**Tarea 02 Arreglos**

Rafael De Jesús Martínez Ovando

21311231

al21311231@utcj.edu.mx

TDW51

Desarrollo de Software, UTCJ

Estructura De Datos Aplicadas

Rincón Zuluaga Joam Manuel

04/11/2023

Contenido

[Ejemplo 3](#_Toc149991283)

[Leer e imprimir un arreglo 3](#_Toc149991284)

[Suma de los elementos de un arreglo que no pertenezcan a la diagonal principal 4](#_Toc149991285)

[Triangulo de pascal 5](#_Toc149991286)

[Llenar un arreglo de forma aleatorio y que la suma de los primeros 10 elementos sean mayor que la suma de los últimos 10 5](#_Toc149991287)

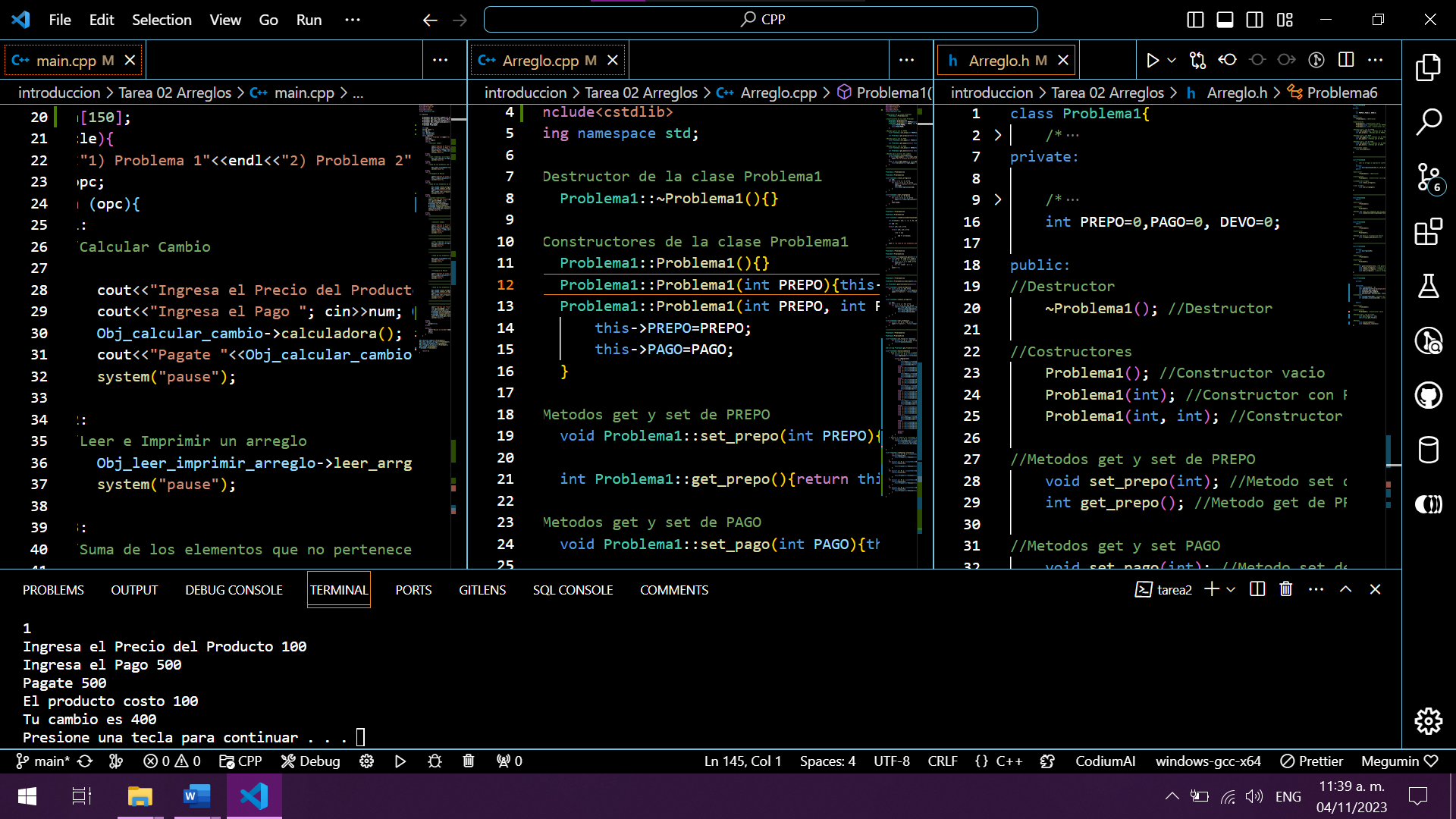
[Leer una frase y remplazar las vocales con el carácter que mas se repita 5](#_Toc149991288)

[Código del archivo main.cpp 6](#_Toc149991289)

