Programación de computadores 2018-I/Grupo 11

Santiago Vargas Avendaño
28791735

29 de Mayo de 2018

1. La granja

En una granja se crían un número de V - Vacas, A - Aves (pollos y gallinas) y E - escorpiones. Las vacas estan encerradas en un corral de N*M metros cuadrados, las aves en un galpón y los escorpiones en vitrinas.

Problema 1. *Litos de leche*
Si una vaca necesita M metros cuadrados de pasto para producir X litros de leche, ¿cuántos litros de leche se producen en la granja?
granja.h: linea 4
granja.cpp: linea 3-9

Modelo Matemático:

$$litrosleche : \mathbb{N} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$$
$$(V, N, M, m, X) \mapsto \begin{cases} 0, & \text{si V=0;} \\ \frac{N*M*X}{m}, & \text{en otro caso;} \end{cases}$$

Codificación C++:

```
double litrosleche(int V, double M, double N, double m, double X){
    if(V==0){
        return 0;
    }else{
        return M*N*X/m;
    };
};
```

Problema 2. *Huevos por mes*
Si 1/3 de las aves que hay en la granja son gallinas, y la mitad de las gallinas

ponen 1 huevo cada 3 días y la otra mitad 1 huevo cada 5 días, ¿en un mes cuántos huevos producen? (1 mes=30 días).
granja.h: linea 5
granja.cpp: linea 11-13

Modelo Matemático:

$$huevos : \mathbb{N} \to \mathbb{N}$$
$$(A) \mapsto 8 * \frac{A}{3};$$

Codificación C++:

```
int huevos(int A){
    return 8*(A/3);
};
```

Problema 3. *Maximo de kilos de escorpiones que se puede vender*
Si los escorpiones de la granja se venden a China, y hay escorpiones de tres diferentes tamaños, P- pequeños (con un peso de 20 gramos), M- medianos (con un peso 30 gramos) y G - grandes (con un peso de 50 gramos), ¿cúantos kilos de escorpiones se pueden vender sin que decrezca la población a menos de 2/3?
granja.h: linea 6
granja.cpp: linea 15-26

Modelo Matemático:

$$kilosescorpiones : \mathbb{N} \times \mathbb{N} \times \mathbb{N} \to \mathbb{R}$$

$$(G, M, P) \mapsto \begin{cases} \frac{G+M+P}{3} * 0.05, & si\ G \geq \frac{G+M+P}{3}; \\ G * 0.05 + (\frac{G+M+P}{3} - G) * 0.03, & si\ G + M \geq \frac{G+M+P}{3}; \\ G * 0.05 + M * 0.03 + (\frac{G+M+P}{3} - G - M) * 0.02, & \text{en otro caso.} \end{cases}$$

Codificación C++:

```
double kilos_escorpiones(int G,int M, int P){
    int t = (G + M + P)* 1/3;
    double peso = 0;
    if(G>=t){
        peso = t * 0.05;
    }else if(G + M >=t){
```

```
        peso = G * 0.05 + (t-G) * 0.03;
    }else{
        peso = G * 0.05 + M * 0.03 + (t-G-M) * 0.02;
    }
    return peso;
};
```

Problema 4. *Cercado mas barato del corral*
Al granjero se le daño el corral y no sabe si volver a cercar el corral con madera, alambre de puas o poner reja de metal. Si va a cercar con madera debe poner 4 hileras de tablas, con varilla 8 hileras y con alambre solo 5 hileras, el quiere saber que es lo menos costoso para cercar si sabe que el alambre de puas vale P por metro, las tablas a Q por metro y las varillas S por metro. Dado el tamaño del corral y los precios de los elementos, ¿cual cercamiento es mas económico?
granja.h: linea 7
granja.cpp: linea 28-39

Modelo Matemático:

$$cercacorral : \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{ASCII}^*$$

$$(P, Q, S, M, N) \mapsto \begin{cases} Alambre, si\ 2*(N+M)*P*5 < 2*(N+M)*Q*4 \\ \wedge\ 2*(N+M)*P*5 < 2*(N+M)*S*8; \\ Madera, si\ 2*(N+M)*Q*4 < 2*(N+M)*P*5 \\ \wedge\ 2*(N+M)*Q*4 < 2*(N+M)*S*8; \\ Varilla, \text{en otro caso.} \end{cases}$$

Codificación C++:

```
char* cercacorral(double P, double Q, double S, double M, double N){
    double m = 2*M*N*P*5;
    double n = 2*M*N*Q*4;
    double j = 2*M*N*S*8;
    if(m<n && m<j){
        return "Alambre";
    }else if(n<m && n<j){
        return "Madera";
    }else{
        return "Varilla";
    };
};
```

2. Numéricos

Problema 5. *Potencia*
Función potencia de un entero elevado a un entero.
numericos.h: linea 4
numericos.cpp: linea 3-11

Modelo Matemático:

$$potencia : \mathbb{Z} \times \mathbb{Z} \to \mathbb{R}$$
$$(b,p) \mapsto \begin{cases} 1, & \text{si p=0;} \\ b * potencia(b,-p), & si\ p > 0; \\ \frac{1}{b*potencia(b,p-1)}, & \text{en otro caso.} \end{cases}$$

Codificación C++:

```
double potencia(int b, int p){
    if(p == 0){
        return 1;
    }else if(p > 0){
        return b*potencia(b,p-1);
    }else{
        return 1/potencia(b,-p);
    }
};
```

Problema 6. *Divisible*
Una función que determine si un número es divisible por otro.
numericos.h: linea 5
numericos.cpp: linea 13-15

Modelo Matemático:

$$divisible : \mathbb{Z} \times \mathbb{Z} \to \mathbb{B}$$
$$(n,d) \mapsto \begin{cases} Verdadero, & \text{si n \%d=0;} \\ Falso, & \text{en otro caso.} \end{cases}$$

