**Universidad de oriente**

**Núcleo de Anzoátegui**

**Escuela de Ingeniería y Ciencias Aplicadas**

**Departamento de Computación y Sistemas**

**Asignatura: Estructura de datos**

**Profesor: Grupo:**

**Luis Correa CI: 19.840.230**

**Frank Rondón CI: 19.738.854**

**Manuel Lara**

**Sección: 01**

**Barcelona, 27 de Agosto del 2012**

## Desarrollo

## 1.- Diferencia entre un archivo de texto y un binario:

La diferencia está en la forma en que se almacenaran los datos y se accederá a ellos, ya que en un archivo de texto podrás escribir solo texto para luego leerlos, en cambio un archivo binario podrá escribir cualquier tipo de dato pero tendrá que especificar el espacio que va a ocupar en el archivo y luego para leerlo tendrá que especificar cual espacio del archivo va a leer.

## 2.- ¿Por qué existen los archivos binarios y de texto?:

Primero son diferentes ambos las ventajas de cada uno son:

* Archivos binarios:

En los archivos binarios no se traducen las secuencias de retorno de carro, caracteres de nueva línea se puede trabajar fácilmente con los apuntadores de lectura y escritura, el cálculo del tamaño es muy sencillo, se pueden guardar estructuras.

* Archivos de texto:

En los archivos de texto se escribe solo texto no se establece cuánto espacio ocupara, también se lee sin especificar hasta donde leer solo se detendrá en espacio en blanco o salto de línea.

## 3.- ¿Por qué Linux considera que todos los archivos deben ser binarios?:

En Linux se utiliza archivos binarios ya que se permite reiniciar, restaurar, recuperar y/o reparar el sistema operativo sin necesidad de reiniciar. Proporciona métodos y herramientas adecuadas para el sistema.

Los archivos binarios se utilizan para administrar el sistema operativo ya que poseen comandos esenciales para manipular, restaurar y controlar parámetros y secuencias de comandos. Tampoco requiere convertir los archivos a ninguna otra representación al utilizarlos ya que todos son binarios.

En los sistemas [Unix](http://es.wikipedia.org/wiki/Unix) los nombres de los archivos pueden contener cualquier carácter excepto el separador de directorios ("/") y no tienen límite de tamaño.

La extensión no es un elemento diferenciador sino una convención dada a los nombres de archivo para reconocer su contenido.

Ya que todos los archivos son binarios, los usuarios y creadores de Linux lo consideran más avanzado, lo único que va a variar es el contenido y estará definido con la extensión.

En Linux existen básicamente 5 tipos de archivos:

* **Archivos ordinarios.** Contienen la información con la que trabaja cada usuario.
* **Enlaces físicos o duros (hard links).** No es específicamente una clase de archivo sino un segundo nombre que se le da a un archivo. Supón que dos usuarios necesitan compartir información de un mismo archivo. Si cada uno tuviera una copia del archivo se soluciona el problema, pero las modificaciones que realice un usuario no las vería el otro. Sin embargo, si creamos un enlace duro al archivo para cada usuario cada vez que uno de ellos modifique cualquier cosa en el archivo, el otro lo podrá ver puesto que realmente están viendo y modificando el mismo archivo.
* **Enlaces simbólicos.** También se utilizan para asignar un segundo nombre a un archivo. La diferencia con los enlaces duros es que los simbólicos solamente hacen referencia al nombre del archivo original, mientras que los duros hacen referencia al inodo en el que están situados los datos del archivo original. De esta manera, si tenemos un enlace simbólico y borramos el archivo original perderemos los datos, mientras que si tenemos un enlace duro los datos no se borrarán hasta que se hayan borrado todos y cada uno de los enlaces duros que existen hacia esos datos en el sistema de ficheros.
* **Directorios.** Son archivos especiales que contienen referencias a otros archivos o directorios.
* **Archivos especiales.** Suelen representar dispositivos físicos, como unidades de almacenamiento, impresoras, terminales, etc. En Linux, todo dispositivo físico que se conecte al ordenador está asociado a un archivo. Linux trata los archivos especiales como archivos ordinarios.

## 4.- ¿Por qué Microsoft Windows defiende que existan archivos de texto y binarios?:

Windows considera que los archivos para trabajar con el sistema deben ser datos o ejecutables. Los creados por el usuario deben ser de texto, como anotaciones hechas con el bloc de notas, Excel, Word etc. Estos utilizan una codificación de fácil interpretación por el usuario.