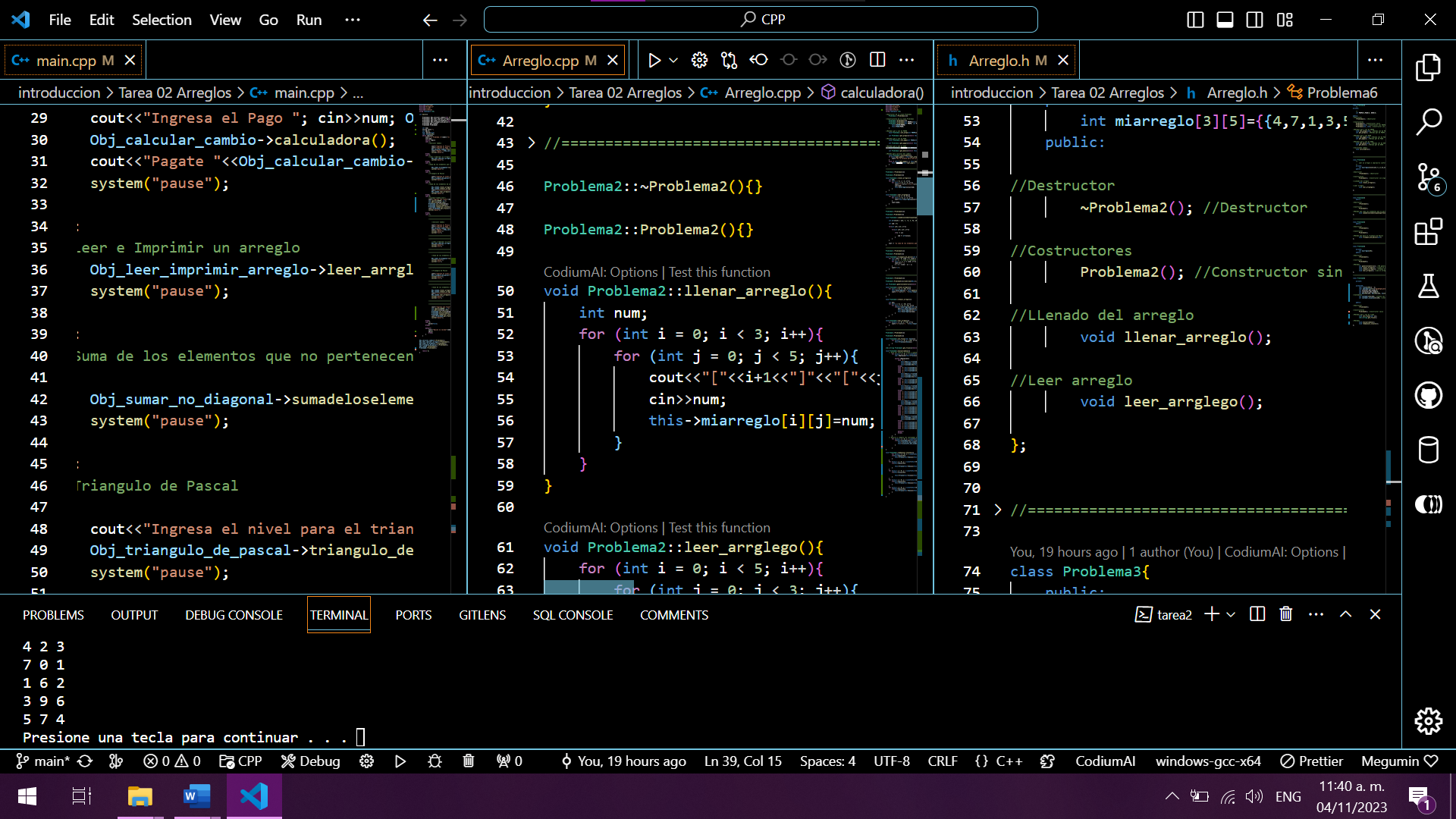
[Codigo del archivo Arreglo.cpp 11](#_Toc149991290)

[Codigo del archivo Arreglo.h 17](#_Toc149991291)

