

UNIVERSIDAD DE GUADALAJARA

CUCEI

DIVISIÓN DE TECNOLOGÍAS PARA LA INTEGRACIÓN CIBER-HUMANA

DEPARTAMENTO DE CIENCIAS COMPUTACIONALES

PRÁCTICA No. 3

TEMA: Archivos de Acceso Aleatorio (Consultas individuales y Consultas Generales)

EQUIPO No. 4

INTEGRANTES:

Munguía Guízar Marlon Uriel

Padilla Perez Jorge Daray

Pérez Palacios Perla Michelle

Pulido Tobías Rafael Agustín

SEMINARIO DE SOLUCIÓN DE PROBLEMAS DE ESTRUCTURAS DE DATOS II | SECCIÓN D19 | 2022B

PROF. Mariscal Lugo Luis Felipe

**MARCO TEÓRICO**

* **¿Para qué sirve la función read( )?**

La función miembro read introduce cierto número de bytes en un arreglo de caracteres en la memoria. Si se leen menos caracteres que el número designado, se establece el bit failbit.

Usa los siguientes parámetros:

fd: Descriptor de fichero, creado en una llamada previa open, del cual se leerán los datos.

buf: Puntero al buffer de caracteres donde se recibirán los datos leídos.

count: Número máximo de bytes que se van a almacenar en el buffer de caracteres.

La llamada al sistema read se realiza desde un proceso de usuario y permite la lectura de datos de un archivo. Esta llamada lee los datos de los dispositivos de bloques a la caché del núcleo para luego copiarlos al espacio de direcciones del proceso. La llamada al sistema read devuelve el número de bytes que se han podido leer del archivo y en caso de error devuelve -1. Si esto último ocurre, se almacena el tipo de error en la variable errno.

Estos pueden ser, por ejemplo:

EBADF: El descriptor de fichero especificado no es válido.

EFAULT: buf contiene una dirección no válida.

EINVAL: fd se refiere a un objeto sobre el que no es posible una lectura.

**Bibliografía.**

Deitel., H. M. & Deitel., P. J. (2008). *C++: cómo programar* (6.a ed.). Pearson.

ULPGC. (s.f). Llamadas al sistema read and write. 16/09/22, de ULPGC. Sitio web: http://sopa.dis.ulpgc.es/iidso/leclinux/fs/readwrite/LEC20\_READWRITE.pdf

**Código:**

**Main:**

1 /\*Equipo 4:

2 Munguia Guizar Marlon Uriel

3 Padilla Perez Jorge Daray

4 Perez Palacios Perla Michelle

5 Pulido Tobias Rafael Agustin

6 Seccion: D19

7 Calendario: 2022B

8 \*/

9 #include <iostream>

10 #include <iomanip>

11 #include <fstream>

12 #include <cstdlib>

13 #include "../../tareas\_entregar/Posible\_practica2/include/deportista.h"

14

15 **using namespace std**;

16

17 **void** imprimirLinea( ostream&, **const** Deportista & ); // prototipo

18

19 **int** main()

20 {

21 **int** opc = 0;

22 Deportista d;//objeto o registro Deportista

23 **ifstream** archdeportEntrada; //archivo logico

24 archdeportEntrada.open ("deportistas.dat", ios::in | ios::binary);

25 //ifstream archdeportEntrada( "deportistas.dat", ios::in | ios::binary );//el constructor abre el archivo

26

27 // sale del programa si ifstream no puede abrir el archivo

28 **if** ( !archdeportEntrada)

29 {

30 **cerr** << "No se pudo abrir el archivo." << **endl**;

31 exit( 1 );

32 } // fin de if

33 **do**{

34 system("cls");

35 **cout** << "1) Consulta individual (numero de socio)" << **endl**;

36 **cout** << "2) Consultas generales" << **endl**;

37 **cout** << "3) Salir" << **endl**;

38 **cout** << "Ingresar opcion: "<<**endl**;

39 **cin**>> opc;

40 archdeportEntrada.**clear**();

41 archdeportEntrada.seekg(0, archdeportEntrada.beg);

42 **switch**(opc)

43 {

44 **case** 1:

45 {

46 system("cls");

47 **cout** << "Consulta individual" << **endl**;

48 **int** numero;

49 **cout** << "Dame el numero de socio: "<<**endl**;

50 **cin** >> numero;

51 d.establecerNumeroDeSocio(numero);

52 **cout** << left << **setw**( 10 ) << "Numero de socio" << **setw**( 16 )