Tipos de Archivos:

* Ejecutables y datos:

Se pueden dividir los archivos en dos grandes grupos: los ejecutables y los no ejecutables o archivos de datos. La diferencia fundamental entre ellos es que losprimeros están creados para funcionar por sí mismos y los segundos almacenan información que sólo podrá ser utilizada con algún programa que la interprete.

* Binario vs ASCII:

Toda información se graba como ceros y unos en sistema binario, así un trozo de programa contiene instrucciones en código de máquina, que son secuencias de ceros y unos.

El código ASCII es un código de caracteres basado en el alfabeto latino tal como se usa en inglés moderno y en otras lenguas occidentales.

Fue creado en 1963 por el Comité Estadounidense de Estándares como una refundición o evolución de los conjuntos de códigos utilizados entonces en telegrafía.

Más tarde, en 1967, se incluyeron las minúsculas, y se redefinieron algunos códigos de control.

Casi todos los sistemas informáticos actuales utilizan el código ASCII o una extensión compatible para representar textos y para el control de dispositivos que manejan texto.

El código ASCII define una relación entre caracteres específicos y secuencias de bits, además de reservar unos cuantos códigos de control para el procesador de textos.

No todos los caracteres del código ASCII están permitidos para nombrar un archivo, pues algunos (ejemplo: /\* ¿?+\ y otros) son reservados para otras funciones.

## 5.- ¿Métodos de acceso y manejo de archivos en c++?:

Métodos de acceso a archivos:

* Secuencial:

Es en el que los elementos en el archivo tienen una relación de orden total, se puede saber que elemento esta antes y después.

* Aleatorio:

Es en el que los elementos en el archivo fueron colocados sin seguir un orden y pueden estar en cualquier parte del archivo.

Manejo de archivos:

* **Formatos de archivos binarios:**

Los archivos binarios contienen bytes a ser interpretados, las aplicaciones o programas compilados son conocidos como binarios, un archivo binario puede almacenar imágenes, sonido, versiones comprimidas de otros archivos, etc.

Algunos archivos binarios tienen una [cabecera](http://es.wikipedia.org/wiki/Cabecera_(inform%C3%A1tica)). Esta cabecera es un bloque de metadatos que un programa informático usará para interpretar correctamente la información contenida.

Por ejemplo, un archivo [GIF](http://es.wikipedia.org/wiki/GIF) puede consistir en múltiples imágenes y la cabecera se usa para identificar y describir cada bloque de datos de cada imagen. Si el archivo binario no tiene cabecera se dice que es un archivo binario plano.

* **Manipulando archivos binarios:**

Para enviar archivos binarios a través de varios sistemas como el [e-mail](http://es.wikipedia.org/wiki/E-mail) que no permiten todos los tipos de datos, éstos son convertidos a una representación textual como lo dicen las normas usando por ejemplo [Base24](http://es.wikipedia.org/wiki/Base64).

Esta codificación tiene el inconveniente de precisar una decodificación en el receptor para recuperar la información binaria.

[Microsoft Windows](http://es.wikipedia.org/wiki/Microsoft_Windows) permite al programador especificar un [parámetro](http://es.wikipedia.org/wiki/Argumento_(inform%C3%A1tica)) de [llamada al sistema](http://es.wikipedia.org/wiki/Llamada_al_sistema) que indique si un archivo es de texto o binario, [Unix](http://es.wikipedia.org/wiki/Unix) no lo permite y trata a todos los archivos como binarios.

* **Visualizando archivos binarios:**

Si se abre un archivo binario en un [editor de texto](http://es.wikipedia.org/wiki/Editor_de_texto), cada grupo de ocho bits serán traducidos normalmente como un carácter aislado y seguramente se observará un galimatías ininteligible de caracteres.

Si se intentase abrir con algún otro programa dicho programa daría un uso propio a cada [byte](http://es.wikipedia.org/wiki/Byte) y podría tratar cada byte como un número y crear un [flujo de datos](http://es.wikipedia.org/wiki/Flujo_de_datos) de salida de números entre 0 y 255 o tal vez interprete cada byte como un color y visualice un dibujo.

Si el archivo es tratado como un [ejecutable](http://es.wikipedia.org/wiki/Ejecutable) y se ejecuta el ordenador tratará de interpretar el archivo como una serie de instrucciones en su propio [lenguaje máquina](http://es.wikipedia.org/wiki/Lenguaje_m%C3%A1quina).

Se puede usar un [editor hexadecimal](http://es.wikipedia.org/wiki/Editor_hexadecimal) para observar los valores hexadecimales de los correspondientes bytes del archivo binario.

