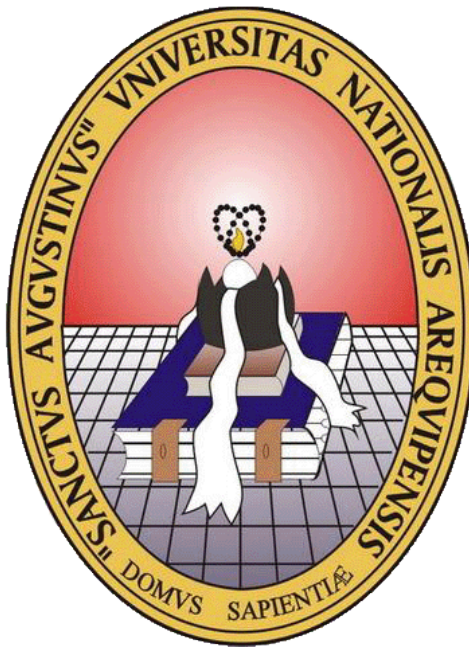


UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA
FACULTAD DE PRODUCCIÓN Y SERVICIOS
ESCUELA DE CIENCIAS DE LA COMPUTACIÓN



PRÁCTICA DE LABORATORIO 9
CURSO DE CIENCIAS DE LA COMPUTACIÓN II

ESTUDIANTE:
RUIZ MAMANI, EDUARDO GERMÁN

EMAIL: eruizm@unsa.edu.pe

CUI: 20193061

TURNO:

C

AREQUIPA- PERÚ

2021

LINK DEL REPOSITORIO: https://github.com/EGRM23/CCII_20193061.git

1. EJERCICIO

1. Se pide escribir una función utilizando plantillas que tome tres argumentos genéricos y devuelva el menor y el máximo de ellos como valor de retorno. La función debe ser capaz de dar este tipo de resultados.

- **CÓDIGO**

```
#include <iostream>
using namespace std;

//EDUARDO GERMAN RUIZ MAMANI
//20193061

template <typename R>
class dat {
    private:
        R d1, d2, d3;
    public:
        dat(const R dato1, const R dato2, const R dato3) :
        d1(dato1), d2(dato2), d3(dato3) {}
        ~dat() {}

        R getmayor() {
            R mayor;

            if (d1 > d2)
                mayor = d1;
            else
                mayor = d2;

            if (d3 > mayor)
                mayor = d3;

            return mayor;
        }

        R getmenor() {
            R menor;

            if (d1 < d2)
                menor = d1;
            else
                menor = d2;

            if (d3 < menor)
                menor = d3;

            return menor;
        }
};

int main(int argc, char *argv[]) {
    dat<int> dat1(3,4,5);
    cout << "Mayor dat1: " << dat1.getmayor() << endl;
```

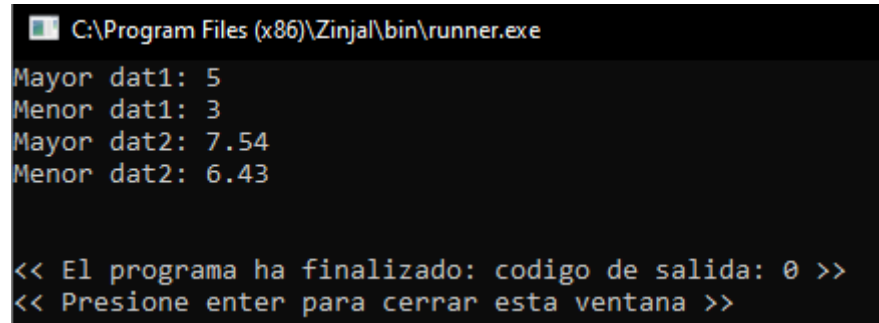
```

        cout << "Menor dat1: " << dat1.getmenor() << endl;

        dat<float> dat2(6.43,7.54,6.76);
        cout << "Mayor dat2: " << dat2.getmayor() << endl;
        cout << "Menor dat2: " << dat2.getmenor() << endl;
        return 0;
    }

```

- **CAPTURAS**



```

C:\Program Files (x86)\Zinjal\bin\runner.exe
Mayor dat1: 5
Menor dat1: 3
Mayor dat2: 7.54
Menor dat2: 6.43

<< El programa ha finalizado: codigo de salida: 0 >>
<< Presione enter para cerrar esta ventana >>

```

2. EJERCICIO

2. Se pide escribir una función utilizando plantillas que tome dos argumentos genéricos de tipo entero y flotante que devuelva las cuatro operaciones básicas.

- **CÓDIGO**

```

#include <iostream>
using namespace std;

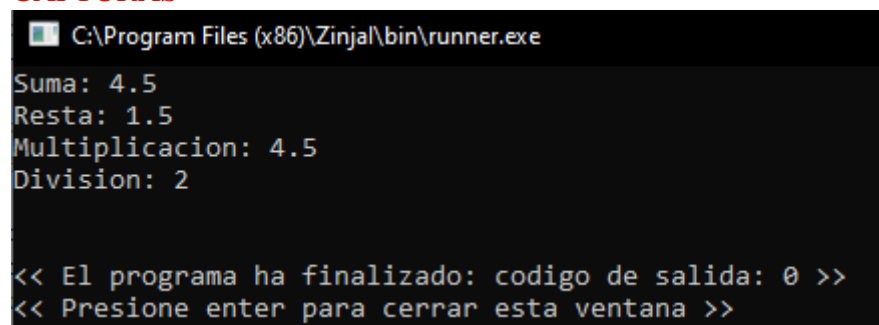
//EDUARDO GERMAN RUIZ MAMANI
//20193061

template <typename I, typename F>
void operaciones (const I num1, const F num2) {
    cout << "Suma: " << num1 + num2 << endl;
    cout << "Resta: " << num1 - num2 << endl;
    cout << "Multiplicacion: " << num1 * num2 << endl;
    cout << "Division: " << num1 / num2 << endl;
}

int main(int argc, char *argv[]) {
    operaciones<int,float> (3,1.5);
    return 0;
}

```

- **CAPTURAS**



```

C:\Program Files (x86)\Zinjal\bin\runner.exe
Suma: 4.5
Resta: 1.5
Multiplicacion: 4.5
Division: 2

<< El programa ha finalizado: codigo de salida: 0 >>
<< Presione enter para cerrar esta ventana >>

```

3. EJERCICIO

3. Se pide escribir una función utilizando plantillas que tome dos valores genéricos de tipo char y string (5 veces); char corresponde a una letra y string corresponde al apellido. El programa debe mostrar por pantalla el siguiente formato de correo electrónico: char/string@unsa.edu.pe.

- **CÓDIGO**

```
#include <iostream>

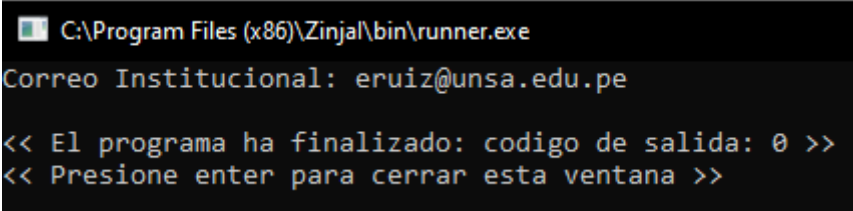
using namespace std;

//EDUARDO GERMAN RUIZ MAMANI
//20193061

template <typename C, typename S>
void corinst (const C nom, const S apellido) {
    string correo = "@unsa.edu.pe";
    correo = nom + apellido + correo;
    cout << "Correo Institucional: " << correo;
}

int main(int argc, char *argv[]) {
    corinst<char,string> ('e',"ruiz");
    return 0;
}
```

- **CAPTURAS**



```
C:\Program Files (x86)\Zinjal\bin\runner.exe
Correo Institucional: eruiz@unsa.edu.pe
<< El programa ha finalizado: codigo de salida: 0 >>
<< Presione enter para cerrar esta ventana >>
```

4. EJERCICIO

4. Implemente un programa que haga uso de plantillas para determinar el mínimo y máximo valor de un arreglo de elementos dado. Debe de existir dos funciones, la primera que retorne el mayor de los valores y la segunda que retorne el menor de los valores. Asimismo, en la función main, se hace una prueba de estas funciones, con arreglos de enteros y flotantes.

```
int ArrayEntero [5] = {10,7,2, 8, 6};
float ArrayFloat [5] = {12.1, 8.7, 5.6, 8.4, 1.2};
```

- **CÓDIGO:**

```
#include <iostream>
using namespace std;

//EDUARDO GERMAN RUIZ MAMANI
//20193061
```

```

template <typename R>
R mayorval (const R arr[], int tam) {
    R mayor = arr[0];

    for(int i = 1; i < tam; i++) {
        if (mayor < arr[i])
            mayor = arr[i];
    }

    return mayor;
}

template <typename R>
R menorval (const R arr[], int tam) {
    R menor = arr[0];

    for(int i = 1; i < tam; i++) {
        if (menor > arr[i])
            menor = arr[i];
    }

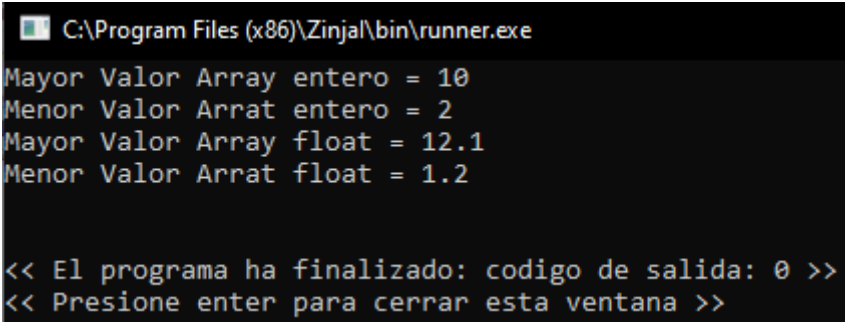
    return menor;
}

int main(int argc, char *argv[]) {
    int ArrayEntero [5] = {10, 7, 2, 8, 6};
    int tamArrayEntero = sizeof(ArrayEntero) /
sizeof(ArrayEntero[0]);
    float ArrayFloat [5] = {12.1, 8.7, 5.6, 8.4, 1.2};
    float tamArrayFloat = sizeof(ArrayFloat) /
sizeof(ArrayFloat[0]);

    cout << "Mayor Valor Array entero = " << mayorval<int>
(ArrayEntero, tamArrayEntero) << endl;
    cout << "Menor Valor Arrat entero = " << menorval<int>
(ArrayEntero, tamArrayEntero) << endl;
    cout << "Mayor Valor Array float = " << mayorval<float>
(ArrayFloat, tamArrayFloat) << endl;
    cout << "Menor Valor Arrat float = " << menorval<float>
(ArrayFloat, tamArrayFloat) << endl;
    return 0;
}

```

- **CAPTURAS**



```

C:\Program Files (x86)\Zinjal\bin\runner.exe
Mayor Valor Array entero = 10
Menor Valor Arrat entero = 2
Mayor Valor Array float = 12.1
Menor Valor Arrat float = 1.2

<< El programa ha finalizado: codigo de salida: 0 >>
<< Presione enter para cerrar esta ventana >>

```

5. EJERCICIO

5. Realizar la implementación de un programa que haga uso de plantillas, para elaborar una función que permita ordenar ascendentemente y descendentemente los elementos de un arreglo de valores enteros y otro arreglo de valores flotantes. Las funciones deben recibir como parámetros, un puntero al tipo de elemento dado, y dos enteros que indican los índices del primero y último elemento.

```
int ArrayEntero [5] = {5,7,2,8,6,1,3,4,9};  
float ArrayFloat [5] = {10.1, 8.4, 3.6, 4.4, 11.2};
```

- **CÓDIGO**

```
#include <iostream>  
using namespace std;  
  
//EDUARDO GERMAN RUIZ MAMANI  
//20193061  
  
template <typename R>  
int ubicarasc (R* lista, int ini, int fin) {  
    R aux;  
    while (ini < fin) {  
        while (lista[fin] >= lista[ini] && ini < fin)  
            fin--;  
  
        aux = lista[fin];  
        lista[fin] = lista[ini];  
        lista[ini] = aux;  
  
        while (lista[ini] <= lista[fin] && ini < fin)  
            ini++;  
  
        aux = lista[fin];  
        lista[fin] = lista[ini];  
        lista[ini] = aux;  
    }  
    return ini;  
}  
  
template <typename R>  
int ubicardesc (R* lista, int ini, int fin) {  
    R aux;  
    while (ini < fin) {  
        while (lista[fin] <= lista[ini] && ini < fin)  
            fin--;  
  
        aux = lista[fin];  
        lista[fin] = lista[ini];  
        lista[ini] = aux;  
  
        while (lista[ini] >= lista[fin] && ini < fin)  
            ini++;  
  
        aux = lista[fin];
```

```

        lista[fin] = lista[ini];
        lista[ini] = aux;
    }
    return ini;
}

template <typename R>
void ascendente (R* lista, int ini, int fin) {
    int u;
    if (ini < fin) {
        u = ubicarasc<R>(lista, ini, fin);
        ascendente<R>(lista, ini, u - 1);
        ascendente<R>(lista, u + 1, fin);
    }
}

template <typename R>
void descendente (R* lista, int ini, int fin) {
    int u;
    if (ini < fin) {
        u = ubicardesc<R>(lista, ini, fin);
        descendente<R>(lista, ini, u - 1);
        descendente<R>(lista, u + 1, fin);
    }
}

template <typename R>
void printLista(R* lista, int tam) {
    for (int i = 0; i < tam; i++)
        cout << lista[i] << " ";
    cout << endl;
}

int main(int argc, char *argv[]) {
    int ArrayEntero [9] = {5, 7, 2, 8, 6, 1, 3, 4, 9};
    int tamArrayEntero = sizeof(ArrayEntero) /
sizeof(ArrayEntero[0]);
    float ArrayFloat [5] = {10.1, 8.4, 3.6, 4.4, 11.2};
    float tamArrayFloat = sizeof(ArrayFloat) /
sizeof(ArrayFloat[0]);

    ascendente<int> (&ArrayEntero[0], 0, tamArrayEntero-1);
    cout << "Lista ArrayEntero ordenada ascendente: \n";
    printLista<int>(&ArrayEntero[0], tamArrayEntero);

    descendente<float> (&ArrayFloat[0], 0, tamArrayFloat-1);
    cout << "Lista ArrayFloat ordenada descendente: \n";
    printLista<float>(&ArrayFloat[0], tamArrayFloat);
    return 0;
}

```

- **CAPTURAS**

 C:\Program Files (x86)\Zinjal\bin\runner.exe

Lista ArrayEntero ordenada ascendente:

1 2 3 4 5 6 7 8 9

Lista ArrayFloat ordenada descendente:

11.2 10.1 8.4 4.4 3.6

<< El programa ha finalizado: codigo de salida: 0 >>

<< Presione enter para cerrar esta ventana >>