53 << "Nombre" << **setw**( 11 ) << "Edad" << left

54 << **setw**( 10 ) << right << "Deporte" << **endl**;

55 archdeportEntrada.seekg((d.obtenerNumeroDeSocio()-1)\***sizeof**(Deportista),ios::beg); //seekp(n, ios::beg);

56 // lee el siguiente registro del archivo

57 archdeportEntrada.read( **reinterpret\_cast**< **char** \* >( &d ), **sizeof**( Deportista ) );

58 **if** ( d.obtenerNumeroDeSocio() != 0)

59 {

60 imprimirLinea( **cout**, d);

61 }

62 **else**

63 {

64 **cout** << "EL NUMERO DE SOCIO NO EXISTE"<<**endl**;

65 }

66 system("pause");

67 **break**;

68 }

69 **case** 2:

70 {

71 system("cls");

72 **cout** << "Consulta general" << **endl**;

73 **cout** << left << **setw**( 10 ) << "Numero de socio" << **setw**( 16 )

74 << "Nombre" << **setw**( 11 ) << "Edad" << left

75 << **setw**( 10 ) << right << "Deporte" << **endl**;

76 // lee el primer registro del archivo

77 archdeportEntrada.read( **reinterpret\_cast**< **char** \* >( &d ), **sizeof**( Deportista ) );

78 // lee todos los registros del archivo

79 **while** ( archdeportEntrada && !archdeportEntrada.eof() )

80 {

81 // muestra un registro

82 **if** ( d.obtenerNumeroDeSocio() != 0)

83 {

84 imprimirLinea( **cout**, d);

85 }

86 // lee el siguiente registro del archivo

87 archdeportEntrada.read( **reinterpret\_cast**< **char** \* >( &d ), **sizeof**( Deportista ) );

88 } // fin de while

89 system("pause");

90 }

91 **break**;

92 }

93 }**while**(opc != 3);

94 **cout** << "Saliendo..."<<**endl**;

95 **return** 0;

96 } // fin de main

97

98 // muestra un solo registro

99 **void** imprimirLinea( ostream &salida, **const** Deportista &registro )

100 {

101 salida << left << **setw**( 10 ) << registro.obtenerNumeroDeSocio()

102 << **setw**( 16 ) << registro.obtenerNombre()

103 << **setw**( 11 ) << registro.obtenerEdad()

104 << **setw**( 10 ) << **setprecision**( 2 ) << right << fixed

105 << showpoint << registro.obtenerDeporte() << **endl**;

106 } // fin de la funciÃ³n imprimirLinea

**Deportista.h:**

1 /\*Equipo 4:

2 MunguÃ­a GuÃ­zar Marlon Uriel

3 Padilla Perez Jorge Daray

4 PÃ©rez Palacios Perla Michelle

5 Pulido TobÃ­as Rafael AgustÃ­n

6 SecciÃ³n: D19

7 Calendario: 2022B

8 \*/

9 #ifndef DEPORTISTA\_H

10 #define DEPORTISTA\_H

11 #include <cstring>

12 #include <string>

13

14 **using namespace std**;

15 **using std**::**string**;

16

17

18 **class** Deportista

19 {

20 **public**:

21 //constructor predeterminado

22 Deportista(**int** = 0, **string** = "", **int** = 0, **string** = "");

23

24 //funciones de acceso para NumeroDeSocio

25 **void** establecerNumeroDeSocio(**int**);

26 **int** obtenerNumeroDeSocio() **const**;

27

28 //funciones de acceso para nombre

29 **void** establecerNombre(**string**);

30 **string** obtenerNombre() **const**;

31

32 //funciones de acceso para edad

33 **void** establecerEdad(**int**);

34 **int** obtenerEdad() **const**;

35

36 //funciones de acceso para deporte

37 **void** establecerDeporte(**string**);

38 **string** obtenerDeporte() **const**;

39

40

41 **protected**:

42

43 **private**:

44 **int** numeroDeSocio;

45 **char** nombre[15];

46 **int** edad;

47 **char** deporte[15];

48

49 };//Fin de la clase Deportista

50

51 #endif // DEPORTISTA\_H

**DeportistaCPP:**

1 /\*Equipo 4:

2 MunguÃ­a GuÃ­zar Marlon Uriel

3 Padilla Perez Jorge Daray

4 PÃ©rez Palacios Perla Michelle

5 Pulido TobÃ­as Rafael AgustÃ­n

6 SecciÃ³n: D19

7 Calendario: 2022B

8 \*/

9 #include "Deportista.h"

