.

**INSTRUCCIONES**

1. Utilice como material principal, aquel indicado en clase por el docente.
2. Utilice información consultada en Internet y conocimiento adquirido en clase.
3. Realizar un Tetris con las validaciones válidas de todo el juego
4. Cuando el número cae implementarlo a una lista
5. Si el número se repite se eliminará

**ACTIVIDADES**

1. **Ubicación de recursos**
2. Formar grupos de máximo 2 personas por computador
3. Instalar la herramienta C++
4. Investigar temas por cuenta propia
5. **Planteamiento del problema**

**Ejercicio No 1**

Escribir un programa el cual simule un tetris que será manejado por números que se podrán insertar entre ,al final o inicio de la lista la mismas que será generada en un archivo pdf, hay que recordar que hay que utilizar backtraking en el mismo. Si existen dos números seguidos repetidos se procederá a eliminarlo de la parte gráfica y de la lista circular doblemente enlazada.

Funciones C++ a utilizar:

* **kbhit() -** Revisa si una tecla pulsada está disponible
* **getch() -** Lee un solo carácter directamente desde el teclado, sin mostrar tal carácter en pantalla.
* **rand() -** Se utiliza para obtener un [número aleatorio](https://es.wikipedia.org/wiki/N%C3%BAmero_aleatorio).
* **system() -** Sirve para ejecutar subprocesos o comandos del sistema operativo.
* **pthread\_mutex\_lock() -** Permite solicitar acceso al mutex, el hilo se bloquea hasata su obtención
* **pthread\_mutex\_unlock()** - Permute liberar un mutex.

**Marco Teórico**

**Tetris**

Es un [videojuego de puzzle](https://es.wikipedia.org/wiki/Videojuego_de_puzzle) originalmente diseñado y programado por [Alekséi Pázhitnov](https://es.wikipedia.org/wiki/Aleks%C3%A9i_P%C3%A1zhitnov" \o "Alekséi Pázhitnov) en la [Unión Soviética](https://es.wikipedia.org/wiki/Uni%C3%B3n_Sovi%C3%A9tica). Fue lanzado el 6 de junio de 1984, mientras trabajaba para el [Centro de Computación Dorodnitsyn](https://es.wikipedia.org/w/index.php?title=Centro_de_Computaci%C3%B3n_Dorodnitsyn&action=edit&redlink=1) de la [Academia de Ciencias de la Unión Soviética](https://es.wikipedia.org/wiki/Academia_de_las_Ciencias_de_Rusia) en [Moscú](https://es.wikipedia.org/wiki/Mosc%C3%BA), [RSFS de Rusia](https://es.wikipedia.org/wiki/RSFS_de_Rusia). Su nombre deriva del prefijo numérico griego tetra- y del [tenis](https://es.wikipedia.org/wiki/Tenis), el deporte favorito de Pázhitnov.

En el Tetris se juega con los [tetrominós](https://es.wikipedia.org/wiki/Tetromin%C3%B3" \o "Tetrominó), el caso especial de cuatro elementos de [poliominós](https://es.wikipedia.org/wiki/Poliomin%C3%B3" \o "Poliominó). Los poliominós se han utilizado en los rompecabezas populares por lo menos desde 1907, y el nombre fue dado por el matemático Solomon W. Golomb en 1953. Sin embargo, incluso la enumeración de los pentominós data de la antigüedad.

El juego está disponible para casi cada [consola de videojuegos](https://es.wikipedia.org/wiki/Videoconsola) y [sistemas operativos](https://es.wikipedia.org/wiki/Sistema_operativo) de PC, así como en dispositivos tales como las [calculadoras gráficas](https://es.wikipedia.org/wiki/Calculadora_gr%C3%A1fica), [teléfonos móviles](https://es.wikipedia.org/wiki/Telefon%C3%ADa_m%C3%B3vil), [reproductores de multimedia portátiles](https://es.wikipedia.org/wiki/Reproductor_multimedia_digital), [PDAs](https://es.wikipedia.org/wiki/PDA" \o "PDA), reproductores de música en red e incluso como [huevo de pascua](https://es.wikipedia.org/wiki/Huevo_de_pascua_(virtual)) en productos no mediáticos como los [osciloscopios](https://es.wikipedia.org/wiki/Osciloscopio). También ha inspirado servicios de mesa y ha sido jugado en los costados de varios edificios, manteniendo el récord de ser el juego completamente funcional más grande del mundo gracias al esfuerzo de estudiantes holandeses en 1995 que iluminaron quince pisos del Departamento de Ingeniería Eléctrica en la [Universidad Técnica de Delft](https://es.wikipedia.org/wiki/Universidad_T%C3%A9cnica_de_Delft).