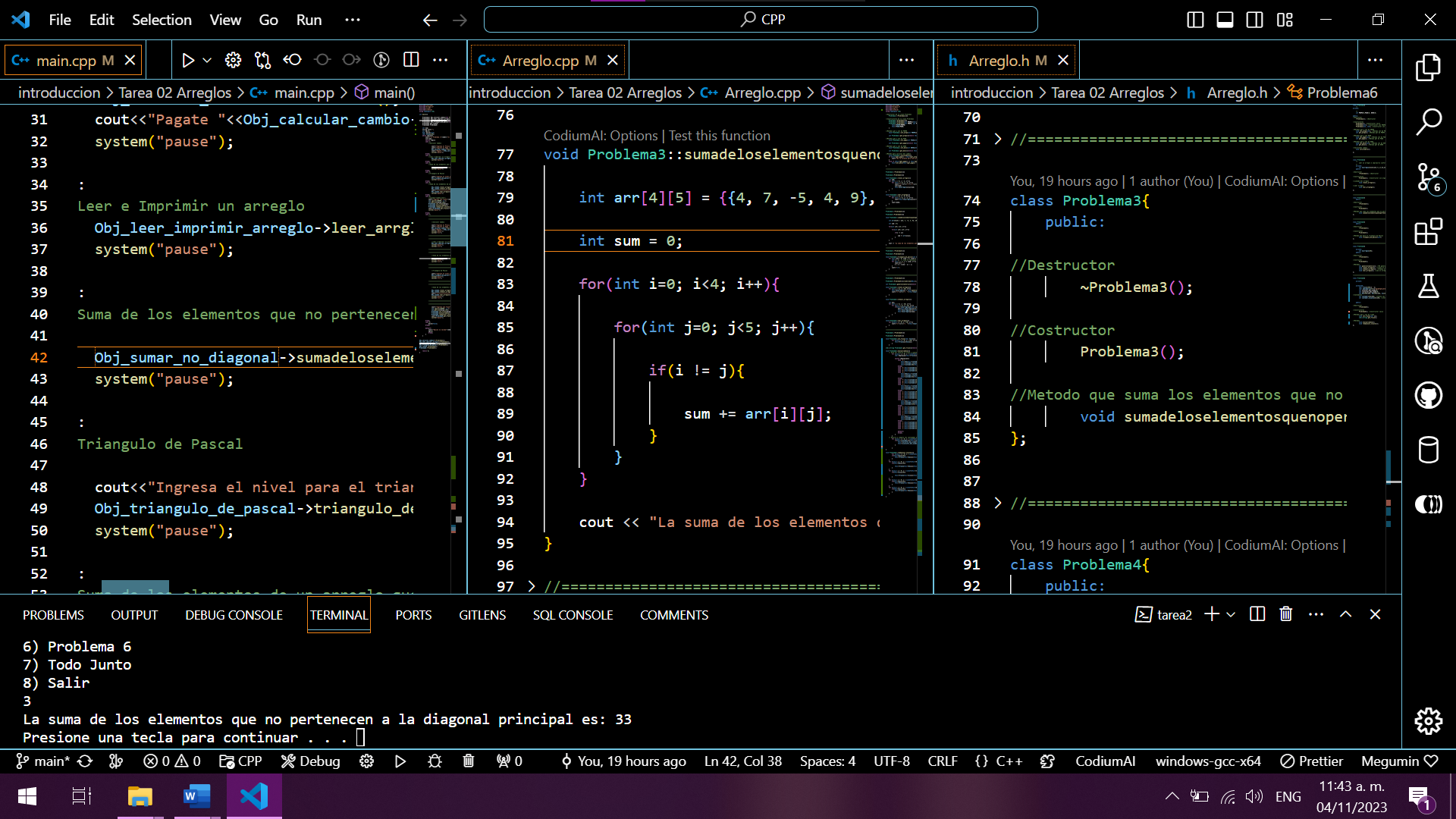
# Ejemplo



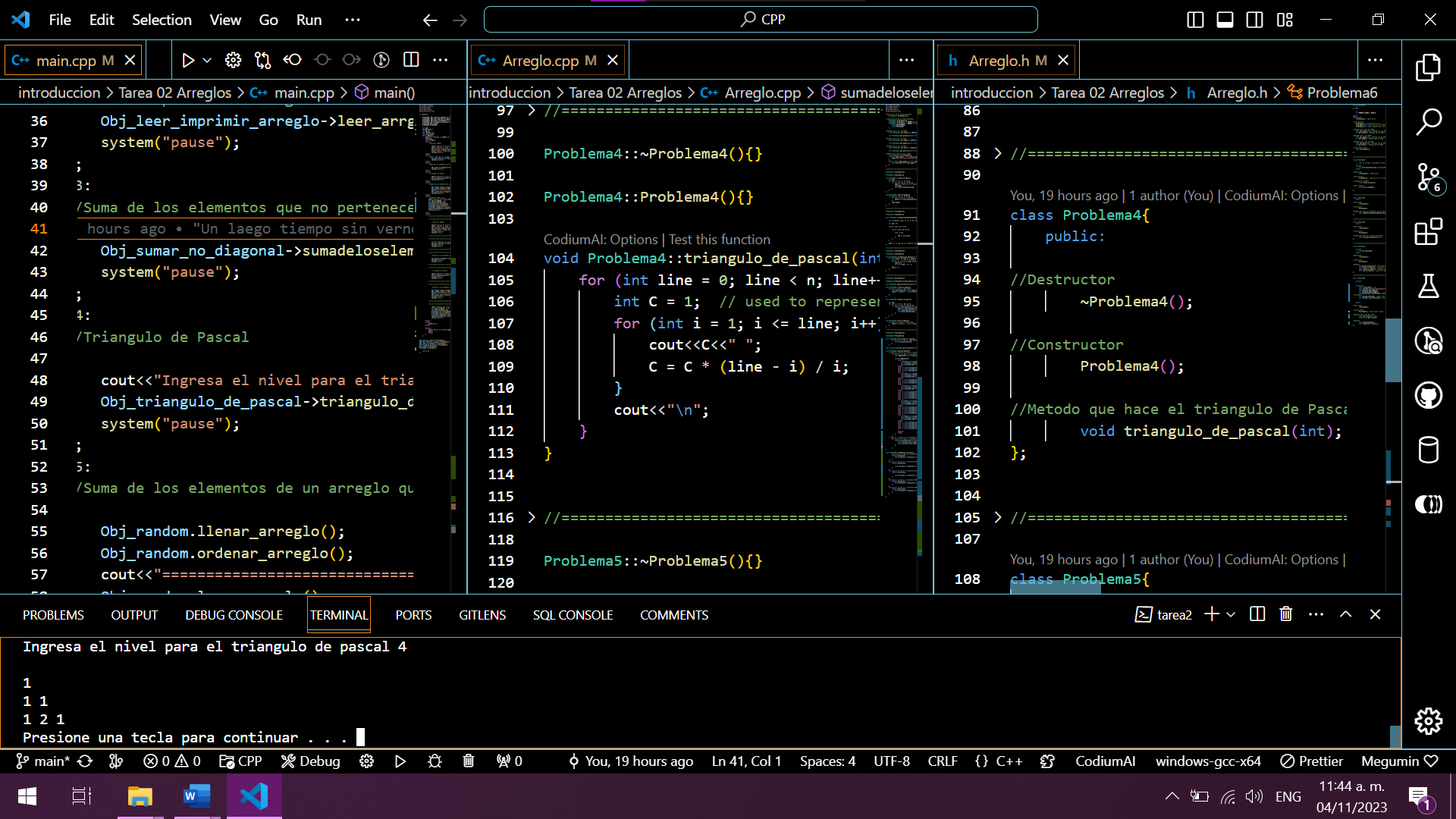
# Leer e imprimir un arreglo