10 #include <cstring>

11

12 **using namespace std**;

13

14 //Constructor

15 Deportista::Deportista(**int** valorNumeroDeSocio, **string** valorNombre, **int** valorEdad, **string** valorDeporte)

16 {

17 //ctor

18 establecerNumeroDeSocio(valorNumeroDeSocio);

19 establecerNombre(valorNombre);

20 establecerEdad(valorEdad);

21 establecerDeporte(valorDeporte);

22 }//fin constructor

23

24 //Obtiene el valor del nÃºmero de socio

25 **int** Deportista::obtenerNumeroDeSocio() **const**{

26 **return** numeroDeSocio;

27 }//fin int obtenerNumeroDeSocio

28

29 //Establece el valor del nÃºmero de Socio

30 **void** Deportista::establecerNumeroDeSocio(**int** valorNumeroDeSocio){

31 numeroDeSocio=valorNumeroDeSocio;

32 }//fin void establecerNumeroDeSocio

33

34 //Obtiene el valor del nombre

35 **string** Deportista::obtenerNombre() **const**{

36 **return** nombre;

37 }//fin string obtenerNombre

38

39 //Establece el valor del nombre

40 **void** Deportista::establecerNombre(**string** cadenaNombre){

41 **const char**\*valorNombre = cadenaNombre.data();

42 **int** longitud = cadenaNombre.**size**();

43 longitud = (longitud < 15 ? longitud : 14);

44 strncpy(nombre, valorNombre, longitud);

45 nombre[longitud] = '\0';

46 }//fin void establecerNombre

47

48 //Obtiene el valor de la edad

49 **int** Deportista::obtenerEdad() **const**{

50 **return** edad;

51 }//fin int obtenerEdad

52

53 //Establece el valor de la edad

54 **void** Deportista::establecerEdad(**int** valorEdad){

55 edad=valorEdad;

56 }//fin void establecerEdad

57

58

59 //Obtiene el valor del deporte

60 **string** Deportista::obtenerDeporte() **const**{

61 **return** deporte;

62 }//fin string obtenerDeporte

63

64 //Establece el valor del deporte

65 **void** Deportista::establecerDeporte(**string** cadenaDeporte){

66 **const char**\*valorDeporte = cadenaDeporte.data();

67 **int** longitud = cadenaDeporte.**size**();

68 longitud = (longitud < 15 ? longitud : 14);

69 strncpy(deporte, valorDeporte, longitud);

70 deporte[longitud] = '\0';

71 }//fin void establecerDeporte

**imágenes del archivo físico:**

Tabla

Descripción generada automáticamente con confianza media

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

**Capturas:**

Menu del programa:

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

Opción 2) consulta general:

Texto

Descripción generada automáticamente

Opción 1) numero de socio ingresado ‘1’:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Opción 1) número de socio ingresado ‘5’:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza media

Opción 1) número de socio ingresado ‘80’:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Opción 1) número de socio ingresado ‘90’:

Interfaz de usuario gráfica

Descripción generada automáticamente

Opción 1) número de socio ingresado ‘70’:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Opción 1) número de socio ingresado ‘11’:

Interfaz de usuario gráfica

Descripción generada automáticamente

Opción 3) Salir :

Texto

Descripción generada automáticamente

**Conclusiones:**

* **Munguía Guízar Marlon Uriel.**
  + Dentro de lo que cabe esta práctica fue sencilla gracias a la ayuda del profesor y del libro de Deitel, ya que con él puedes llevar una guía para solucionar errores, además de la gran cooperación de parte de mis compañeros de equipo, con los cuales se vuelve mucho más ameno hacer este tipo de actividades.
* **Padilla Perez Jorge Daray.**
  + En conclusión la practica estuvo interesante, para mi un poco complicada por el hecho de tener que usar lo aprendido anteriormente pero con la función read(), además de tener que hacer consultas, importante también que el apuntador este al inicio en este caso.
* **Pérez Palacios Perla Michelle.**
  + En esta práctica pudimos ver los métodos seekg y read, cómo son muy útiles para la escritura y lectura de los archivos de texto en los que hemos estado trabajando.
* **Pulido Tobías Rafael Agustín.**
  + El uso y entendimiento de la función read() me pareció sencillo y útil, además el uso de seekg() también es bastante útil a la hora de buscar o consultar algún registro en específico, gracias a esto el entendimiento de este tema me pareció mas sencillo.