Codificación C++:

```
bool divisible(int n, int d){
    return n%d == 0;
};
```

Problema 7. *Primo*
Determinar si un número es primo.
numericos.h: linea 7
numericos.cpp: linea 17-25

Modelo Matemático:

$$primo : \mathbb{N} \to \mathbb{B}$$
$$(n) \mapsto \begin{cases} Verdadero, & si\ n\,\%x_i \neq 0, \forall_{i=2}^{\frac{n}{2}} x_i \\ falso, & \text{en otro caso;} \end{cases}$$

Codificación C++:

```
bool primo(int n){
    bool c=true;
    for(int i=2;i<n;i++){
        if(n%i==0){
            c = false;
        }
    }
    return c;
};
```

Problema 8. *Primo relativo*
Dados dos naturales, determinar si son primos relativos.
numericos.h: linea 8
numericos.cpp: linea 35-37

Modelo Matemático:

$$mcd : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$$
$$(d, b) \mapsto \begin{cases} d, & si\ b = 0; \\ mcd(b, d\,\%b), & \text{en otro caso;} \end{cases}$$

numericos.h: linea 7
numericos.cpp: linea 27-33

$$primorelativo : \mathbb{N} \times \mathbb{N} \to \mathbb{B}$$
$$(a, b) \mapsto \begin{cases} Verdadero, & si\ mcd(a, b) = 1; \\ Falso, & \text{en otro caso;} \end{cases}$$

Codificación C++:

```
int mcd(int a, int b){
    if(b==0){
        return a;
    }else{
        return mcd(b,a%b);
    }
};

bool primorelativo(int a, int b){
    return mcd(a,b)==1;
};
```

Problema 9. *Multiplo*
Determinar si un número es múltiplo de la suma de otros dos números.
numericos.h: linea 9
numericos.cpp: linea 39-41

Modelo Matemático:

$$multiplo : \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}, \to \mathbb{B}$$
$$(a,b,m) \mapsto \begin{cases} Verdadero, & \text{si m \%(a+b)=0;} \\ Falso, & \text{en otro caso.} \end{cases}$$

Codificación C++:

```
bool multiplo(int m, int a, int b){
    return m%(a+b)==0;
};
```

Problema 10. *Calcular un polinomio cuadratico*
Dados los coeficientes de un polinomio de grado dos, evaluar el polinomio en un punto dado.
numericos.h: linea 10
numericos.cpp: linea 43-45

Modelo Matemático:

$$poldado : \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$$
$$(a,b,c,x) \mapsto a * x^2 + b * x + c;$$

Codificación C++:

```
double poldado(double a, double b, double c, double x){
    return a*x*x + b*x + c;
};
```

Problema 11. *Coeficiente lineal de la derivada cuadratica*
Dados los coeficientes de un polinomio de grado dos, calcular coeficiente lineal de la derivada.
numericos.h: linea 11
numericos.cpp: linea 47-49

Modelo Matemático:

$$pollinder : \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$$
$$(a, b, c) \mapsto 2 * a;$$

Codificación C++:

```
double pollinder(double a, double b, double c){
    return 2*a;
};
```

Problema 12. *Calcular la derivada de un polinomio cuadratico*
Dados los coeficientes de un polinomio de grado dos, calcular la derivada en un punto dado.
numericos.h: linea 12
numericos.cpp: linea 51-53

Modelo Matemático:

$$poldadoder : \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$$
$$(a, b, c, x) \mapsto 2 * a * x + b;$$

Codificación C++:

```
double poldadoder(double a, double b, double c, double x){
    return 2*a*x + b;
};
```

Problema 13. *Pertenece a Fibonacci*
Dado un natural, determinar si es un número de fibonacci o no.
numericos.h: linea 13
numericos.cpp: linea 55-65

Modelo Matemático:

$$esfibo : \mathbb{N} \to \mathbb{B}$$
$$(n) \mapsto \begin{cases} Verdadero, si\ n \in \text{en algun momento a la sucesion} \\ \qquad\qquad c = a + b\ \wedge a = b \wedge b = c, con\ a = 1\ y\ b = 1; \\ Falso, \text{en otro caso}; \end{cases}$$

Codificación C++:

```
bool esfibo(int n){
    int a=1;
    int b=1;
    int c;
    do{
        c=b+a;
        a=b;
        b=c;
    }while(n>c);
    return n == c;
}
```

3. Geométricos

Problema 14. *Las rectas son paralelas, perpendiculares o ninguna*
Dados la pendiente y el punto de corte de dos rectas, determinar si son paralelas, perpendiculares o ninguna de las anteriores.
geometricos.h: linea 4
geometricos.cpp: linea 5-13

Modelo Matemático:

$$parpennin : \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \rightarrow \mathbb{ASCII}^*$$

$$(a, ya, b, yb) \mapsto \begin{cases} son\ paralelas, & si\ a = b \land ya \neq yb; \\ son\ perpendiculares, & si\ a * b = -1; \\ \text{no son paralelas ni perpendiculares}, & \text{en otro caso;} \end{cases}$$

Codificación C++:

```
char* par_per_nin(double a, double ya, double b, double yb){
    if(a==b && ya!=yb){
        return "son paralelas";
    }else if(a*b == -1){
        return "son perpendiculares";
    }else{
        return "no son paralelas ni perpendiculares";
    };
};
```

Problema 15. *Punto de intersección de dos rectas*
Dados la pendiente y el punto de corte de dos rectas, determinar el punto de intersección.
geometricos.h: linea 5
geometricos.cpp: linea 15-20

Modelo Matemático:

$$punint : \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \rightarrow \mathbb{ASCII}^*$$
$$(a, ya, b, yb) \mapsto (\frac{yb - ya}{a - b},\ a * \frac{yb - ya}{a - b} + ya;)$$