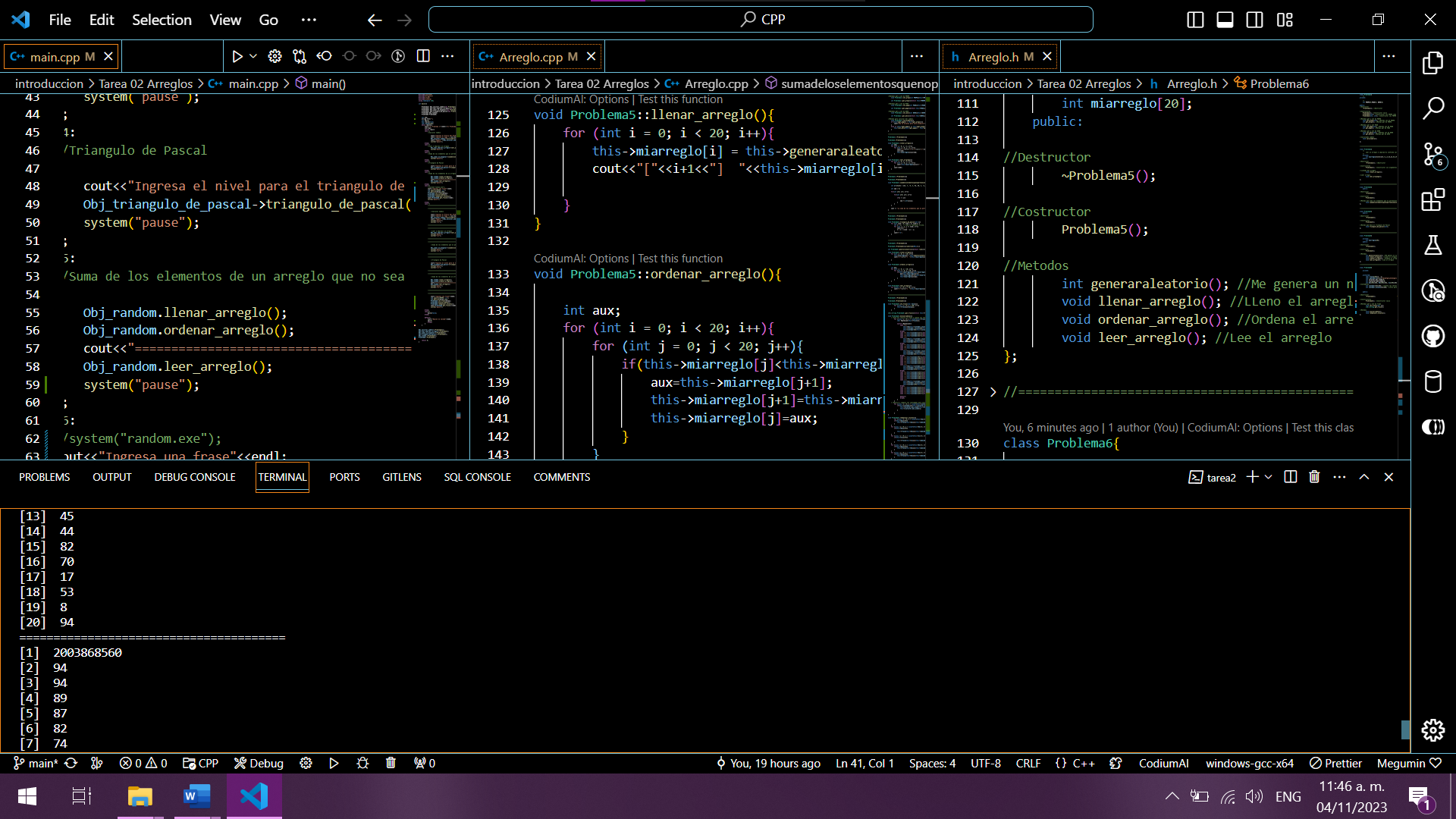
# Suma de los elementos de un arreglo que no pertenezcan a la diagonal principal



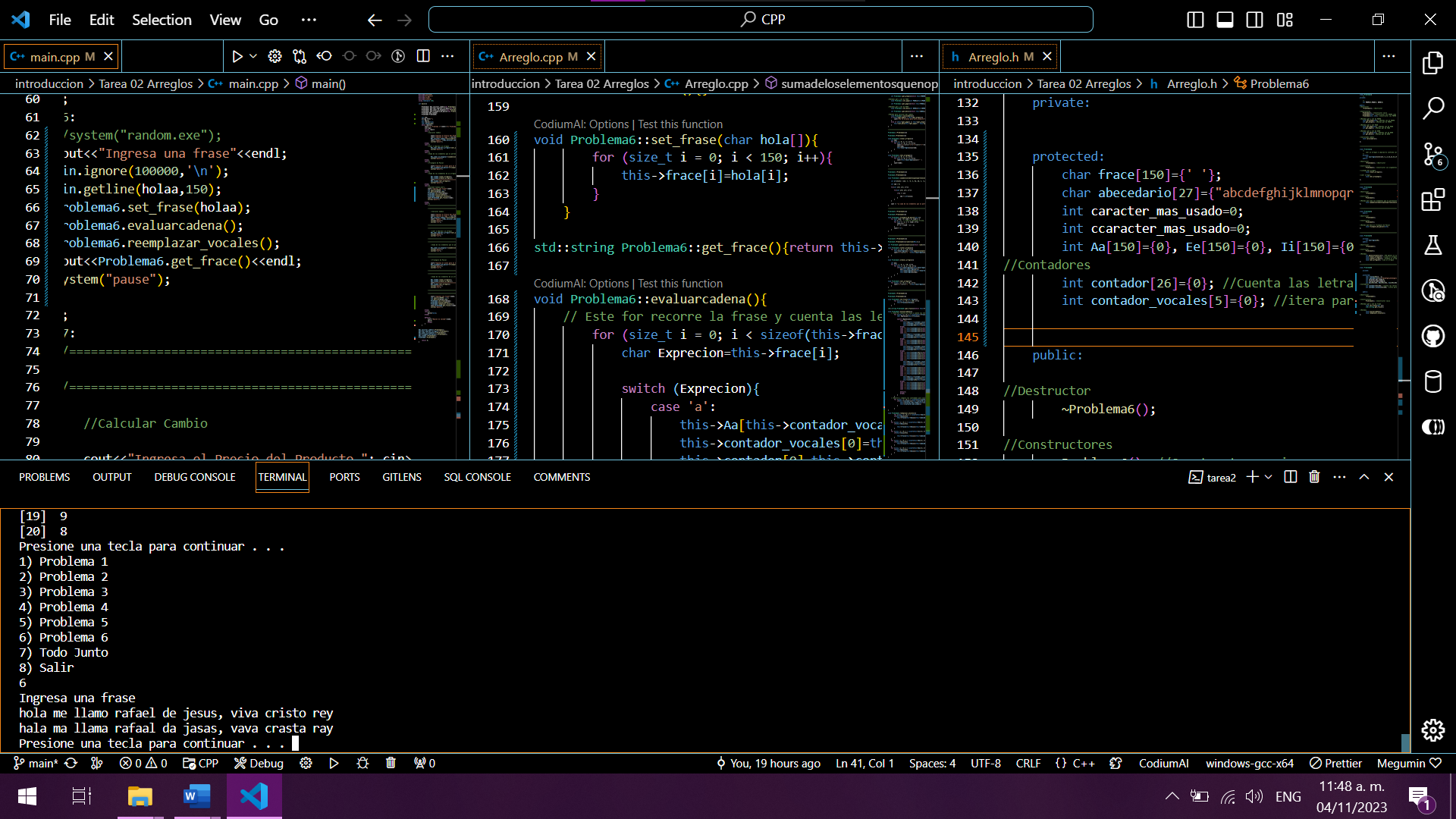
# Triangulo de pascal



# Llenar un arreglo de forma aleatorio y que la suma de los primeros 10 elementos sean mayor que la suma de los últimos 10



# Leer una frase y remplazar las vocales con el carácter que mas se repita



# Código del archivo main.cpp

#include<iostream>

#include"Arreglo.h"

#include<ctime>

#include<cstdlib>

using namespace std;

int main(){

    Problema1\* Obj\_calcular\_cambio=new Problema1();

    Problema2\* Obj\_leer\_imprimir\_arreglo=new Problema2();

    Problema3\* Obj\_sumar\_no\_diagonal=new Problema3();

    Problema4\* Obj\_triangulo\_de\_pascal=new Problema4();

    Problema5 Obj\_random;

    Problema6 Problema6;

    int num;

    string tttt="";

    int opc=0;

    bool bucle=true;

    char holaa[150];

    while (bucle){

        cout<<"1) Problema 1"<<endl<<"2) Problema 2"<<endl<<"3) Problema 3"<<endl<<"4) Problema 4"<<endl<<"5) Problema 5"<<endl<<"6) Problema 6"<<endl<<"7) Todo Junto"<<endl<<"8) Salir"<<endl;

        cin>>opc;

        switch (opc){

        case 1:

            //Calcular Cambio

                cout<<"Ingresa el Precio del Producto "; cin>>num; Obj\_calcular\_cambio->set\_prepo(num);

                cout<<"Ingresa el Pago "; cin>>num; Obj\_calcular\_cambio->set\_pago(num);

                Obj\_calcular\_cambio->calculadora();

                cout<<"Pagate "<<Obj\_calcular\_cambio->get\_pago()<<endl<<"El producto costo "<<Obj\_calcular\_cambio->get\_prepo()<<endl<<"Tu cambio es "<<Obj\_calcular\_cambio->get\_devo()<<endl;

                system("pause");

        break;

        case 2:

            //Leer e Imprimir un arreglo

                Obj\_leer\_imprimir\_arreglo->leer\_arrglego();

                system("pause");

        break;

        case 3:

            //Suma de los elementos que no pertenecen a la diagonal

                Obj\_sumar\_no\_diagonal->sumadeloselementosquenopertenecenaladiagonal();

                system("pause");

        break;

        case 4:

            //Triangulo de Pascal

                cout<<"Ingresa el nivel para el triangulo de pascal "; cin>>num;

