CS1112: Programación 2

Unidad 9: Archivos

Sesión de Teoría - 14

Profesor:
José Antonio Fiestas Iquira
jfiestas@utec.edu.pe

Material elaborado por:
Maria Hilda Bermejo, José Fiestas, Rubén Rivas

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Índice:

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Logro de la sesión:

Al finalizar la sesión, los alumnos desarrollan sus programas utilizando archivos.

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

9 Unidad 9: Archivos

Archivo: Introducción

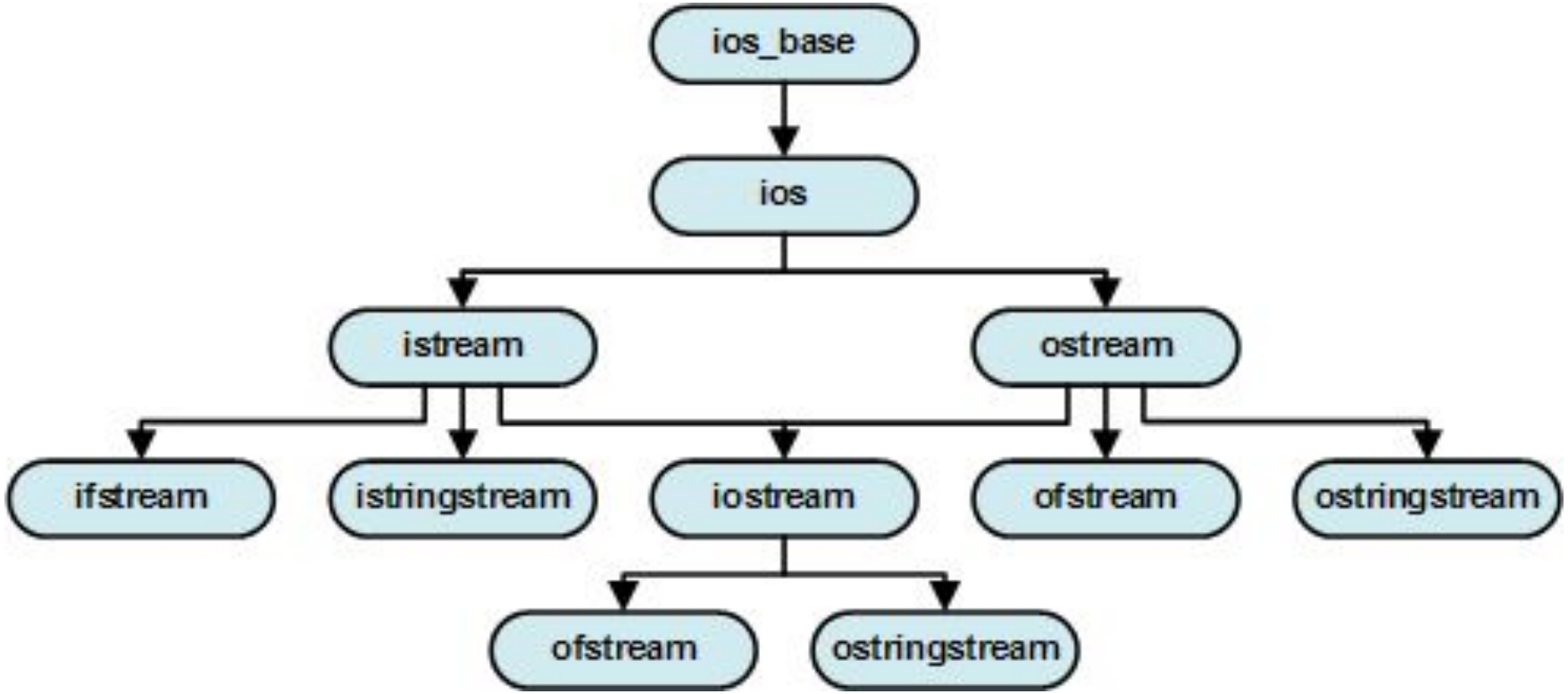
Un archivo en C++ es una unidad de almacenamiento digital y permanente en el computador.