Codificación C++:

```
double* pun_int(double a, double ya, double b, double yb){
    double* x = new double[2];
    x[0]= (yb-ya)/(a-b);
    x[1]= a*x[0] + ya;
    cout << x[0] << " , " << x[1] << endl;
}
```

Problema 16. *Area de un triangulo circunscrito*
Calcular área del triángulo que circunscribe el circulo (triangulo afuera)
geometricos.h: linea 6
geometricos.cpp: linea 22-24

Modelo Matemático:

$$areatricir : \mathbb{R} \rightarrow \mathbb{R}$$
$$(r) \mapsto 3 * r^2 * \sqrt{3};$$

Codificación C++:

```
double area_tri_cir(double r){
    return 3*r*r*1.7320508075;
};
```

Problema 17. *Areas y perimetros de poligonos circunscritos e inscritos*
Área y perímetro de cuadrado, pentágono y hexágono dentro (inscrito en círculo) y afuera (inscribiendo al círculo)
geometricos.h: linea 7
geometricos.cpp: linea 26-41

Modelo Matemático:

$$perarepoli : \mathbb{R} \to \mathbb{R}^*$$

$$(r) \mapsto \begin{cases} 4 * r * \sqrt{2}, & \text{Perimetro del cuadrado inscrito} \\ 2 * r^2, & \text{Area cuadrado inscrito} \\ 8 * r, & \text{Perimetro cuadrado circunscrito} \\ 4 * r^2, & \text{Area cuadrado circunscrito} \\ 10 * r + sen(36), & \text{Perimetro del pentagono inscrito} \\ 5 * r^2 + sen(36) + tan(54), & \text{Area pentagono inscrito} \\ 10 * r + tan(36), & \text{Perimetro pentagono circunscrito} \\ 5 * r^2 + tan(36), & \text{Area pentagono circunscrito} \\ 6 * r, & \text{Perimetro del hexagono inscrito} \\ \frac{3*r^2*\sqrt{3}}{2}, & \text{Area hexagono inscrito} \\ 4 * r * \sqrt{3}, & \text{Perimetro hexagono circunscrito} \\ 2 * r^2 * \sqrt{3}, & \text{Area hexagono circunscrito} \end{cases}$$

Codificación C++:

Problema 18. *Telaraña*
Si una araña utiliza un patrón hexagonal para su telaraña, y cada hexágono está separado del otro por 1cm, y la araña quiere hacer una telaraña de Pi*r^2, ¿qué cantidad de telaraña requiere la araña?
geometricos.h: linea 9
geometricos.cpp: linea 51-53

Modelo Matemático:

$$fact : \mathbb{N} \to \mathbb{N}$$

$$(d) \mapsto \begin{cases} 1, & si\ n = 1; \\ n * fact(n-1), & \text{en otro caso;} \end{cases}$$

geometricos.h: linea 8
geometricos.cpp: linea 43-49

$$telarana : \mathbb{R} \to \mathbb{R}$$

$$(r) \mapsto 6 * (fact(r) + r);$$

Codificación C++:

```
int fact(int n){
    if(n==1){
        return 1;
    }else{
        return n*fact(n-1);
    }
};

double telarana(double r){
    return 6*(fact(r)+r);
}
```

4. Otros

Problema 19. *Podar arboles en la UN*
Si en la UN están podando árboles y cada rama tiene P hojas, y a cada árbol le quitaron K ramas, ¿cuántos árboles se deben podar para obtener T hojas?
otros.h: linea 4
otros.cpp: linea 3-9

Modelo Matemático:

$$arboles : \mathbb{N} \times \mathbb{N} \times \mathbb{N} \to \mathbb{N}$$
$$(t, p, k) \mapsto \lceil \frac{t}{p * k} \rceil$$

Codificación C++:

```
int arboles(int t, int p, int k){
    if(t%(p*k)!=0){
        return t/(p*k) + 1;
    }else{
        return t/(p*k);
    }
};
```

Problema 20. *Intereses simple y compuesto* Si un amigo, no tan amigo, me presta K pesos a i pesos de interés diario, ¿cuánto le pagaré en una semana si el interés es simple?(a) y ¿si el interés es compuesto?(b)
a. **otros.h:** linea 5
otros.cpp: linea 11-13
b. **otros.h:** linea 6
otros.cpp: linea 15-21

Modelo Matemático:

$$a.\ insimple : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$$
$$(K, i) \mapsto K * (1 + (i * 7));$$

$$b.\ incompuesto : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$$
$$(K, i) \mapsto K * (1 + i)^7;$$

Codificación C++:

```
a. double in_simple(double K, double i){
       return K *(1.0+ i*7.0);
   };

b. double in_compuesto(double K, double i){
       double d = i+1;
       for(int p = 1; p<7; p++){
          d = d * (i+1);
       }
       return K*d;
   };
```

Problema 21. *Fichas de lego y posibilidad de armado*
Un niño se la pasó jugando con fichas de lego, tenia dos tipos de fichas de lego, fichas de cuadros de 1x1 (rojas) y fichas azules de 2x1 (azules), y le dieron una base de 1xn cuadrito, ¿cuantas formas diferentes puede construir con esa hilera?(a) y ¿si le dan una amarilla de 1x3?(b)
a. **otros.h:** linea 8
otros.cpp: linea 31-33
b. **otros.h:** linea 10
otros.cpp: linea 26-