                Obj\_triangulo\_de\_pascal->triangulo\_de\_pascal(num);

                system("pause");

        break;

        case 5:

            //Suma de los elementos de un arreglo que no sea la diagonal principal

                Obj\_random.llenar\_arreglo();

                Obj\_random.ordenar\_arreglo();

                cout<<"======================================="<<endl;

                Obj\_random.leer\_arreglo();

                system("pause");

        break;

        case 6:

            //system("random.exe");

            cout<<"Ingresa una frase"<<endl;

            cin.ignore(100000,'\n');

            cin.getline(holaa,150);

            Problema6.set\_frase(holaa);

            Problema6.evaluarcadena();

            Problema6.reemplazar\_vocales();

            cout<<Problema6.get\_frace()<<endl;

            system("pause");

        break;

        case 7:

            //===========================================================================================================================================

            //===========================================================================================================================================

                //Calcular Cambio

                cout<<"Ingresa el Precio del Producto "; cin>>num; Obj\_calcular\_cambio->set\_prepo(num);

                cout<<"Ingresa el Pago "; cin>>num; Obj\_calcular\_cambio->set\_pago(num);

                Obj\_calcular\_cambio->calculadora();

                cout<<"Pagate "<<Obj\_calcular\_cambio->get\_pago()<<endl<<"El producto costo "<<Obj\_calcular\_cambio->get\_prepo()<<endl<<"Tu cambio es "<<Obj\_calcular\_cambio->get\_devo()<<endl;

                system("pause");

                system("cls");

            //===========================================================================================================================================

            //===========================================================================================================================================

                //Leer e Imprimir un arreglo

                Obj\_leer\_imprimir\_arreglo->leer\_arrglego();

                system("pause");

                system("cls");

            //===========================================================================================================================================

            //===========================================================================================================================================

                //Suma de los elementos que no pertenecen a la diagonal

                Obj\_sumar\_no\_diagonal->sumadeloselementosquenopertenecenaladiagonal();

                system("pause");

                system("cls");

            //===========================================================================================================================================

            //===========================================================================================================================================

                //Triangulo de Pascal

                cout<<"Ingresa el nivel para el triangulo de pascal "; cin>>num;

                Obj\_triangulo\_de\_pascal->triangulo\_de\_pascal(num);

                system("pause");

                system("cls");

            //===========================================================================================================================================

            //===========================================================================================================================================

                //Suma de los elementos de un arreglo que no sea la diagonal principal

                Obj\_random.llenar\_arreglo();

                Obj\_random.ordenar\_arreglo();

                cout<<"======================================="<<endl;

                Obj\_random.leer\_arreglo();

                system("pause");

                system("cls");

            //===========================================================================================================================================

            //===========================================================================================================================================

                cout<<"Ingresa una frase"<<endl;

                cin.ignore(100000,'\n');

                cin.getline(holaa,150);

                Problema6.set\_frase(holaa);

                Problema6.evaluarcadena();

                Problema6.reemplazar\_vocales();

                cout<<Problema6.get\_frace()<<endl;

                system("pause");

        break;

        case 8:

            bucle=false;

        break;

        default:

            cout<<"Opcion no valida"<<endl;

            opc=0;

            break;

        }

    }

Obj\_calcular\_cambio->~Problema1();

Obj\_leer\_imprimir\_arreglo->~Problema2();

Obj\_sumar\_no\_diagonal->~Problema3();

Obj\_triangulo\_de\_pascal->~Problema4();

Obj\_random.~Problema5();

Problema6.~Problema6();

    return 0;

}

# Codigo del archivo Arreglo.cpp

#include<iostream>

#include"Arreglo.h"

#include<time.h>

#include<cstdlib>

using namespace std;

//Destructor de la clase Problema1

    Problema1::~Problema1(){}

//Constructores de la clase Problema1

    Problema1::Problema1(){}

    Problema1::Problema1(int PREPO){this->PREPO=PREPO;}

    Problema1::Problema1(int PREPO, int PAGO){

        this->PREPO=PREPO;

        this->PAGO=PAGO;

    }

//Metodos get y set de PREPO

    void Problema1::set\_prepo(int PREPO){this->PREPO=PREPO;}

    int Problema1::get\_prepo(){return this->PREPO;}

//Metodos get y set de PAGO

    void Problema1::set\_pago(int PAGO){this->PAGO=PAGO;}

    int Problema1::get\_pago(){return this->PAGO;}

//Metodos get y set de DEVO

    void Problema1::set\_devo(int DEVO){this->DEVO=DEVO;}

    int Problema1::get\_devo(){return this->DEVO;}

//Metodo para calcular del cambio

    void Problema1::calculadora(){

    if (this->get\_pago() < this->get\_prepo()){

        cout<<"No te alcanza para este producto"<<endl<<"No se realizo la compra"<<endl;

    }

    else if (this->get\_pago() >= this->get\_prepo()){

        this->set\_devo(this->get\_pago() - this->get\_prepo());

    }

}

//===========================================================================================================================================

//===========================================================================================================================================

Problema2::~Problema2(){}

Problema2::Problema2(){}

void Problema2::llenar\_arreglo(){

    int num;

    for (int i = 0; i < 3; i++){

        for (int j = 0; j < 5; j++){

            cout<<"["<<i+1<<"]"<<"["<<j+1<<"]"<<endl;

            cin>>num;

            this->miarreglo[i][j]=num;