- El sistema operativo administra los archivos, los organiza en estructuras jerárquicas conocidas como **folder o directorios**, facilitando su organización, búsqueda y almacenamiento.
- El nombre de los archivos utiliza el **punto (.)** para añadir la **extensión del archivo**, para clasificarlos por tipos de archivos.

Archivo: Introducción

- La interfaz de los dispositivos como el teclado, monitor, la impresora, redes, etc., se manejan también como archivos, e.g. **std::cin** y **std::cout** ,que derivan de la misma clase base que se utiliza para representar **archivos en C++** o sea, la clase **std::ios_base**.
- Un archivo de texto, es un tipo especial de archivo que no tiene un formato específico y que contiene únicamente texto que puede ser leído por las personas.

Archivo: Introducción



Estructura de la clase ios_base y sus derivadas. Fuente: Stackoverflow

Convenciones de Nombres de Archivo

En C++ la extensión de los archivos representa lo siguientes tipos:

- archivo fuente: **.cpp, .cc, .cxx**
- archivo de cabecera: **.hpp: .h, hxx**
- archivo ejecutable: **.exe, .a .out**
- archivo biblioteca: **.bin, .lib, .dll**

Objetos de Archivos Stream

- El uso de archivos requiere de objetos de archivo stream
- Existen tres tipos de objetos de archivo stream:

(1) ifstream : usado para leer
(2) ofstream : usado para escribir
(3) fstream : usado para ambos leer y escribir

Nombre del Archivo

- El nombre del archivo puede incluir la ruta completa al archivo:
  - **`c:\datos\`**`notas.dat` ***en MS Windows***
  - **`/usr/ubuntu/`**`notas.dat` ***en derivados de UNIX***

  Esto le indica al compilador exactamente donde ubicarlo.
- El nombre del archivo puede ser también un nombre simple:

  `notas.dat`

  Este debe estar en el mismo directorio del programa ejecutable, o en el directorio por defecto del compilador.

Pasos para usar un Archivo

1. Abrir el Archivo
2. Usar (leer de, escribir a) el archivo
3. Cerrar el Archivo

ifstream: Abriendo un Archivo para Leer

- Crea un objeto `ifstream` en tu programa

```
ifstream archivo_entrada;
```

- Abre el archivo para pasar su nombre a la función miembro open del objeto stream

```
archivo_entrada.open("notas.dat");
archivo_entrada >> x;
archivo_entrada.close();
```

ofstream: Abriendo un Archivo para Escribir

- Crea un objeto **`ofstream`** en tu programa

```
ofstream archivo_salida;
```

- Abre el archivo por pasar el nombre a la función miembro open del objeto stream

```
archivo_salida.open("notas.dat");
archivo_salida << x;
arhcivo_salida.close();
```

fstream: Archivo para Leer o Escribir

- **El objeto `fstream` puede ser usado para Leer o Escribir**

```
fstream archivo_entrada_salida;
```

- **Para Leer se debe de especificar `ios::in` como el segundo argumento para abrir el archivo**

```
archivo_entrada_salida.open( "notas.dat",
                             ios::in);
```

- **Para Escribir se debe de especificar `ios::out` como el segundo argumento para abrir el archivo**

```
archivo_entrada_salida.open( "notas.dat",
                             ios::out);
```

Abriendo un Archivo para Leer y Escribir

- El objeto **`fstream`** puede ser usado para Leer y Escribir al mismo tiempo

- Crea el objeto **`fstream`** y especifica ambos **`ios::in`** y **`ios::out`** como el segundo argumento para la función miembro open

```
fstream archivo_entrada_salida;
archivo_entrada_salida.open(
        "notas.dat",
        ios::in|ios::out);
```

Abriendo Archivos con Constructores

Incluyen el **open**

```
fstream archivo_entrada(
        "notas.dat",
        ios::in);
```

Modos de Abrir un archivo

- El modo de abrir un archivo especifica como el archivo es abierto y que se puede hacer con el archivo una vez abierto.

- ios::in y ios::out son ejemplos de modos de abrir un archivo, también llamado indicadores del modo de archivo

- Los modos de archivo pueden ser combinados y pasados como segundo argumento de la función miembro open

Indicadores del Modo de Archivo

`ios::app`	crea un nuevo archivo, o agrega al final de un archivo existente
`ios::ate`	va al final de un archivo existente; y puede escribir en cualquier parte del archivo
`ios::binary`	lee/escribe en modo binario (no en modo texto)
`ios::in`	abre para leer
`ios::out`	abre para escribir

`app` seek to end before each write `ate` open and seek to end immediately after opening

With `ios::app` the write position in the file is "sticky" -- all writes are at the end, no matter where you seek.