Modelo Matemático:

$$fibo : \mathbb{N} \to \mathbb{N}$$
$$(n) \mapsto \begin{cases} n, & si\ n < 2; \\ fibo(n-1) + fibo(n-2), & \text{en otro caso;} \end{cases}$$

otros.h: linea 7
otros.cpp: linea 23-29

$$a.\ pos2fich : \mathbb{N} \to \mathbb{N}$$
$$(n) \mapsto fibo(n+1);$$

$$tribo : \mathbb{N} \to \mathbb{N}$$
$$(0) \mapsto \begin{cases} 0, & si\ n < 3; \\ 1, & si\ n = 3; \\ tribo(n-1) + tribo(n-2) + tribo(n-3), & \text{en otro caso;} \end{cases}$$

otros.h: linea 9

otros.cpp: linea 35-41

$$b.\ pos3fich : \mathbb{N} \to \mathbb{N}$$
$$(n) \mapsto tribo(n+3);$$

Codificación C++:

```
a. int fibo(int n){
          if(n<2){
           return n;
       }else{
           return fibo(n-1)+fibo(n-2);
       }
   };

   int pos_2fich(int n){
       return fibo(n+1);
   };

b. int tribo(int n){
       if(n<2){
           return n;
       }else{
           return tribo(n-1)+tribo(n-2)+tribo(n-3);
       }
   };

   int pos_3fich(int n){
       return tribo(n+3);
   };
```

5. Arreglos

Problema 22. *Criba de Eratostenes*
Implementar la criba de Eratostenes para calcular los números primos en el

rango 1 a n, donde n es un número natural dado por el usuario.
arreglos.h: linea 4
arreglos.cpp: linea 5-15

Modelo Matemático:

$$arreglocriba\_erast : \mathbb{N} \to \mathbb{B}^*$$
$$(n) \mapsto \begin{cases} 1, & \nexists x : \forall_{i=0}^{n-1} i \ B_{[i]} \ mod \ x \neq 0 \\ 0, & \text{en otro caso;} \end{cases}$$

$$escribircriba\_erast : \mathbb{B}^* \times \mathbb{N} \to \mathbb{ASCII}^*$$
$$(A, n) \mapsto \text{números primos desde 1 hasta n}$$

arreglos.h: linea 5
arreglos.cpp: linea 17-23

Codificación C++:

```
bool* arreglocriba_erast(int n){
    bool* criba_erast=new bool[n];
    for(int i=1;i<=n;i++){
            criba_erast[i]=true;
    }for(int i = 2; i <= n; i++){
        for(int j = 2; i*j <=n; j++){
                criba_erast[i*j] = false;
        }
    }
    return criba_erast;
}
void escribircribaerast(bool *a, int n){
    for(int i = 2; i<=n;i++){
            if(a[i]){
                cout<<i<<" ";
            }
    };
};
```

Problema 23. *Suma de un arreglo*
Desarrollar un algoritmo que calcule la suma de los elementos de un arreglo de números enteros (reales).
arreglos.h: linea 6
arreglos.cpp: linea 25-31

Modelo Matemático:

$$suma : \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}$$
$$(A, n) \mapsto \sum_{i=1}^{n} A_i$$

Codificación C++:

```
double suma(int* x, int n){
    int ac = 0;
    for(int i = 0; i<n ; i++){
        ac = x[i] + ac;
    };
    return ac;
};
```

Problema 24. *Promedio de un arreglo*
Desarrollar un algoritmo que calcule el promedio de un arreglo de enteros (reales).
arreglos.h: linea 7
arreglos.cpp: linea 33-35

Modelo Matemático:

$$promedio : \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}$$
$$(A, n) \mapsto \frac{1}{n} * \sum_{i=1}^{n} A_i$$

Codificación C++:

```
double promedio(int* x, int n){
    return suma(x,n)/n;
};
```

Problema 25. *Producto entre dos arreglos*
Desarrollar un algoritmo que calcule el producto de dos arreglos de números enteros (reales) de igual tamaño. Sean v=(v1,v2,..., vn) y w=(w1,w2,..., wn) dos arreglos, el producto de v y w (notado v.w) es el número: v1 * w1 + v2*w2 +...+ vn * wn
arreglos.h: linea 8
arreglos.cpp: linea 37-43

Modelo Matemático:

$$producto : \mathbb{Z}^* \times \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}$$
$$(v, w, n) \mapsto \sum_{i=1}^{n} x_i, \ con \ x_i = v_i * w_i;$$

Codificación C++:

```
double producto(int* v, int* w, int n){
    int* x = new int[n];
    for(int i=0; i<n; i++){
        x[i] = v[i] * w[i];
    };
    return suma(x,n);
};
```

Problema 26. *Minimo de un arreglo*
Desarrollar un algoritmo que calcule el mínimo de un arreglo de números enteros (reales).
arreglos.h: linea 9
arreglos.cpp: linea 45-54

Modelo Matemático:

$$minimoarr : \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}$$
$$(A, n) \mapsto \begin{cases} x_1, & si \ n = 1; \\ M, & \text{donde M es minimoarr(a,n-1) y M} < x_n \\ x_n, & \text{en otro caso;} \end{cases}$$

Codificación C++:

```
double minimoarr(int* x, int n){
    if(n == 1){
        return x[0];
    };
    int M = minimoarr(x, n - 1);
    if(M < x[n - 1]){
        return M;
    };
    return x[n - 1];
};
```

Problema 27. *Maximo de un arreglo*
Desarrollar un algoritmo que calcule el máximo de un arreglo de números enteros (reales).
arreglos.h: linea 10
arreglos.cpp: linea 56-65

Modelo Matemático:

$$maximoarr : \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}$$
$$(A, n) \mapsto \begin{cases} x_1, & si\ n = 1; \\ M, & \text{donde M es maximoarr(a,n-1) y M} > x_n \\ x_n, & \text{en otro caso;} \end{cases}$$

Codificación C++:

```
double maximoarr(int* x, int n){
    if(n == 1){
        return x[0];
    };
    int M = maximoarr(x, n - 1);
    if(M > x[n - 1]){
        return M;
    };
    return x[n - 1];
};
```