## 6.- ¿Por qué un archivo de texto plano puede ser considerado como una lista doble?:

Debido a que tienen dos apuntadores uno de lectura y otro de escritura, los cuales pueden ser desplazados independientemente por el archivo de texto del inicio al final y devolverlos al inicio cuando se desee. Estos se mueven a través de los elementos contenidos en el archivo elemento por elemento, hasta llegar al objetivo asignado.

## 7.- Diferencias de Linux y Windows:

|  |  |  |
| --- | --- | --- |
| **Diferencias de Linux y Windows** | | |
| Programa | Windows | Linux |
| 1.- Instalación | Es mínimamente configurable aunque es muy sencilla. | No resulta sencilla siempre, pero te permite personalizar totalmente los paquetes que quieras instalar. |
| 2.- Robustez | Siempre hay que reiniciar cuando se cambia la configuración del sistema, se bloquea cuando ejecuta operaciones y luego hay que reiniciar el equipo. | Se ha caracterizado siempre por la robustez de su sistema ya que pueden pasar meses e incluso años sin la necesidad de apagar o reiniciar el equipo, también si una aplicación falla simplemente no bloquea totalmente al equipo. |
| 3.- Mantenimiento | Se necesita antivirus que requieren recursos y escaneos frecuentes, un firewall que constantemente pregunta si quiere permitir a un software conectarse a Internet, pasar un anti-spyware rutinariamente y desconfiar de todo el software. | No se requieren antivirus |
| 4.- Inversión | Tienes que pagar por la licencia | Solo necesitas internet para descargarlo |
| 5.- Portable | Tus preferencias están en parte en el registro y por ello no son portátiles. | Todas las configuraciones se almacenan en la carpeta Home, y se puede copiar estas configuraciones de una máquina a otra. |
| 6.- Entender lo que pasa | Es como tener una caja negra, puedes verla por fuera pero no sabes qué pasa por dentro, si se tiene problemas sólo lo puede arreglar Microsoft. | Con sólo mirar los log del sistema puedes ver la mayor parte de los errores, buscar en Google y descubrir qué los causa, puedes incluso ver el  código fuente para encontrar el problema. |
| 7.-Personalizarlo | Puedes pagar para poner skins en el escritorio, pero aún así no son adaptables, es sólo una capa de pintura en el mismo desktop. | Puedes usar Windows manager, el ambiente de escritorio, el motor GTK, los efectos especiales, casi todos los aspectos tienen opciones competitivas, en rara vez se verán dos escritorios iguales. |
| 8.- Beneficiarse de la competencia | No sólo es divertido probar las diferentes opciones de software, sino que lleva a un mejor software cuando múltiples proyectos compiten entre sí. | Solo se cuenta con los programas ofrecidos por el sistema operativo |

## 8.- Archivos de texto y binarios en c++:

* Archivos binarios:

#include <fstream>

#include <string>

#include <iostream>

using namespace std;

using namespace System;

void esribirArchivo();

void leerArchivo();

const int MAX = 30;

int main(array<System::String ^> ^args){

int opc;

do{

Console::Clear();

Console::WriteLine(L"ARCHIVOS BINARIOS EN C++");

Console::WriteLine(L"1->Crear archivo y escribir en C++");

Console::WriteLine(L"2->Leer archivo en C++");

Console::WriteLine(L"3->Salir");

opc = int::Parse(Console::ReadLine());

if(opc==1)

esribirArchivo();

else if(opc==2)

leerArchivo();

Console::ReadKey(true);

}while(opc!=3);

//cadena = Console::ReadLine();

//a = int::Parse(Console::ReadLine());

return 0;

}

void esribirArchivo(){

char cadena[MAX];

string nombreArchivo;

char a;

Console::WriteLine(L"Ingrese nombre de archivo a crear");

cin>>nombreArchivo;

ofstream archivo(nombreArchivo.c\_str(),ios::out|ios::binary);

do{

Console::WriteLine(L"Ingrese cadena a guardar");

fflush(stdin);

cin.getline(cadena,MAX);

archivo.write((char\*)(&cadena),MAX\*sizeof(char));

Console::WriteLine(L"desea seguir guardando cadenas (s/n):");

fflush(stdin);

cin>>a;

}while(a!='n');

archivo.close();

ofstream archivo2(nombreArchivo.c\_str(),ios::binary|ios::app|ios::out);

cout<<"elementos guardados en archivo "<<

archivo2.seekp(0,ios::end).tellp()/MAX\*sizeof(char)<<endl;

archivo2.close();

}

void leerArchivo(){

char cadena[MAX];

string nombreArchivo;

string prueba="dfd";

Console::WriteLine(L"Ingrese nombre de archivo a leer");

cin>>nombreArchivo;

ifstream archivo(nombreArchivo.c\_str(),ios::in|ios::binary);

while(!archivo.eof()){

fflush(stdin);

archivo.read((char\*)(&cadena),MAX\*sizeof(char));

if(cadena!=prueba)

prueba =cadena;

else

break;

cout<<cadena<<' ';

}

archivo.close();

ifstream archivo2(nombreArchivo.c\_str(),ios::binary|ios::app|ios::in);

cout<<"elementos guardados en archivo "<<

archivo2.seekg(0,ios::end).tellg()/MAX\*sizeof(char)<<endl;

archivo2.close();

}

* Archivos de texto:

#include <fstream>

#include <string>

#include <iostream>

using namespace std;

using namespace System;

void esribirArchivo();

void leerArchivo();

int main(array<System::String ^> ^args){

int opc;

do{

Console::Clear();

Console::WriteLine(L"ARCHIVOS DE TEXTO EN C++");

Console::WriteLine(L"1->Crear archivo y escribir en C++");

Console::WriteLine(L"2->Leer archivo en C++");

Console::WriteLine(L"3->Salir");

opc = int::Parse(Console::ReadLine());

if(opc==1)

esribirArchivo();

else if(opc==2)

leerArchivo();

Console::ReadKey(true);

}while(opc!=3);

//cadena = Console::ReadLine();

//a = int::Parse(Console::ReadLine());

return 0;

}

void esribirArchivo(){

string cadena;

string nombreArchivo;

char a;

Console::WriteLine(L"Ingrese nombre de archivo a crear");

cin>>nombreArchivo;

ofstream archivo(nombreArchivo.c\_str(),ios::out);

do{

Console::WriteLine(L"Ingrese cadena a guardar");

fflush(stdin);

getline(cin,cadena);

archivo<<cadena<<' ';

Console::WriteLine(L"desea seguir guardando cadenas (s/n):");

fflush(stdin);

cin>>a;

}while(a!='n');

archivo.close();

}

void leerArchivo(){

string cadena;

string nombreArchivo;

string prueba="dfd";

Console::WriteLine(L"Ingrese nombre de archivo a leer");

cin>>nombreArchivo;

ifstream archivo(nombreArchivo.c\_str(),ios::in);

while(!archivo.eof()){

fflush(stdin);

archivo>>cadena;

if(cadena!=prueba)

prueba =cadena;

else

break;

cout<<cadena<<' ';

}

archivo.close();

}

Para trabajar con los apuntadores de lectura y escritura en se utiliza:

|  |  |
| --- | --- |
| ios :: in | Abierto para operaciones de entrada. |
| ios :: out | Abierto para operaciones de salida. |
| ios :: binary | Abrir en modo binario. |
| ios :: comió | Establecer la posición inicial al final del archivo.  Si este indicador no está establecido en cualquier valor, la posición inicial es el principio del archivo. |
| ios :: app | Todas las operaciones de salida se realizan al final del fichero, añadiendo el contenido para el contenido actual del archivo. Esta bandera sólo se puede utilizar en los flujos abiertos por tan solo salida-operaciones. |
| ios :: trunc | Si el archivo abierto para operaciones de salida ya existían antes, su contenido anterior se borra y se reemplaza por el nuevo. |

|  |  |
| --- | --- |
| ios :: beg | compensar contados desde el comienzo de la corriente |
| ios :: cur | desplazamiento contados a partir de la posición actual del puntero de secuencia |
| ios :: end | compensar contados desde el final de la secuencia |

|  |  |
| --- | --- |
| **clase** | **parámetro por defecto el modo de** |
| ofstream | ios :: out |
| ifstream | ios :: in |
| fstream | ios :: in | ios :: out |

|  |  |
| --- | --- |
| tellp | Se usa para obtener el tamaño del archivo ofstream desde cero al apuntador |
| tellg | Se usa para obtener el tamaño del archivo ifstream desde cero al apuntador |
| seekp | Se usa para mover el apuntador ofstream hasta el donde se le asigne |
| seekg | Se usa para mover el apuntador ifstream hasta el donde se le asigne |

## Conclusiones

* En los diferentes programas de Windows y Linux se ofrece una amplia variedad de archivos, entre otros se tienen los editores texto, gráficos y otros.
* Las ventajas y desventajas de los programas de Windows como de Linux se han expuesto en el capítulo 7.