Modos por Defecto de Abrir un archivo

- **`ofstream`:**
  - abre solo para escribir
  - no puede ser leído el contenido del archivo
  - se crea el archivo si no existe
  - el contenido es borrado si existe el archivo

- **`ifstream`:**
  - abre solo para leer
  - no puede ser escrito el archivo
  - falla al abrir si el archivo no existe

Detectando errores abriendo un Archivo

Dos métodos para detectar si falla al abrir un archivo

(1) Llamar a la función miembro **fail()** en el stream

```
archivo_entrada.open("notas.dat");
if (archivo_entrada.fail())
  { cout << "No puede abrir el archivo";
    exit(1);
  }
```

Detectando errores abriendo un Archivo

(2) Verificando el estatus del stream

```
archivo_entrada.open("notas.dat");
if (!archivo_entrada.is_open())
  { cout << "No se puede abrir el archivo";
    exit(1);
  }
```

Usando fail() para detectar eof

Ejemplo de lectura de todos los enteros en un archivo:

```
// intentando leer
int x;
archivo_entrada >> x;
while (!archivo_entrada.fail()) {
    // Exitoso, no es un eof
    cout << x;
    // lee nuevamente otro entero
    archivo_entrada >> x;
}
```

Usando >> para Detectar eof

El operador de extracción retorna el mismo valor que será retornado por la siguiente llamada a fail:

- (**`archivo_entrada >> x`**)

  no retorna cero si >> es exitoso

- (**`archivo_entrada >> x`**)

  *retorna **cero** si >> es fin de archivo*

Detectando el Final de un archivo

Se lee enteros desde un archivo y se imprimen

```
int x;
while (archivo_entrada >> x) {
    // la lectura fue exitosa
    cout << x;
    // va al tope del bucle while e
    // intenta otra lectura
}
```

Ejemplos

Ejemplo 1:

Escriba un programa que permita realizar lo siguiente:

1. Generar *N* números aleatorios del *1 al N* para que sean almacenados en un archivo llamado ***datos.txt***.
2. Generar un segundo archivo llamado ***datos_ordenados.txt*** que contenga los datos del archivo ***datos.txt*** pero esta vez ordenados de manera ascendente.
3. Genera un tercer archivo llamado ***datos_contados.txt*** en donde figure los datos en dos columnas: en la primera columna debe ir el número y en la segunda columna las veces que aparece ese número en el archivo ***datos_ordenados.txt***

Archivos que forman parte del proyecto.

Archivos con código:

- Main.cpp
- Archivos.h
- Archivos.cpp

Archivos con datos:

- datos.txt
- datos_ordenandos.txt
- datos_contados.txt

datos.txt.
Incluye los cambios de línea

```
16
2
18
6
20
2
6
5
6
8
15
8
19
16
7
4
7
11
5
6
```

datos_ordenados.txt

```
2
2
4
5
5
6
6
6
6
7
7
8
8
11
15
16
16
18
19
20
```

datos_contados.txt

```
2, 2
4, 1
5, 2
6, 4
7, 2
8, 2
11, 1
15, 1
16, 2
18, 1
19, 1
20, 1
```

Ejemplo 1:

Main.cpp

```cpp
#include <iostream>
#include "Archivos.h"

int main() {
   generar_archivo("datos.txt",20);
   generar_archivo_ordenado("datos.txt","datos_ordenados.txt");
   generar_datos_contados("datos_ordenados.txt", "datos_contados.txt");

   return 0;
}
```

Archivos.h

```cpp
#include <iostream>
#include <fstream>
#include <random>
#include <string>
#include <map>
using namespace std;

void generar_archivo(string nombrefisico, int cantidad);
void generar_archivo_ordenado(string nombreArchivoOriginal, string nombreArchivoOrdenado);
void generar_datos_contados(string nombreArchivoOrdenado, string nombreArchivoContados);
```