Problema 28. *Producto directo entre dos arreglos*
Desarrollar un algoritmo que calcule el producto directo de dos arreglos de enteros (reales) de igual tamaño. Sean v=(v1,v2,..., vn) y w=(w1,w2,..., wn) dos arreglos, el producto directo de v y w (notado v*w) es el vector: [v1 * w1 , v2*w2 ,..., vn * wn]
arreglos.h: linea 11
arreglos.cpp: linea 67-73

Modelo Matemático:

$$productodirecto : \mathbb{Z}^* \times \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}^*$$
$$(w, v, n) \mapsto x_i,\ \forall_{i=1}^{n} x_i, donde\ x_i = w_i * v_i;$$

Codificación C++:

```
double* producto_directo(int* v, int* w, int n){
```

```
    double* x = new double[n+1];
    x[0] = n+1;
    for(int i=1; i<=n; i++){
        x[i] = v[i-1] * w[i-1];
    }return x;
};
```

Problema 29. *Mediana de un arreglo*
Desarrollar un algoritmo que determine la mediana de un arreglo de enteros (reales). La mediana es el número que queda en la mitad del arreglo despues de ser ordenado.
arreglos.h: linea 14
arreglos.cpp: linea 98-105

Modelo Matemático:

$$posmaximo : \mathbb{Z}^* \times \mathbb{N} \to \mathbb{Z}$$
$$(A,n) \mapsto \begin{cases} 1, & si\ n = 1; \\ k, & \text{donde k=posmaximo(A,n-1)} \wedge A_k > A_n; \\ n, & \text{en otro caso} \end{cases}$$

arreglos.h: linea 12
arreglos.cpp: linea 75-84

$$ordenar : \mathbb{Z}^* \times \mathbb{N} \to \mathbb{Z}^*$$
$$(A,n) \mapsto \begin{cases} A, & si\ n = 1; \\ ordear(A, n-1), & \text{donde k=posmaximo(A,n)} \wedge \\ & (t = x_k) \wedge (x_k = x_n) \wedge (x_n = t). \end{cases}$$

arreglos.h: linea 13
arreglos.cpp: linea 86-96

$$mediana : \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}$$
$$(A,n) \mapsto ordenar(A,n) \begin{cases} A_{\frac{n}{2}}, & si\ n\,\%2 = 0; \\ \frac{A_{\frac{n}{2}} + A_{\frac{n}{2}+1}}{2}, & \text{en otro caso;} \end{cases}$$

Codificación C++:

```
int pos_max(int* x, int n){
    if(n == 1){
```

```
        return 0;
    };
    int k = pos_max(x, n - 1);
    if(x[k] > x[n - 1]){
      return k;
    };
    return n - 1;
};

int* ordenar(int* x, int n){
    if(n == 1){
        return x;
    }else{
        int k = posmaximo(x, n);
        int t = x[k];
        x[k] = x[n-1];
        x[n-1] = t;
        return ordenar(x, n-1);
    };
};

double mediana(int* x, int n){
    ordenar(x,n);
    if(n%2!=0){
        return x[n/2];
    }else{
        return (x[n/2]+x[n/2-1])/2.0;
    };
};
```

Problema 30. *Ubicar los ceros al final de un arreglo*
Hacer un algoritmo que deje al final de un arreglo de números todos los ceros que aparezcan en dicho arreglo.
arreglos.h: linea 15
arreglos.cpp: linea 107-

Modelo Matemático:

$cerosfinal : \mathbb{Z}^* \times \mathbb{N} \to \mathbb{Z}^*$

$$(A, n) \mapsto \begin{cases} cerosfinal(A, n-1) & \text{, si 1<n donde } \forall_{i=0}^{n-1} i \text{ si } A_{[i]} = 0 \\ & \text{entonces } A_{[i+1]} = 0 \ y \ A_{[i]} = A_{[i+1]} \end{cases}$$

Codificación C++:

```
int* cerosfinal(int* a, int n){
    int temp=0;
    if(n>1){
        for(int i=0;i<n-1;i++){
            if(a[i]==0){
                temp=a[i];
                a[i]=a[i+1];
                a[i+1]=temp;
            }
        }
        return cerosfinal(a, n-1);
    };
};
```

Problema 31. *De binarios a decimal*
Suponga que un arreglo de enteros esta lleno de unos y ceros y que el arreglo representa un número binario al revés. Hacer un algoritmo que calcule los números en decimal que representa dicho arreglo de unos y ceros.
arreglos.h: linea 16
arreglos.cpp: linea 122-132

Modelo Matemático:

$$bin2dec : \mathbb{Z}^* \times \mathbb{N} \to \mathbb{Z}$$
$$(x, n) \mapsto \sum_{i=0}^{n} z_i, \ con \ z_i = x_i * 2^i;$$

Codificación C++:

```
int bin2dec(int* x, int n){
    int* y = new int[n];
    int a = 1;
    y[0] = a;
    for(int i = 1; i< n; i++){
        a = a*2;
        y[i] = a;
    }
    double z = producto(x,y,n);
    return z;
};
```

Problema 32. *De decimal a binarios*
Hacer un algoritmo que dado un número entero no negativo, cree un arreglo de

unos y ceros que representa el número en binario al revés.
arreglos.h: linea 18
arreglos.cpp: linea 143-152

Modelo Matemático:

$$tama : \mathbb{N} \to \mathbb{N}$$
$$(n) \mapsto \begin{cases} 1, & si\ n = 0; \\ c\ |2^i \leq n \wedge 2^{i+1} > n, & \text{en otro caso.} \end{cases}$$

arreglos.h: linea 17
arreglos.cpp: linea 134-141

$$dec2bin : \mathbb{N} \to \mathbb{Z}^*$$
$$(n) \mapsto x, \quad \forall_{i=1}^{u} x_i,\ con\ u = tama(n) \wedge x_i = n\,\%2 \wedge n = n/2;$$

