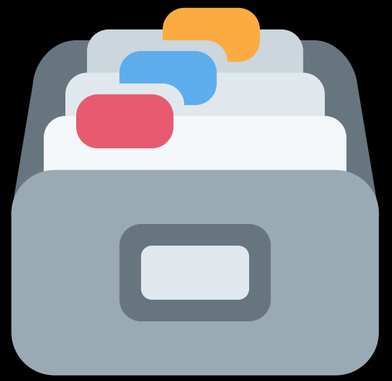
UNIVERSIDAD TECNOLOGICA NACIONAL
Facultad Regional Bahía Blanca

Manejo de Archivos



Lectura y Parseo de Archivos CSV sin Librerías Externas

Aprenderemos a leer y parsear un archivo txt o CSV (valores separados por coma) para obtener todo su contenido, utilizando solo funciones nativas de C++.

Algoritmo

1. **Apertura y Lectura del Archivo:**
   - Abriremos el archivo CSV y lo leeremos línea por línea.
   - La primera línea, que generalmente es el encabezado, será omitida inicialmente.

2. **Lectura Línea por Línea:**
   - Leeremos cada línea del archivo hasta el final usando un ciclo.

3. **Parseo de las Líneas:**
   - Cada línea se separará en valores individuales usando el delimitador (',').

4. **Manipulación de los Valores:**
   - Cada valor separado se almacenará en variables individuales.
   - Podremos realizar acciones como imprimir los valores o escribirlos en otro archivo.

```cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <string>

using namespace std;

int main() {
  ifstream archivo("archivo.csv"); // Abrir archivo CSV
  string linea;

  if (archivo.is_open()) {
    getline(archivo, linea); // Leer y omitir la primera línea (encabezado)

    while (getline(archivo, linea)) {
      stringstream ss(linea);
      string valor;

      while (getline(ss, valor, ',')) {
        // Aquí puedes trabajar con cada valor
        cout << "Valor: " << valor << endl;
      }
    }

    archivo.close(); // Cerrar archivo
  } else {
    cout << "No se pudo abrir el archivo." << endl;
  }

  return 0;
}
```

.CSV

Nombre,Edad,Ciudad
Juan,25,Bahía Blanca
María,30,White
Carlos,28,Santa Rosa

ss: Es un objeto de tipo std::stringstream. Esta clase permite operaciones de entrada y salida sobre cadenas, similar a trabajar con flujos de entrada y salida estándar (std::cin y std::cout), pero en este caso sobre cadenas de texto.

valor: Es una variable donde se almacenará cada elemento encontrado entre las comas. Es de tipo std::string, asumiendo que valor está definido como std::string valor; antes de este bucle.

',': Es el delimitador que indica dónde debe cortar la función getline. En este caso, la función getline buscará el próximo fragmento de texto hasta encontrar una coma (','), y ese fragmento se almacenará en valor.

Entonces, el programa anterior lee el archivo "archivo.csv", omite la línea de encabezado ("Nombre,Edad,Ciudad") y luego imprime cada valor separado por comas en líneas individuales.

```
Valor: Juan
Valor: 25
Valor: Bahía Blanca
Valor: María
Valor: 30
Valor: White
Valor: Carlos
Valor: 28
Valor: Santa Rosa
```

Leer CSV con C++

separar.cpp

C: > Users > parzibyte > Desktop > separar.cpp > NOMBRE_ARCHIVO

```cpp
/*
    https://parzibyte.me/blog
*/
#include <iostream>
#include <sstream>
#include <fstream>
#define NOMBRE_ARCHIVO "ProductosExportados_SPOS3.csv"
using namespace std;

int main()
{
    ifstream archivo(NOMBRE_ARCHIVO);
    string linea;
    char delimitador = ',';
    // Leemos la primer línea para descartarla, pues es el encabezado
    getline(archivo, linea);
    // Leemos todas las líneas
    while (getline(archivo, linea))
```

ProductosExportados_SPOS3.csv

C: > Users > parzibyte > Desktop > ProductosExportados_SPOS3.csv

```
idProducto,codigoBarras,descrip
4,4,NodeMCU ESP8266,100.00,150.
3,3,Nintendo Switch,8000.00,850
2,2,Audifonos HyperX,1200.00,15
1,1,Teclado HyperX,1800.00,2000
```