        }

    }

}

void Problema2::leer\_arrglego(){

    for (int i = 0; i < 5; i++){

        for (int j = 0; j < 3; j++){

            cout<<this->miarreglo[j][i]<<" ";

        }

        cout<<endl;

    }

}

//===========================================================================================================================================

//===========================================================================================================================================

Problema3::~Problema3(){}

Problema3::Problema3(){}

void Problema3::sumadeloselementosquenopertenecenaladiagonal(){

    int arr[4][5] = {{4, 7, -5, 4, 9}, {0, 3, -2, 6, -2}, {1, 2, 4, 1, 1}, {6, 1, 0, 3, 4}};

    int sum = 0;

    for(int i=0; i<4; i++){

        for(int j=0; j<5; j++){

            if(i != j){

                sum += arr[i][j];

            }

        }

    }

    cout << "La suma de los elementos que no pertenecen a la diagonal principal es: " << sum << endl;

}

//===========================================================================================================================================

//===========================================================================================================================================

Problema4::~Problema4(){}

Problema4::Problema4(){}

void Problema4::triangulo\_de\_pascal(int n){

    for (int line = 0; line < n; line++) {

        int C = 1;  // used to represent C(line, i)

        for (int i = 1; i <= line; i++){

            cout<<C<<" ";

            C = C \* (line - i) / i;

        }

        cout<<"\n";

    }

}

//===========================================================================================================================================

//===========================================================================================================================================

Problema5::~Problema5(){}

Problema5::Problema5(){srand(time(NULL));}

int Problema5::generaraleatorio(){return rand()%100;}

void Problema5::llenar\_arreglo(){

    for (int i = 0; i < 20; i++){

        this->miarreglo[i] = this->generaraleatorio();

        cout<<"["<<i+1<<"]  "<<this->miarreglo[i]<<endl;

    }

}

void Problema5::ordenar\_arreglo(){

    int aux;

    for (int i = 0; i < 20; i++){

        for (int j = 0; j < 20; j++){

            if(this->miarreglo[j]<this->miarreglo[j+1]){

                aux=this->miarreglo[j+1];

                this->miarreglo[j+1]=this->miarreglo[j];

                this->miarreglo[j]=aux;

            }

        }

    }

}

void Problema5::leer\_arreglo(){

    for (size\_t i = 0; i < 20; i++){

        cout<<"["<<i+1<<"]  "<<this->miarreglo[i]<<endl;

    }

}

//===========================================================================================================================================

//===========================================================================================================================================

Problema6::~Problema6(){}

Problema6::Problema6(){}

void Problema6::set\_frase(char hola[]){

        for (size\_t i = 0; i < 150; i++){

            this->frace[i]=hola[i];

        }

    }

std::string Problema6::get\_frace(){return this->frace;}

void Problema6::evaluarcadena(){

    // Este for recorre la frase y cuenta las letras, las almecena en el arreglo contador

        for (size\_t i = 0; i < sizeof(this->frace); i++){

            char Exprecion=this->frace[i];

            switch (Exprecion){

                case 'a':

                    this->Aa[this->contador\_vocales[0]]=i;

                    this->contador\_vocales[0]=this->contador\_vocales[0]+1;

                    this->contador[0]=this->contador[0]+1;

                break;

                case 'b':this->contador[1]+=1;break;

                case 'c':this->contador[2]+=1;break;

                case 'd':this->contador[3]+=1;break;

                case 'e':

                    this->Ee[this->contador\_vocales[1]]=i;

                    this->contador\_vocales[1]=this->contador\_vocales[1]+1;

                    this->contador[4]=this->contador[4]+1;

                break;

                case 'f':this->contador[5]+=1;break;

                case 'g':this->contador[6]+=1;break;

                case 'h':this->contador[7]+=1;break;

                case 'i':

                    this->Ii[this->contador\_vocales[2]]=i;

                    this->contador\_vocales[2]=this->contador\_vocales[2]+1;

                    this->contador[8]=this->contador[8]+1;

                break;

                case 'j':this->contador[9]+=1;break;

                case 'k':this->contador[10]+=1;break;

                case 'l':this->contador[11]+=1;break;

                case 'm':this->contador[12]+=1;break;

                case 'n':this->contador[13]+=1;break;

                case 'o':

                    this->Oo[this->contador\_vocales[3]]=i;

                    this->contador\_vocales[3]=this->contador\_vocales[3]+1;

                    this->contador[14]=this->contador[14]+1;

                break;

                case 'p':this->contador[15]+=1;break;

                case 'q':this->contador[16]+=1;break;

                case 'r':this->contador[17]+=1;break;

                case 's':this->contador[18]+=1;break;

                case 't':this->contador[19]+=1;break;

                case 'u':

                    this->Uu[this->contador\_vocales[4]]=i;

                    this->contador\_vocales[4]=this->contador\_vocales[4]+1;

                    this->contador[20]=this->contador[20]+1;

                break;

                case 'v':this->contador[21]+=1;break;

                case 'w':this->contador[22]+=1;break;

                case 'x':this->contador[23]+=1;break;

                case 'y':this->contador[24]+=1;break;

                case 'z':this->contador[25]+=1;break;

                default:

                break;

            }

        }

    // Recorre y compara las cantidades para sacar el caracter mas usado

        for (size\_t i = 0; i < sizeof(this->contador)/4; i++){

            if(this->contador[i]>this->caracter\_mas\_usado){

                this->caracter\_mas\_usado=this->contador[i];

                this->ccaracter\_mas\_usado=i;

            }

        }

}

void Problema6::reemplazar\_vocales(){

        for (size\_t i = 0; i < sizeof(this->Aa)/4; i++){

        if(this->Aa[i]>0){

            this->frace[this->Aa[i]]=this->abecedario[this->ccaracter\_mas\_usado];

        }

    }

    for (size\_t i = 0; i < sizeof(this->Ee)/4; i++){

        if(this->Ee[i]>0){

            this->frace[this->Ee[i]]=this->abecedario[this->ccaracter\_mas\_usado];

        }

    }

    for (size\_t i = 0; i < sizeof(this->Ii)/4; i++){

        if(this->Ii[i]>0){

            this->frace[this->Ii[i]]=this->abecedario[this->ccaracter\_mas\_usado];

        }

    }

    for (size\_t i = 0; i < sizeof(this->Oo)/4; i++){

        if(this->Oo[i]>0){

            this->frace[this->Oo[i]]=this->abecedario[this->ccaracter\_mas\_usado];

        }

    }

    for (size\_t i = 0; i < sizeof(this->Uu)/4; i++){

        if(this->Uu[i]>0){

            this->frace[this->Uu[i]]=this->abecedario[this->ccaracter\_mas\_usado];

        }

    }

}

# Codigo del archivo Arreglo.h

class Problema1{

    /\*

        Construya un programa tal que dado el costo de un artículo vendido y

        la cantidad de dinero entregada por el cliente, calcule e imprima el

        cambio que se debe entregar al mismo.

    \*/

private:

    /\*

        - PREPRO es una variable de tipo real que representa el precio del producto.

        - PAGO es una variable de tipo real que representa el pago que realiza el cliente.

        - DEVO es una variable de tipo real. Almacena el cambio que se debe entregar al cliente.

    \*/

    int PREPO=0,PAGO=0, DEVO=0;

public:

//Destructor

    ~Problema1(); //Destructor

//Costructores

    Problema1(); //Constructor vacio

    Problema1(int); //Constructor con PREPO como argumento

    Problema1(int, int); //Constructor con PREPO y PAGO como argumentos

//Metodos get y set de PREPO

    void set\_prepo(int); //Metodo set de PREPO

    int get\_prepo(); //Metodo get de PREPO

//Metodos get y set PAGO

    void set\_pago(int); //Metodo set de PAGO

    int get\_pago(); //Metodo get de PAGO

//Metodos get y set de DEVO

    void set\_devo(int); //Metodo set de DEVO

    int get\_devo(); //Metodo get de DEVO

//Calculando cambio

    void calculadora();

};

//===========================================================================================================================================

//===========================================================================================================================================

class Problema2{

    /\*

        Leer un arreglo e imprimirlo conforme columnas y filas

    \*/

    private:

        int miarreglo[3][5]={{4,7,1,3,5},{2,0,6,9,7},{3,1,2,6,4}};

    public:

//Destructor

        ~Problema2(); //Destructor

//Costructores

        Problema2(); //Constructor sin argumentos

//LLenado del arreglo

        void llenar\_arreglo();

//Leer arreglo

        void leer\_arrglego();

};

//===========================================================================================================================================

//===========================================================================================================================================

class Problema3{

    public:

//Destructor

        ~Problema3();

//Costructor

        Problema3();

//Metodo que suma los elementos que no pertenecen a la diagonal principal

        void sumadeloselementosquenopertenecenaladiagonal();

};

//===========================================================================================================================================

//===========================================================================================================================================

class Problema4{

    public:

//Destructor

        ~Problema4();

//Constructor

        Problema4();

//Metodo que hace el triangulo de Pascal

        void triangulo\_de\_pascal(int);

};

//===========================================================================================================================================

//===========================================================================================================================================

class Problema5{

    private:

        int miarreglo[20];

    public:

//Destructor

        ~Problema5();

//Costructor

        Problema5();

//Metodos

        int generaraleatorio(); //Me genera un numero aleatorio

        void llenar\_arreglo(); //LLeno el arreglo conforme a los numero aleatorios que retorna el metodo anterior

        void ordenar\_arreglo(); //Ordena el arreglo de forma desendente con el metodo de burbuja para que la suma de los 10 primeros sean mayor que la suma de los 10 ultimos

        void leer\_arreglo(); //Lee el arreglo

};

//===========================================================================================================================================

//===========================================================================================================================================

class Problema6{

    private:

    protected:

        char frace[150]={' '};

        char abecedario[27]={"abcdefghijklmnopqrstuvwxyz"}; //Se usa como diccionario de referencia para el switch

        int caracter\_mas\_usado=0;

        int ccaracter\_mas\_usado=0;

        int Aa[150]={0}, Ee[150]={0}, Ii[150]={0}, Oo[150]={0}, Uu[150]={0}; //Ubicacion de las vocales

//Contadores

        int contador[26]={0}; //Cuenta las letras y las almacena en una ubicacion que pretende estar vinculado al abecedario

        int contador\_vocales[5]={0}; //itera para almecenar las ubicaciones

    public:

//Destructor

        ~Problema6();

//Constructores

        Problema6(); //Constructor vacio

//Metodos get y set de frade

        void set\_frase(char []);

        std::string get\_frace();

//Metodos

        void evaluarcadena();

        void reemplazar\_vocales();

};