Codificación C++:

```
int tama(int n){
    if(n==0) return 1;
    int c=0;
    while(potencia(2,c)<=n){
        c++;
    }
    return c;
}

int* dec2bin(int n){
    int u = tama(n)+1;
    int* x= new int[u];
    x[0] = u;
    for(int i=1;i<u;i++){
        x[i] = n%2;
        n = n/2;
    }
    return x;
}
```

Problema 33. *Maximo comun divisor del arreglo*
Hacer un algoritmo que calcule el máximo común divisor para un arreglo de

enteros positivos.
arreglos.h: linea 19
arreglos.cpp: linea 154-160

Modelo Matemático:

$$mcd : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$$
$$(d, b) \mapsto \begin{cases} d, & si\ b = 0; \\ mcd(b, d\,\%b), & \text{en otro caso;} \end{cases}$$

numericos.h: linea 7
numericos.cpp: linea 27-33

$$max\_divi : \mathbb{Z}^* \times \mathbb{N} \to \mathbb{Z}$$
$$(x, n) \mapsto \begin{cases} x_1, & si\ n = 1; \\ mcd(max\_divi(x, n-1), x_n), & \text{en otro caso.} \end{cases}$$

Codificación C++:

```
int max_div(int* x, int n){
    if(n==1){
        return x[0];
    }else{
        return mcd(max_div(x,n-1),x[n-1]);
    }
}
```

Problema 34. *Minimo comun multiplo del arreglo*
Hacer un algoritmo que calcule el mínimo común multiplo para un arreglo de enteros positivos.
arreglos.h: linea 21
arreglos.cpp: linea 166-172

Modelo Matemático:

$$mcm : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$$
$$(a, b) \mapsto \frac{a * b}{mcd(a, b)}$$

arreglos.h: linea 20
arreglos.cpp: linea 162-164

$$min\_mul : \mathbb{Z}^* \times \mathbb{N} \to \mathbb{Z}$$
$$(x, n) \mapsto \begin{cases} x_1, & si\ n = 1; \\ mcm(min\_mul(x, n-1), x_n), & \text{en otro caso.} \end{cases}$$

Codificación C++:

```
int mcm(int a, int b){
    return a*b/(mcd(a,b));
}

int min_mul(int* x, int n){
    if(n==1){
        return x[0];
    }else{
        return mcm(min_mul(x,n-1),x[n-1]);
    }
}
```

6. Arreglos como conjuntos

Un arreglo de elementos de tipo T se puede utilizar para representar un conjunto finito de elementos del tipo T. Esta representación es como sigue: El conjunto A = x0, x1, x2, ... , xn-1 se representa como el arreglo [x0, x1, x2, ... , xn-1]

$$crear\_arreglo\_int : \mathbb{N} \to \mathbb{Z}^*$$
$$(n) \mapsto x, \qquad \text{donde x} \in \mathbb{Z}\ n\ \not\subseteq \mathbb{Z}$$

arr_conjuntos.h: linea 4
arr_conjuntos.cpp: linea 3-5

Codificación C++: \

```
int* crear_arreglo_int(int n){
    return new int[n];
};
```

Problema 35. *Union*
Calcula en un arreglo la unión de los conjuntos y la imprime.
arr_conjuntos.h: linea 5
arr_conjuntos.cpp: linea 7-18

Modelo Matemático:

$$unio : \mathbb{Z}^* \times \mathbb{N} \times \mathbb{Z}^* \times \mathbb{N} \to \mathbb{Z}^*$$
$$(x, m, y, n) \mapsto c, \qquad | \ \forall_{i=1}^{m+n} c_i = x_i \vee y_i;$$

Codificación C++:

```
int* unio(int* x,int m, int* y,int n){
    int s = 1+n+m;
    int* c = crear_arreglo_int(s);
    c[0] = s;
    for(int i=1;i<=m;i++){
        c[i] = x[i-1];
    }
    for(int i=m+1;i<s;i++){
        c[i] = y[i-m-1];
    }
    return c;
}
```

Problema 36. *Interseccion*
Calcula en un arreglo la intersección de los conjuntos y la imprime.
arr_conjuntos.h: linea 6
arr_conjuntos.cpp: linea 20-32

Modelo Matemático:

$$interseccion : \mathbb{Z}^* \times \mathbb{N} \times \mathbb{Z}^* \times \mathbb{N} \to \mathbb{Z}^*$$
$$(x, m, y, n) \mapsto c, \qquad | \ \forall c_i = x_i \wedge y_i;$$

Codificación C++:

```
int* interseccion(int* x,int m, int* y,int n){
    int h =1;
    int* c = crear_arreglo_int(h);
    for(int i=0;i<m;i++){
        for(int l=0;l<n;l++){
            if(y[l]==x[i]){
                c[h++] = x[i];
            }
        }
    }
    c[0] = h;
```

```
    return c;
}
```

Problema 37. *Diferencia*
Calcula en un arreglo la diferencia del primero con el segundo y la imprime.
arr_conjuntos.h: linea 7
arr_conjuntos.cpp: linea 34-55

Modelo Matemático:

$$diferencia : \mathbb{Z}^* \times \mathbb{N} \times \mathbb{Z}^* \times \mathbb{N} \to \mathbb{Z}^*$$
$$(x, m, y, n) \mapsto x, \qquad \forall_{i=1}^{m} x_i |\ x_i \notin y$$

Codificación C++:

```
int* diferencia(int* x,int m, int* y,int n){
    int* u = crear_arreglo_int(m);
    for(int i=0;i<m;i++){
        u[i] = x[i];
    }
    for(int i=0;i<m;i++){
        for(int l=0;l<n;l++){
            if(x[i]==y[l]){
                u[i] = 0;
            }
        }
    }
    int h =1;
    int* c = new int[h];
    for(int i=0;i<m;i++){
        if(u[i]!=0){
            c[h++] = u[i];
        }
    }
    c[0] = h;
    return c;
}
```

Problema 38. *Diferencia simetrica*
Calcula en un arreglo la diferencia simétrica de los conjuntos y la imprime.
arr_conjuntos.h: linea 8
arr_conjuntos.cpp: linea 57-73

Modelo Matemático:

$$diferencia\_simetrica : \mathbb{Z}^* \times \mathbb{N} \times \mathbb{Z}^* \times \mathbb{N} \to \mathbb{Z}^*$$
$$(x, m, y, n) \mapsto c, \quad \forall c_i |\ c_i \in x \vee c_i \in y \wedge c_i \notin interseccion(x, m, y, n);$$

Codificación C++:

```
int* diferencia_simetrica(int* x,int m, int* y,int n){
    int* p = diferencia(x,m,y,n);
    int* o = diferencia(y,n,x,m);
    int a = p[0];
    int b = o[0];
    int s = a+b-1;
    int* d = crear_arreglo_int(s);
    d[0] = s;
    for(int i=1;i<a;i++){
        d[i] = p[i];
    }
    int t=1;
    for(int i=a;i<s;i++){
            d[i] = o[t++];
    }
    return d;
}
```

Problema 39. *Pertenece*
Leer un entero y determina si el elemento pertenece o no a cada uno de los conjunto y la imprime.
arr_conjuntos.h: linea 9
arr_conjuntos.cpp: linea 75-89

Modelo Matemático:

$$pertenece : \mathbb{Z}^* \times \mathbb{N} \times \mathbb{Z}^* \times \mathbb{N} \times \mathbb{Z} \to \mathbb{Z}^*$$
$$(x, m, y, n, p) \mapsto \begin{cases} \text{V y V}, & si\ p \in x \wedge p \in y; \\ \text{V y F}, & si\ p \in x \wedge p \notin y; \\ \text{F y V}, & si\ p \notin x \wedge p \in y; \\ \text{F y F}, & \text{en otro caso;} \end{cases}$$

Codificación C++:

```
bool* pertenece(int* x,int m, int* y,int n, int p){
    bool* o = new bool[2];
    o[0] = false;
    o[1] = false;
    for(int i=0;i<m;i++){
        if(x[i]== p){
            o[0] = true;
        }
    }for(int l=0;l<n;l++){
        if(y[l]==p){
            o[1] = true;
        }
    }
    return o;
}
```

Problema 40. *Contenido*
Determina sí el primer conjunto esta contenido en el segundo y la imprime.
arr_conjuntos.h: linea 10
arr_conjuntos.cpp: linea 91-108

Modelo Matemático:

$$contenido : \mathbb{Z}^* \times \mathbb{N} \times \mathbb{Z}^* \times \mathbb{N} \to \mathbb{B}$$

$$(x, m, y, n) \mapsto \begin{cases} V, & si\ \forall_{i=1}^{m} x_i,\ x_i \in y; \\ F, & \text{en otro caso;} \end{cases}$$

Codificación C++:

```
bool contenido(int* x,int m, int* y,int n){
    bool* c = new bool[m];
    for(int f=0;f<m;f++){
        c[f] = false;
    }for(int i=0;i<m;i++){
        for(int l=0;l<n;l++){
            if(x[i]==y[l]){
                c[i] = true;
            }
        }
    }int p=0;
    for(int r=0;r<m;r++){
        if(c[r]==true){
```

```
            p++;
        }
    }
    return p==m;
}
```

7. Arreglos como polinomios

Un polinomio de grado n, como P(x) = anxn + an-1xn-1 + ... + a1x1 + a0x0 se puede representar mediante un arreglo de reales de la siguiente manera: [a0, a1, ... , an-1, an].

$$crear\_arreglo\_double : \mathbb{N} \to \mathbb{R}^*$$
$$(n) \mapsto x, \qquad \text{donde x} \in \mathbb{R}\ n\ \not\subseteq \mathbb{R}$$

arr_conjuntos.h: linea 4
arr_conjuntos.cpp: linea 6-8

Codificación C++: \

```
double* crear_arreglo_double(int n){
    return new double[n];
}
```

$$leer\_polinomio : \mathbb{N} \to \mathbb{R}^*$$
$$(n) \mapsto a, \qquad \forall_{i=0}^{n} a_i$$

arr_conjuntos.h: linea 5
arr_conjuntos.cpp: linea 10-16

Codificación C++:

```
double* leer_polinomio(int n){
    double* y = crear_arreglo_double(n+1);
    for(int i=n;i>=0;i--){
        cout << "Ingrese el coeficiente de x^" << i << " : " <<  endl;
        cin >> y[i];
    }return y;
}
```

Problema 41. *Evaluar dos polinomios*
Lee un real e imprime la evaluación de los dos polinomios en dicho dato.
arr_polinomios.h: linea 6
arr_polinomios.cpp: linea 18-31

Modelo Matemático:

$$evaluar : \mathbb{R}^* \times \mathbb{N} \times \mathbb{R}^* \times \mathbb{N} \times \mathbb{N} \to \mathbb{R}^*$$
$$(a, n, b, m, x) \mapsto \begin{cases} p_1, & \sum_{i=0}^{n}(a_i * potencia(x, i)) \\ p_2, & \sum_{i=0}^{m}(b_i * potencia(x, i)) \end{cases}$$

Codificación C++:

```
double* evaluar(double* a,int n, double* b, int m, int x){
    double* p = crear_arreglo_double(2);
    int ac = 0;
    for(int i=n;i>=0;i--){
        ac = ac + a[i] * potencia(x,i);
    }
    int bc = 0;
    for(int i=m;i>=0;i--){
        bc = bc + b[i] * potencia(x,i);
    }
    p[0] = ac;
    p[1] = bc;
    return p;
}
```