Aunque diferentes versiones de Tetris se habían vendido para una amplia gama de plataformas de [ordenadores domésticos](https://es.wikipedia.org/wiki/Ordenador_dom%C3%A9stico) y [arcades](https://es.wikipedia.org/wiki/Arcade) durante los años 1980, fue la inmensamente exitosa versión portátil para la [Game Boy](https://es.wikipedia.org/wiki/Tetris_(Game_Boy)" \o "Tetris (Game Boy)) lanzada en 1989 la que lo convirtió en uno de los juegos más populares de todos los tiempos. La edición número 100 del [Electronic Gaming Monthly](https://es.wikipedia.org/wiki/Electronic_Gaming_Monthly) otorgó a Tetris el número 1 en el ranking de **"Mejores juegos de todos los tiempos"**. En 2007, Tetris ocupó el segundo lugar en los **«100 mejores videojuegos de todos los tiempos»** para [IGN](https://es.wikipedia.org/wiki/IGN_(sitio_web)). Ha vendido más de ciento setenta millones de copias para el año 2016. En enero de 2010, se anunció que el Tetris había vendido más de cien millones de unidades para teléfonos móviles desde el año 2005.

**Listas circulares doblemente enlazadas**

Una lista doble circular es una estructura donde el último elemento tiene como referencia siguiente al primer elemento y la referencia al anterior del primer elemento de la lista también es el último.

Es una especie de lista enlazada “doblemente enlazada”, pero que posee una característica adicional para el desplazamiento dentro de la lista, “ésta no tiene fin” y tiene 2 apuntadores a sí misma.

A través del uso de listas dobles podemos acceder a los datos recorriendo los hacia delante hasta el final o hacia atrás hasta el inicio.

El primer paso es crear un nodo para el dato que vamos a insertar.

Si Lista está vacía, o el valor del primer elemento de la lista es mayor que el del nuevo, insertaremos el nuevo nodo en la primera posición de la lista.

A veces las listas enlazadas tienen un nodo centinela al principio y/o al final de la lista, el cual no es usado para guardar datos. Su propósito es simplificar o agilizar algunas operaciones, asegurando que cualquier nodo tiene otro anterior o posterior, y que toda la lista siempre tenga un “primer y último” nodo.

Para mostrar la lista completa, es necesario posicionarse al inicio de la lista. Luego, utilizando el puntero siguiente de cada elemento, la lista es recorrida del 1er al último elemento. Mostrar la lista sin una condición para detenerse.

**Código**

**Tetris.h**

1. #include <conio.h>
2. #include <fstream>
3. #include <windows.h>
5. #define IZQUIERDA 75
6. #define DERECHA 77
7. #define ESC 27
9. using namespace std;
11. pthread\_mutex\_t ptmutex1;
12. ofstream archivo;
14. class Tetris {
15. private:
16. char tecla;
17. public:
18. void guardarPosicion();
19. int moverNumero();
20. void borrarNumero();
21. void teclear();
22. bool posicion();
23. void mantener();
24. void moverAbajo();
25. void ubicar(int);
26. void principal();
27. void crearMenu();
28. };
29. Tetris.cpp
30. #include "Marquesina.cpp"
31. #include "Tetris.h"
32. #include "VectorMatriz.cpp"
33. #include "Numero.cpp"
34. #include "Cifrado.cpp"
36. static HWND  hConWnd;
37. HWND BCX\_Bitmap(char\*, HWND = 0, int = 0, int = 0, int = 0, int = 0, int = 0, int = 0, int = 0, int = 0);
38. HWND GetConsoleWndHandle(void);
39. void Tetris::guardarPosicion() {
40. \*(\*(cuerpo+n)+0) = x;
41. \*(\*(cuerpo+n)+1) = y;
42. n++;
43. if(n == tamanio) {
44. n = 1;
45. }
46. }
47. int Tetris::moverNumero() {
48. Pantalla pantalla;
49. for(int i=1; i<tamanio; i++) {
50. pantalla.gotoxy(\*(\*(cuerpo+i)+0), \*(\*(cuerpo+i)+1));
51. cout << numero;
52. }
53. return numero;
54. }
56. void Tetris::borrarNumero() {
57. Pantalla pantalla;
58. pantalla.gotoxy(\*(\*(cuerpo+n)+0), \*(\*(cuerpo+n)+1));
59. cout <<" ";
60. }
62. void Tetris::teclear() {
63. if(kbhit()) {
64. tecla = getch();
65. switch(tecla) {
66. case DERECHA:
67. if(direccion != 4) {
68. direccion=3;
69. }
70. break;
71. case IZQUIERDA:
72. if(direccion != 3) {
73. direccion=4;
74. }
75. break;
76. }
77. }
78. }
80. bool Tetris::posicion() {
81. Pantalla pantalla;
82. ubicar(mantenerY);
83. }
84. void Tetris::mantener() {
85. Pantalla pantalla;
86. pantalla.gotoxy(0,0);
87. }
88. void Tetris::moverAbajo() {
89. if(direccion != 1) {
90. direccion=2;
91. }
92. }
93. void Tetris::ubicar(int auxiliarY) {
94. Numero \*lista = new Numero;
95. Pantalla pantalla;
96. auxiliarY = mantenerY;
97. if(y == auxiliarY) {
98. pantalla.gotoxy(x-1,y-1);
99. cout << numero;
100. archivo << numero << endl;
101. pantalla.gotoxy(auxiliar,10);
102. lista->insertarEntre(x,numero);
103. lista->imprimir();
104. numero = (rand()%9)+1;
105. numero = moverNumero();
106. y = 1;
107. auxiliar++;
108. if(auxiliar == 167) {
109. auxiliar = 1;
110. aumenta++;
111. mantenerY--;
112. }
113. }
114. if(x==0 || x ==167) {
115. while(tecla != ESC) {
116. Pantalla pantalla;
117. pantalla.gotoxy(80,20);
118. cout << "Fin del Juego";
119. system("pause");
120. tecla = ESC;
121. }
122. }
123. }
125. void Tetris::principal() {
126. Pantalla pantalla;
127. VectorMatriz generar;
128. cuerpo = generar.inicializar(100);
129. cuerpo = generar.encerar(100,cuerpo);
130. pantalla.pintar();
131. while(tecla != ESC && posicion()) {
132. borrarNumero();
133. guardarPosicion();
134. moverNumero();
135. mantener();
136. moverAbajo();
137. teclear();
138. if(direccion == 2) {
139. y++;
140. Sleep(10);
141. }
142. if(direccion == 3) {
143. x++;
144. Sleep(10);
145. }
146. if(direccion == 4) {
147. x--;
148. Sleep(10);
149. }
150. Sleep(10);
151. }
152. pantalla.pintar();
153. }
154. void\* funcion1(void \*arg) {
155. Tetris tetris;
156. pthread\_mutex\_lock(&ptmutex1);
157. Sleep(100);
158. tetris.crearMenu();
159. pthread\_mutex\_unlock(&ptmutex1);
160. return NULL;
161. }
162. void\* funcion2(void \*arg) {
163. Marquesina marquesina;
164. pthread\_mutex\_lock(&ptmutex1);
165. //marquesina.hacerMarquesina();
166. //Sleep(5000);
167. pthread\_mutex\_unlock(&ptmutex1);
168. return NULL;
169. }
171. void\* funcion3(void \*arg) {
172. pthread\_mutex\_lock(&ptmutex1);
173. hConWnd = GetConsoleWndHandle();
174. //Sleep(5000);
175. //BCX\_Bitmap("logo.bmp", hConWnd, 123, 600, 400, 0, 0);
176. //Sleep(5000);
177. pthread\_mutex\_unlock(&ptmutex1);
178. return NULL;
179. }
181. void Tetris::crearMenu() {
182. Cifrado cifrar; //objeto para manejar el proceso
183. char \*nombreArchivo = (char\*)malloc(30\*sizeof(char)); //direccion del archivo
184. char \*claveArchivo = (char\*)malloc(30\*sizeof(char));
185. \*(claveArchivo+0)=0;
186. Pantalla pantalla;
187. pantalla.abrirPantalla();
188. int opcion;
189. const char \*opciones[]={"1. Jugar","2. Abrir QR","3. Abrir codigo de barras","4. Abrir Ayuda","5. Seleccionar Archivo","6. Encriptar","7. Desencriptar",
190. "8. Salir"};
191. opcion=pantalla.menu("TETRIS",opciones,8);
192. bool bandera=false;