...Ejemplo 1: generar_archivo

Archivo.cpp

```cpp
#include "Archivos.h"
using namespace std;

void generar_archivo(string nombrefisico, int cantidad)
{
    random_device r;
    fstream archivo(nombrefisico, ios::out);
    // Generar los n numeros
    for (int i = 0; i < cantidad; ++i) {
        archivo << 1 + r() % cantidad << '\n';
    }
    archivo.close();
}
```

...Ejemplo 1: generar_archivo_ordenado

Archivos.cpp

```
void generar_archivo_ordenado(string nombreArchivoOriginal, string nombreArchivoOrdenado)
{ fstream original(nombreArchivoOriginal, ios::in);
   // Verificando si se pudo abrir el archivo
   if (!original.is_open()) {
       cout << "Error abriendo archivo \"datos.txt\"\n";
       return;
   }
   // Crear un objeto archivo que crea un archivo "datos_ordenados.txt"
   fstream ordenada(nombreArchivoOrdenado, ios::out);
   // Verificando si se pudo abrir el archivo
   if (!ordenada.is_open()) {
       cout << "Error abriendo archivo \"datos_ordenados.txt\"\n";
       return;
   }
   // Lee las lineas desde el archivo fuente en el ejemplo "datos.txt"
   // hacia un mapa (se ordena automaticamente)
   map<int, int> data;
   string key;
   while(getline(original, key))
   {
       data[stoi(key)]++; // Convirtiendo string a entero (stoi)
   }
```

...Ejemplo 1: generar_archivo_ordenado

Archivos.cpp

```
// Grabar los elementos del mapa al archivo que tendrá
los datos ordenados
   // en el ejemplo : "datos_ordenados.txt"
   for (auto it = begin(data); it != end(data); ++it) {
       for (int i = 0; i < it->second; ++i)
           ordenada << it->first << '\n';
   }
   original.close();
   ordenada.close();
}
```

...Ejemplo 1: generar_datos_contados

Archivos.cpp

```
// 3. Escribir un programa que a partir del archivo ordenado ("datos_ordenados.txt")
// genere directamente un archivo con 2 columnas ("datos_contados.txt").

void generar_datos_contados(string nombreArchivoOrdenado, string nombreArchivoContados)
{
   // Crear un objeto archivo que abra un archivo "datos_ordenados.txt"
   fstream ordenada(nombreArchivoOrdenado, ios::in);
   // Verificando si se pudo abrir el archivo
   if (!ordenada.is_open()) {
       cout << "Error abriendo archivo \"datos_ordenados.txt\"\n";
       return;
   }
   // Crear un objeto archivo que crea un archivo "datos_contados.txt"
   fstream contados(nombreArchivoContados, ios::out);
   // Verificando si se pudo abrir el archivo
   if (!contados.is_open()) {
       cout << "Error abriendo archivo \"datos_contados.txt\"\n";
       return;
   }
```

...Ejemplo 1: generar_datos_contados

Archivos.cpp

```
    // Contando y Grabando los valores en el nuevo archivo "datos_contados.txt"
    string line;
    getline(ordenada, line);
    int value = stoi(line);
    int contar = 1;
    while (getline(ordenada, line)) {
        if (value != stoi(line)) {
            contados << value << ", " << contar << '\n';
            value = stoi(line);
            contar = 0;
        }
        contar++;
    }
    // Grabando el ultimo valor
    contados << value << ", " << contar << '\n';
    ordenada.close();
    contados.close();
}
```

Explorando lo aprendido

- ¿Qué diferencia existe entre ofstream y ifstream?
- ¿Qué diferencia existe entre ofstream y fstream?
- ¿Cuáles son las formas de abrir un archivo?
- Considerando que el objeto **archivo** es **std::fstream** y **texto** es **std::string** ¿Que valor retorna **archivo >> texto** si se alcanzó el final del archivo?

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Bibliografía:

Deitel. P.J. and Deitel. H. M. (2016) C++ How to Program, Prentice Hall.

Stroustrup, Bjarne (2013). The C++ Programming Language, 4th Addison-Wesley.

Eckel, Bruce, 2000. Thinking in C++, Vol 1: Introduction to Standard C++, 2nd Edition, Prentice Hall

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

¡Nos vemos en la siguiente clase!

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA