

[Fecha]

TIPOS DE DATO PRIMITIVOS Y TIPOS DE DATO ESTRUCTURADOS



CUCEI

david gutierrez alvarez
Estructura de datos I

RESUMEN PERSONAL Y FORMA DE ABORDAR EL PROBLEMA

Lo primero que hice fueron algunas practicas de OOP para introducirme mas a este concepto, después de esto empecé haciendo el menú ya que era lo mas fácil, luego continúe haciendo el inciso 'a' y encontré una librería que me permitía resolver el problema de utilizar los recursos del lenguaje `#include <limits>` y sobre la función `sizeof()` no tuve problemas ya que ya la conocía, así fue como logre cumplir esta primer mitad de la actividad.

La segunda parte continuando con el inciso 'b' realmente no fue tan complicada como lo pensaba, inicia dando un limite de 10 para la matriz con `static const int MAX = 10;` después de esto cree los setters y getters para poder trabajar con las matrices hasta llegar al problema de generar números aleatorios y que estos sean distintos en cada compilación, al investigar encontré esto `srand((unsigned)time(NULL));` que utiliza el tiempo para generar estos números aleatorios, ya después de esto cree las funciones que trabajan con las matrices para generarlas, operarlas y mostrarlas.

Fue una practica muy interesante y buena para introducirme a la Programación Orientada a Objetos, seguiré mejorando.

Main.cpp

```
#include <menu.h>
#include <iostream>

int main() {
    Menu inicio;
    inicio.MostrarMenu();
}
```

Matrices.cpp

```
#include "matrices.h"
#include <stdlib.h>
#include <ctime>

matrices::matrices() {
    srand((unsigned)time(NULL));
}

void matrices::setMatriz() {
    /*con esto se rellenara la matriz*/
    matriz = float(-1000+(rand()%2000))*10/100;
}

float matrices::getMatriz() const {
    return matriz;
}

void matrices::setMatrizSuma(float matriz1, float matriz2) {
    matriz = matriz1+matriz2;
}

void matrices::setMatrizMult(float matriz1, float matriz2) {
    matriz = matriz1*matriz2;
}
```

Matrices.h

```
#ifndef MATRICES_H
#define MATRICES_H

class matrices {
private:
    float matriz;

public:
    matrices();
    void setMatriz();
    float getMatriz() const;
    void setMatrizSuma(float m1, float m2);
    void setMatrizMult(float m1, float m2);
};

#endif // MATRICES_H
```

Menú.cpp

```
#include "menu.h"
#include <iostream>
#include <limits>
#include <float.h> /*buscar*/

//using namespace std;

void Menu::MostrarMenu() {
    bool salir = false;
    char option;

    do {
        std::cout << "Menu" << std::endl
            << "a) Tamano y rangos de los Tipos de Dato Primitivos" << std::endl
            << "b) Ejemplo de uso de Tipo de dato Estructurado" << std::endl
            << "C) Salir"
            << "Opcion a elejir: ";
        std::cin >> option;

        switch (option) {
            case 'a':
                DatosPrimitivos();
                break;
            case 'b':
                DatoEstructurado();
                break;
            case 'c':
                salir = true;
                break;
            default:
                std::cout << "Opcion no valida, por favor escoja una valida" <<
std::endl;
        }
        system ("pause");
        system ("cls");
    } while(!salir);
}

void Menu::DatosPrimitivos() {
    std::cout << Separacion << std::endl
        << "| \tTIPO DE DATO\t | \tBITS\t | VALOR MINIMO | VALOR MAXIMO |\n"
        << Separacion << std::endl

        << "|Caracter con signo | \t" << sizeof(char) << "\t | \t"
        << SCHAR_MIN << " \t | \t" << SCHAR_MAX << "\t |" << std::endl
        << Separacion << std::endl

        << "|Caracter sin signo | \t" << sizeof(unsigned char) << "\t | \t"
        << 0 << " \t | \t" << UCHAR_MAX << "\t |" << std::endl
        << Separacion << std::endl

        << "|Entero corto con signo | \t" << sizeof(short) << "\t | \t"
        << SHRT_MIN << " \t | \t" << SHRT_MAX << "\t |" << std::endl
        << Separacion << std::endl

        << "|Entero corto sin signo | \t" << sizeof(unsigned short) << "\t | \t"
        << 0 << " \t | \t" << USHRT_MAX << "\t |" << std::endl
        << Separacion << std::endl

        << "|Entero largo con signo | \t" << sizeof(long) << "\t | "
        << LONG_MIN << " | " << LONG_MAX << " |" << std::endl
```

```

        << Separacion << std::endl

        <<"|Entero largo sin signo |\t"<< sizeof(unsigned long int) <<"\t |\t"
        << 0 << " |\t | " << ULONG_MAX << " |\t |" << std::endl
        << Separacion << std::endl

        <<"|Real de precision simple|\t"<< sizeof(float) <<"\t | "
        << FLT_MIN << " | " << FLT_MAX << " |\t |" << std::endl
        << Separacion << std::endl

        <<"|Real de precision doble |\t"<< sizeof(double) <<"\t | "
        << DBL_MIN << " | " << DBL_MAX << " |\t |" << std::endl
        << Separacion << std::endl;
    }

void Menu::DatoEstructurado() {
    do {
        std::cout << "Tamano de Matriz: ";
        std::cin >> TamMatriz;
        if ((TamMatriz<3) || (TamMatriz>10)) {
            std::cout<< "Dato no valido"<< std::endl;
            std::cout<<"El tamaño de la matriz debe ser como minimo y como maximo 10\n";
        }
    }while(TamMatriz<3 || TamMatriz>10);

    RellenarMatriz(Matriz_1);
    RellenarMatriz(Matriz_2);

    /*Mostrando Matrices de nxn*/
    std::cout<<"\t\tMatriz 1"<< std::endl;
    MostrarMatriz(Matriz_1);
    std::cout << line << std::endl;

    std::cout <<"\t\tMatriz 2"<< std::endl;
    MostrarMatriz(Matriz_2);
    std::cout << line << std::endl;

    SumaYProducto(Matriz_1, Matriz_2);

    /*MOSTRANDO RESULTADOS DE LAS OPERACIONES CON MATRICES*/
    std::cout << "\t\tMultiplicacion de entre Matriz 1 y Matriz 2 : " << std::endl;
    MostrarMatriz(productoMatriz);

    std::cout << line << std::endl;
    std::cout<<"\t\tSuma de Matriz 1 Y Matriz 2 \n";
    MostrarMatriz(sumaMatriz);
}

void Menu::RellenarMatriz(matrices matriz[MAX][MAX]) {
    for(int i=0;i<TamMatriz;i++) {
        for(int j=0;j < TamMatriz;j++) {
            matriz[i][j].setMatriz();
        }
    }
}

void Menu::MostrarMatriz(matrices matriz[MAX][MAX]) {
    for(int i=0;i < TamMatriz;i++) {
        for(int j=0;j < TamMatriz;j++){
            std::cout << matriz[i][j].getMatriz()<<" ";
        }
        std::cout << std::endl;
    }
}

```

```

    }
}

void Menu::SumaYProducto(matrices matriz1[MAX][MAX], matrices matriz2[MAX][MAX]) {
    for(int i = 0; i < TamMatriz; i++){
        for(int j = 0; j < TamMatriz; j++) {
            productoMatriz[i][j].setMatrizMult(matriz1[i][j].getMatriz(),
matriz2[i][j].getMatriz());
            sumaMatriz[i][j].setMatrizSuma(matriz1[i][j].getMatriz(),
matriz2[i][j].getMatriz());
        }
    }
}
}

```

Menu.h

```

#ifndef MENU_H
#define MENU_H

#include "matrices.h"

class Menu {
    static const int MAX = 10;

    /*ATRIBUTOS*/
private:
    matrices Matriz_1[MAX][MAX],
        Matriz_2[MAX][MAX],
        productoMatriz[MAX][MAX],
        sumaMatriz[MAX][MAX];

    int TamMatriz;

    char Separacion[75] = " | _____ | "
                        " _____ | "
                        " _____ | "
                        " _____ | ";

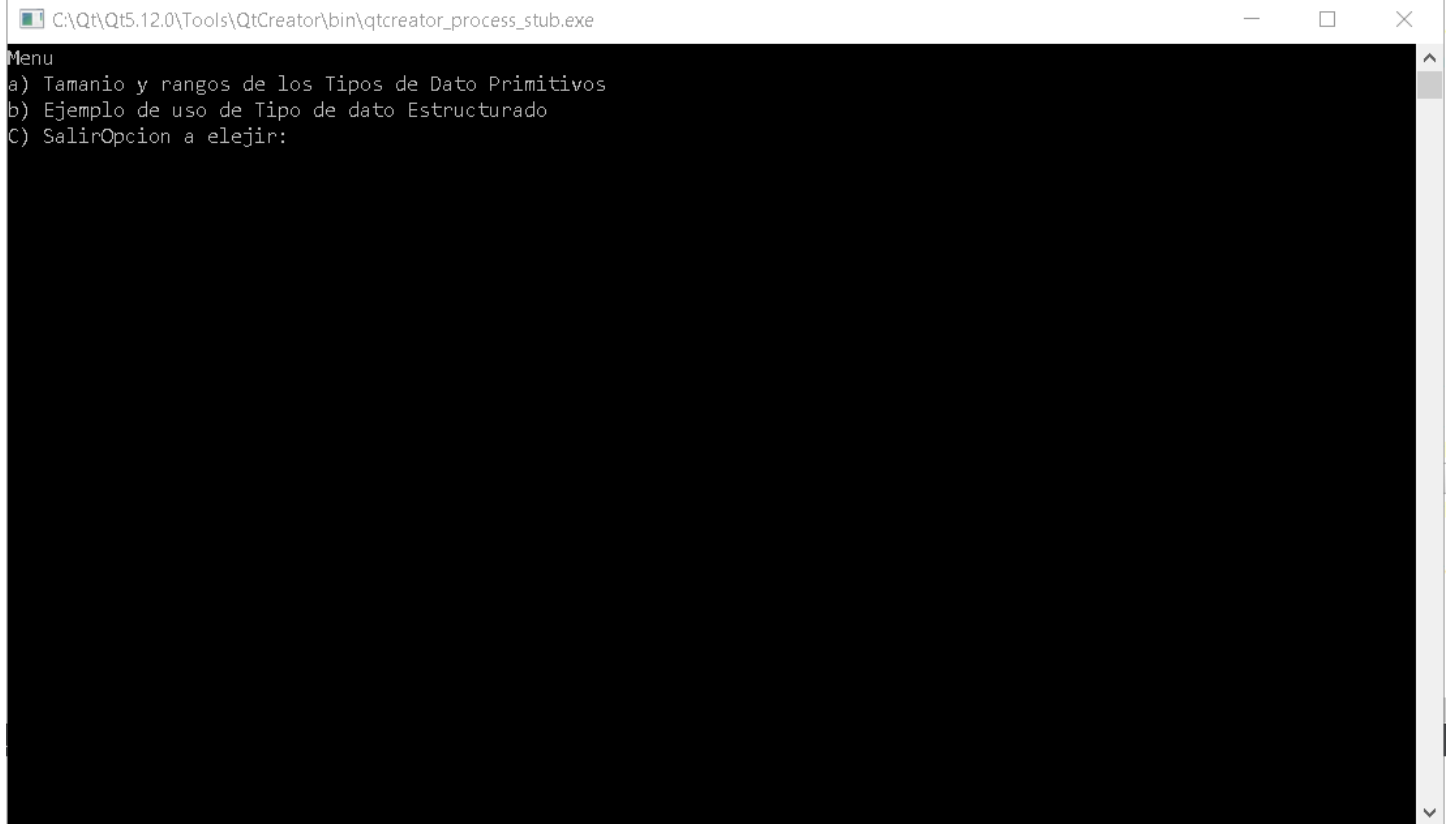
    char line[50] = " _____ ";
    /*METODOS*/
public:

    void DatosPrimitivos();
    void DatoEstructurado();
    void MostrarMenu();
    void RellenarMatriz(matrices [MAX][MAX]);
    void MostrarMatriz(matrices [MAX][MAX]);
    void Separacion_();
    void SumaYProducto(matrices [MAX][MAX], matrices [MAX][MAX]);
};

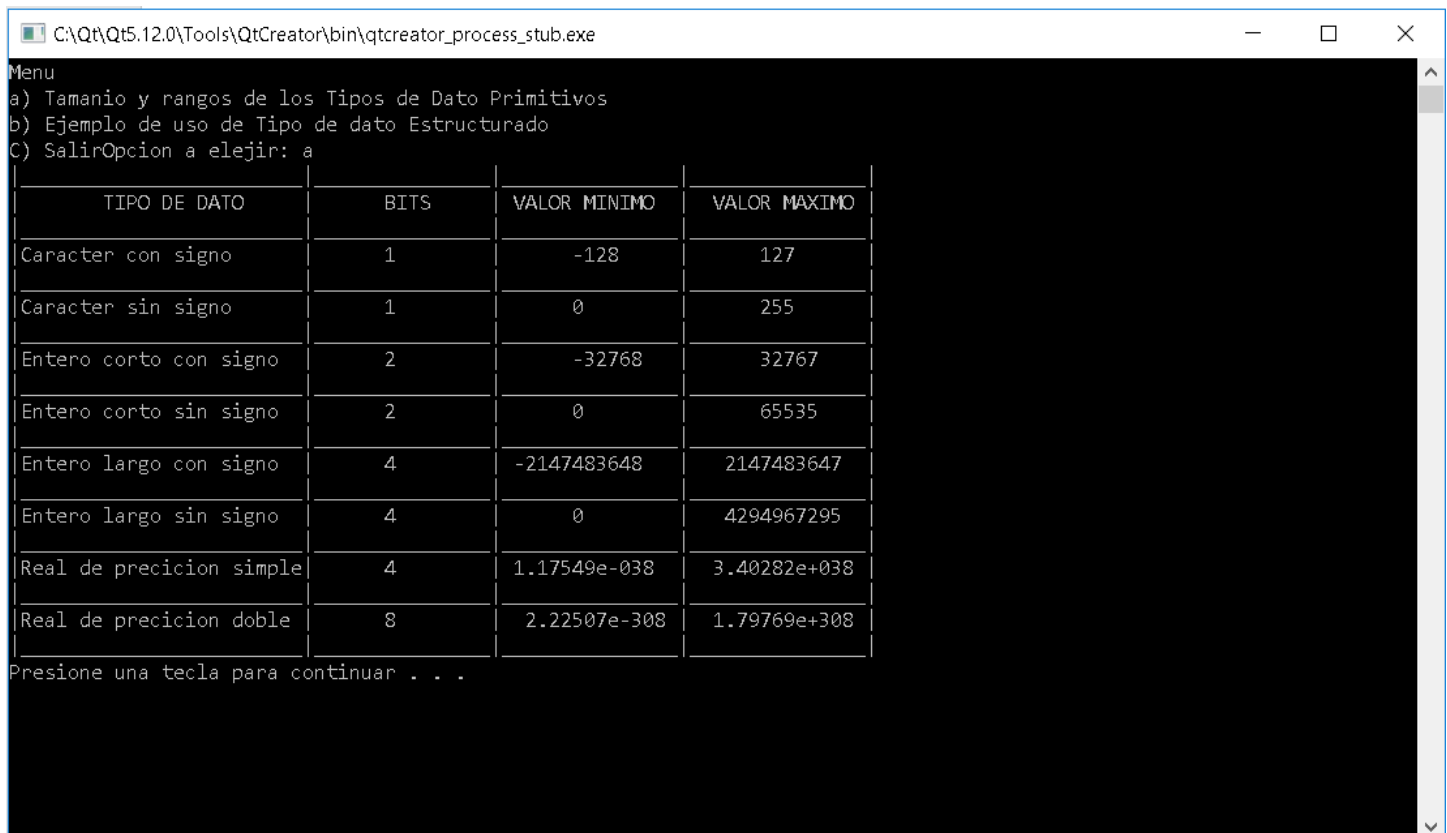
#endif // MENU_H

```

CAPTURAS DE PANTALLA



Captura del menú



Captura de la tabla de los datos primitivos

```
C:\Qt\Qt5.12.0\Tools\QtCreator\bin\qtcreator_process_stub.exe
Menu
a) Tamaño y rangos de los Tipos de Dato Primitivos
b) Ejemplo de uso de Tipo de dato Estructurado
c) SalirOpcion a elegir: b
Tamaño de Matriz: 1
Dato no valido
El tamaño de la matriz debe ser como minimo y como maximo 10
Tamaño de Matriz:
```

Captura demostrando que es invalida una matriz inferior a 3

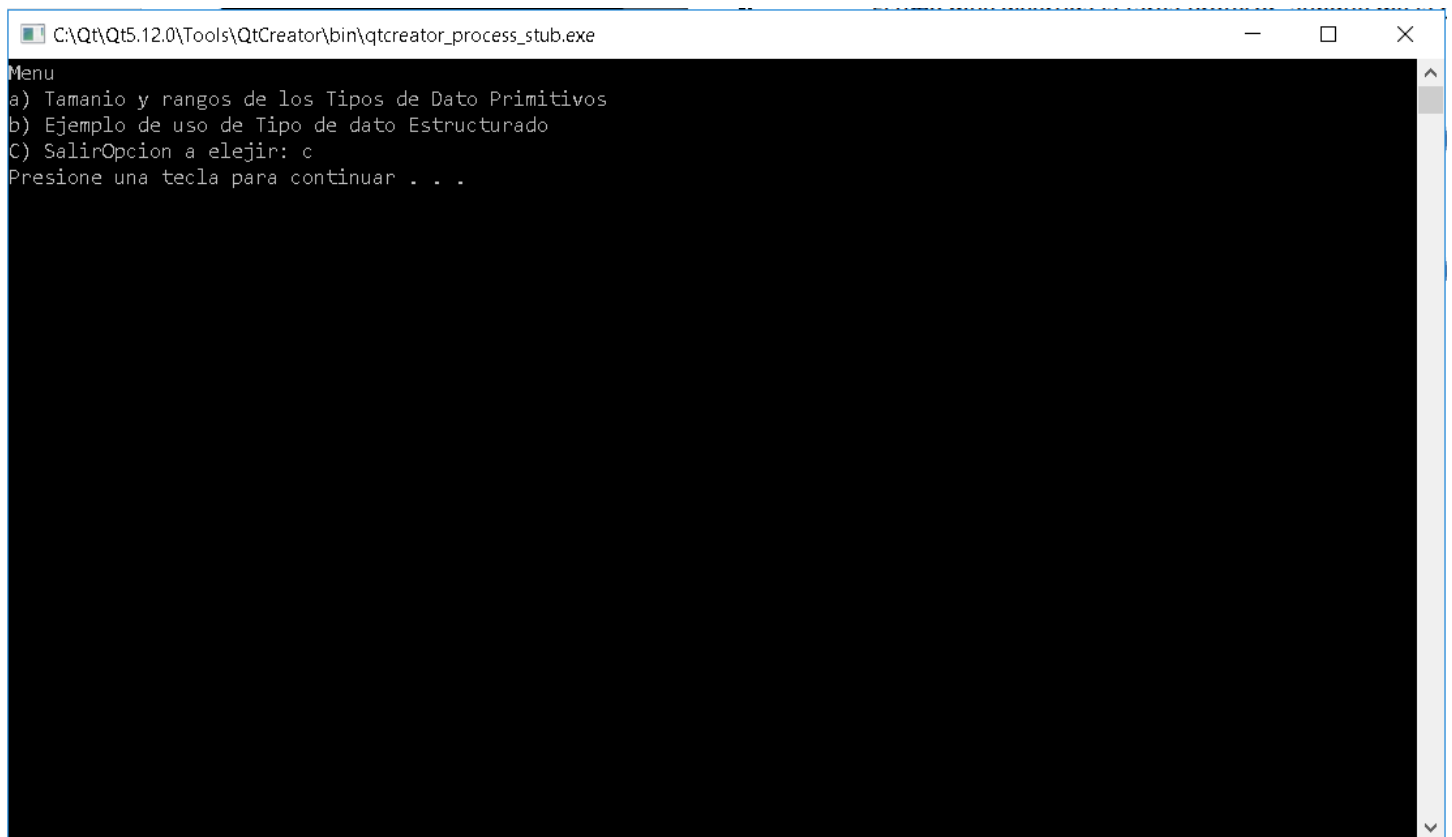
```
C:\Qt\Qt5.12.0\Tools\QtCreator\bin\qtcreator_process_stub.exe
a) Tamaño y rangos de los Tipos de Dato Primitivos
b) Ejemplo de uso de Tipo de dato Estructurado
c) SalirOpcion a elegir: b
Tamaño de Matriz: 1
Dato no valido
El tamaño de la matriz debe ser como minimo y como maximo 10
Tamaño de Matriz: 11
Dato no valido
El tamaño de la matriz debe ser como minimo y como maximo 10
Tamaño de Matriz: 3
Matriz 1
1.6 7.5 50.7
-21.9 -58.4 -83.6
-26.2 -99.5 50.8

Matriz 2
-50.7 -53.1 75.2
-83 -96.9 -48.4
-58.8 -44.1 32.9

Multiplicacion de entre Matriz 1 y Matriz 2 :
-81.12 -398.25 3812.64
1817.7 5658.96 4046.24
1540.56 4387.95 1671.32

Suma de Matriz 1 Y Matriz 2
-49.1 -45.6 125.9
-104.9 -155.3 -132
-85 -143.6 83.7
Presione una tecla para continuar . . .
```

Otra demostrando que también es invalido una superior a 10 y devolviendo el resultado de una matriz 3x3



```
C:\Qt\Qt5.12.0\Tools\QtCreator\bin\qtcreator_process_stub.exe
Menu
a) Tamaño y rangos de los Tipos de Dato Primitivos
b) Ejemplo de uso de Tipo de dato Estructurado
c) SalirOpcion a elegir: c
Presione una tecla para continuar . . .
```

Captura usando la opción Salir