193. do {
194. switch(opcion) {
195. case 1: {
196. system("cls");
197. principal();
198. muestraPDF();
199. opcion=pantalla.menu("TETRIS",opciones,8);
200. archivo.close();
201. break;
202. }
203. case 2:
204. system("cls");
205. system("QR.png");
206. opcion=pantalla.menu("TETRIS",opciones,8);
207. break;
209. case 3:
210. system("cls");
211. system("barra.jpeg");
212. opcion=pantalla.menu("TETRIS",opciones,8);
213. break;
215. case 4:
216. system("cls");
217. system("Arimaa.chm");
218. opcion=pantalla.menu("TETRIS",opciones,8);
219. break;
221. case 5:
222. system("cls");
223. cout << "Nombre del archivo: ";
224. cin >> nombreArchivo;
225. //si la cadena esta vacia no hace nada
226. if(\*(nombreArchivo+0)==0 )break;
227. cifrar.leer(nombreArchivo);
228. opcion=pantalla.menu("TETRIS",opciones,8);
229. break;
231. case 6:
232. system("cls");
233. //si la cadena esta vacia no hace nada
234. if(\*(nombreArchivo+0)==0 )break;
235. cout << "Ingrese la clave: ";
236. cin >> claveArchivo;
237. //si la clave esta vacia no hace nada
238. if(\*(claveArchivo+0)==0 )break;
239. cifrar.cifrar(claveArchivo ); //cifra
240. cifrar.grabar(nombreArchivo ); //guarda
241. opcion=pantalla.menu("TETRIS",opciones,8);
242. break;
244. case 7:
245. system("cls");
246. //si la cadena esta vacia no hace nada
247. if(\*(nombreArchivo)==0 )break;
248. cout << "Ingrese la clave: ";
249. cin >> claveArchivo;
250. //si la clave esta vacia no hace nada
251. if(\*(claveArchivo)==0 )break;
252. cifrar.descifrar(claveArchivo ); //descifra
253. cifrar.grabar(nombreArchivo ); //guarda
254. opcion=pantalla.menu("TETRIS",opciones,8);
255. break;
257. case 8:
258. exit(0);
259. break;
261. default:
262. system("cls");
263. cout << endl << endl << endl << endl << "Opcion no valida.**\a\n**";
264. break;
265. }
266. }while(bandera!=true);
267. }
269. HWND BCX\_Bitmap(char\* Text, HWND hWnd, int id, int X, int Y, int W, int H, int Res, int Style, int Exstyle)
270. {
271. HWND A;
272. HBITMAP hBitmap;
274. // set default style
275. if (!Style) Style = WS\_CLIPSIBLINGS | WS\_CHILD | WS\_VISIBLE | SS\_BITMAP | WS\_TABSTOP;
277. // form for the image
278. A = CreateWindowEx(Exstyle, "static", NULL, Style, X, Y, 0, 0, hWnd, (HMENU)id, GetModuleHandle(0), NULL);
280. // Text contains filename
281. hBitmap = (HBITMAP)LoadImage(0, Text, IMAGE\_BITMAP, 0, 0, LR\_LOADFROMFILE | LR\_CREATEDIBSECTION);
283. // auto-adjust width and height
284. if (W || H) hBitmap = (HBITMAP)CopyImage(hBitmap, IMAGE\_BITMAP, W, H, LR\_COPYRETURNORG);
285. SendMessage(A, (UINT)STM\_SETIMAGE, (WPARAM)IMAGE\_BITMAP, (LPARAM)hBitmap);
286. if (W || H) SetWindowPos(A, HWND\_TOP, X, Y, W, H, SWP\_DRAWFRAME);
287. return A;
288. }
290. HWND GetConsoleWndHandle(void)
291. {
292. HWND hConWnd;
293. OSVERSIONINFO os;
294. char szTempTitle[64], szClassName[128], szOriginalTitle[1024];
296. os.dwOSVersionInfoSize = sizeof(OSVERSIONINFO);
297. GetVersionEx(&os);
298. // may not work on WIN9x
299. if (os.dwPlatformId == VER\_PLATFORM\_WIN32s) return 0;
301. GetConsoleTitle(szOriginalTitle, sizeof(szOriginalTitle));
302. sprintf(szTempTitle, "%u - %u", GetTickCount(), GetCurrentProcessId());
303. SetConsoleTitle(szTempTitle);
304. Sleep(60);
305. // handle for NT and XP
306. hConWnd = FindWindow(NULL, szTempTitle);
307. SetConsoleTitle(szOriginalTitle);
309. // may not work on WIN9x
311. if (os.dwPlatformId == VER\_PLATFORM\_WIN32\_WINDOWS)
312. {
313. hConWnd = GetWindow(hConWnd, GW\_CHILD);
314. if (hConWnd == NULL) return 0;
315. GetClassName(hConWnd, szClassName, sizeof(szClassName));
316. // while ( \_stricmp( szClassName, "ttyGrab" ) != 0 )
317. while (strcmp(szClassName, "ttyGrab") != 0)
318. {
319. hConWnd = GetNextWindow(hConWnd, GW\_HWNDNEXT);
320. if (hConWnd == NULL) return 0;
321. GetClassName(hConWnd, szClassName, sizeof(szClassName));
322. }
323. }
324. return hConWnd;
325. }

**Conclusión**

* Se desarrolló el programa de Tetris que almacenaba los números en una lista circular doblemente enlazada.
* Para realizar el programa fue necesario realizar investigaciones de mayor profundidad sobre el juego y la implementación de funciones básicas de la lista.
* Se tuvo que realizar las respectivas validaciones para obtener un óptimo funcionamiento de las listas.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DOCENTE RESPONSABLE Integrantes del Grupo

Ing. Fernando Solis. MsC. Carlos Puco, Kevin Duy