Cmder

```
C:\Users\parzibyte\Desktop
λ g++ separar.cpp -o separar.exe

C:\Users\parzibyte\Desktop
λ separar.exe
==================
Id: 4
Codigo de barras: 4
Descripcion: NodeMCU ESP8266
Precio de compra: 100.00
Precio de venta: 150.00
Existencia: 200.00
Stock: 1.00
==================
Id: 3
Codigo de barras: 3
Descripcion: Nintendo Switch
Precio de compra: 8000.00
Precio de venta: 8500.00
Existencia: 20.00
Stock: 2.00
==================
Id: 2
Codigo de barras: 2
Descripcion: Audifonos HyperX
Precio de compra: 1200.00
Precio de venta: 1500.00
Existencia: 20.00
Stock: 1.00
==================
Id: 1
Codigo de barras: 1
Descripcion: Teclado HyperX
Precio de compra: 1800.00
Precio de venta: 2000.00
Existencia: 500.00
Stock: 20.00
```

Leer y extraer valores de CSV con C++

Ahora sí veamos el código. Primero abrimos el archivo y lo leemos:

```
ifstream archivo(NOMBRE_ARCHIVO);
string linea;
char delimitador = ',';
// Leemos la primer línea para descartarla, pues es el encabezado
getline(archivo, linea);
// Leemos todas las líneas
while (getline(archivo, linea))
{

    // Magia aquí...
}

archivo.close();
```

Dentro del ciclo tendremos la línea, y es momento de separarla para almacenar sus valores. Como bien dice arriba, el delimitador es la coma. Entonces hacemos lo siguiente:

```
stringstream stream(linea); // Convertir la cadena a un stream
string idProducto, codigoBarras, descripcion, precioCompra, precioVenta, existencia, stock;
// Extraer todos los valores de esa fila
getline(stream, idProducto, delimitador);
getline(stream, codigoBarras, delimitador);
```

```
getline(stream, descripcion, delimitador);
getline(stream, precioCompra, delimitador);
getline(stream, precioVenta, delimitador);
getline(stream, existencia, delimitador);
getline(stream, stock, delimitador);
```

Leemos de la línea y colocamos cada valor dentro de cada variable. Recuerda que debe ser en el mismo orden que tienen las columnas dentro del archivo CSV.

```
// Imprimir
cout << "====================" << endl;
cout << "Id: " << idProducto << endl;
cout << "Codigo de barras: " << codigoBarras << endl;
cout << "Descripcion: " << descripcion << endl;
cout << "Precio de compra: " << precioCompra << endl;
cout << "Precio de venta: " << precioVenta << endl;
cout << "Existencia: " << existencia << endl;
cout << "Stock: " << stock << endl;
```

Escritura de datos en un archivo CSV

Definición de la estructura Persona: Se define una estructura Persona para almacenar el nombre, edad y ciudad de una persona.

Vector de datos de ejemplo: Se crea un vector de objetos Persona con datos de ejemplo.

Nombre del archivo CSV: Se define el nombre del archivo CSV de salida como "datos_salida.csv".

Apertura del archivo para escritura: Se abre el archivo "datos_salida.csv" en modo escritura usando ofstream.

Escritura en el archivo:

- Primero se escribe el encabezado "Nombre,Edad,Ciudad".
- Luego se recorre el vector personas y se escribe cada objeto Persona en una nueva línea del archivo, separando los campos por comas.

Cierre del archivo: Se cierra el archivo después de escribir todos los datos.

- Mensaje de confirmación: Si el archivo se abre correctamente y se escribe correctamente, se imprime un mensaje indicando que la escritura fue exitosa.

```cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector> //permite almacenar un número variable
//de elementos y su tamaño puede crecer
//dinámicamente según sea necesario.
using namespace std;

// Estructura para representar los datos

struct Persona {
   string nombre;
   int edad;
   string ciudad;
};

int main() {
   // Crear algunos datos de ejemplo
   vector<Persona> personas = {
      {"Juan", 25, "Bahia Blanca"},
      {"María", 30, "White"},
      {"Carlos", 28, "Santa Rosa"}
   };

   // Nombre del archivo CSV
   string nombreArchivo = "datos_salida.csv";


      }


}
```

```cpp
// Abrir el archivo CSV para escritura
ofstream archivo(nombreArchivo);

if (archivo.is_open()) {
// Escribir encabezado
archivo << "Nombre,Edad,Ciudad" << endl;

// Escribir datos de cada persona
for (const auto& persona : personas) {
archivo << persona.nombre << "," << persona.edad << "," << persona.ciudad << endl;

  // Cerrar el archivo
     archivo.close();
     cout << "Se ha escrito exitosamente en " << nombreArchivo << endl;
  } else {
     cout << "No se pudo abrir el archivo para escritura." << endl;
  }

  return 0;
```

```
Se ha escrito exitosamente en datos_salida.csv
```