Problema 42. *Suma de dos polinomios*
Calcula el polinomio suma y lo imprime.
arr_polinomios.h: linea 8
arr_polinomios.cpp: linea 38-54

Modelo Matemático:

$$max\_int : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$$
$$(a, b) \mapsto \begin{cases} a, & si\ a > b; \\ b, & \text{en otro caso;} \end{cases}$$

arr_polinomios.h: linea 7

arr_polinomios.cpp: linea 33-36

$suma\_pol : \mathbb{R}^* \times \mathbb{N} \times \mathbb{R}^* \times \mathbb{N} \to \mathbb{R}^*$

$$(a, n, b, m) \mapsto y, \ con\ j = max\_int(a, b)| \begin{cases} y_j = a_n, & si\ n > m, n = n - 1; \\ y_j = b_m, & si\ n < m, m = m - 1; \\ \sum_{i=m}^{j}(a_i + b_i), & si\ n = m; \end{cases}$$

Codificación C++:

```
int max_int(int a, int b){
    if(a>b) return a;
    else return b;
}

double* suma_pol(double* a,int n, double* b, int m){
    int j = max_int(m,n);
    double* y = crear_arreglo_double(j);
    for(int i=j;i>=0;i--){
        if(n==m){
            y[i] = a[i] + b[i];
        }if(n>m){
            y[i] = a[n];
            n--;
        }if(n<m)
            {
            y[i] = b[m];
            m--;
        }
    }
    return y;
}
```

Problema 43. *Resta de dos polinomios*
Calcula el polinomio resta y lo imprime.
arr_polinomios.h: linea 9
arr_polinomios.cpp: linea 56-72

Modelo Matemático:

$resta\_pol : \mathbb{R}^* \times \mathbb{N} \times \mathbb{R}^* \times \mathbb{N} \to \mathbb{R}^*$

$$(a, n, b, m) \mapsto y, \ con\ j = max\_int(a, b)| \begin{cases} y_j = a_n, & si\ n > m, n = n - 1; \\ y_j = b_m, & si\ n < m, m = m - 1; \\ \sum_{i=m}^{j}(a_i - b_i), & si\ n = m; \end{cases}$$

Codificación C++:

```
double* resta_pol(double* a,int n, double* b, int m){
    int j = max_int(m,n);
    double* y = crear_arreglo_double(j);
    for(int i=j;i>=0;i--){
        if(n==m){
            y[i] = a[i] - b[i];
        }if(n>m){
            y[i] = a[n];
            n--;
        }if(n<m)
            {
            y[i] = b[m];
            m--;
        }
    }
    return y;
}
```

Problema 44. *Multiplicación de dos polinomios*
Calcula el polinomio multiplicación y lo imprime.
arr_polinomios.h: linea 10
arr_polinomios.cpp: linea 74-84

Modelo Matemático:

$$mult\_pol : \mathbb{R}^* \times \mathbb{N} \times \mathbb{R}^* \times \mathbb{N} \to \mathbb{R}^*$$
$$(a, n, b, m) \mapsto c \mid \forall_{i=0}^{i<=n} \mid \forall_{j=0}^{j<=m} \; c_{i+j} = c_{i+j} + a_{[i]} * b_{[j]}$$

Codificación C++:

```
double* mult_pol(double* a,int n, double* b, int m){
    int t=m+n;
    double* c= crear_arreglo_double(t);
    for(int i=0;i<t;i++){
            c[i]=0;
    }for(int i=0; i<=n; i++){
        for(int j=0; j<=m; j++){
                c[i+j]= c[i+j]+ (a[i]*b[j]);
        }
    }return c;
}
```

Problema 45. *Division de dos polinomios*
Calcula el polinomio división del primer polinomio por el segundo y lo imprime.
arr_polinomios.h: linea 11
arr_polinomios.cpp: linea 86-98

Modelo Matemático:

$$div\_pol : \mathbb{R}^* \times \mathbb{N} \times \mathbb{R}^* \times \mathbb{N} \times \mathbb{R}^* \times \mathbb{N} \to \mathbb{R}^*$$
$$(a, n, b, m, s, t) \mapsto s |\ \forall_{i=1}^{n-m+1} s_i = a_n/b_m,\ y\ a_n = a_n - b_m * s_i;$$

Codificación C++:

```
double* div_pol(double* a,int n, double* b, int m, double* s, double t){
    double* d = crear_arreglo_double(n+1);
    for(int i=0;i<=n;i++){
        d[i] = a[i];
    }
    for(int i=n;i>=m;i--){
        s[i-m] = (d[i]/b[m]);
        for(int j=0;j<=m;j++){
            d[i-j]-= b[m-j]*s[i-m];
        }
    }
    return s;
}
```

Problema 46. *Residuo de la division*
Calcula el polinomio residuo de la división del primero por el segundo y lo imprime.
arr_polinomios.h: linea 12
arr_polinomios.cpp: linea 100-112

Modelo Matemático:

$$res\_pol : \mathbb{R}^* \times \mathbb{N} \times \mathbb{R}^* \times \mathbb{N} \times \mathbb{R}^* \times \mathbb{N} \to \mathbb{R}^*$$
$$(a, n, b, m, s, t) \mapsto a | \forall_{i=1}^{n-m+1} a_n = a_n - b_m * s_i,\ y\ s_i = a_n/b_m, ;$$

Codificación C++:

```
double* res_pol(double* a,int n, double* b, int m, double* s, double t){
    double* d = crear_arreglo_double(n+1);
    for(int i=0;i<=n;i++){
        d[i] = a[i];
```

```
    }
    for(int i=n;i>=m;i--){
        s[i-m] = (d[i]/b[m]);
        for(int j=0;j<=m;j++){
            d[i-j]-= b[m-j]*s[i-m];
        }
    }
    return d;
}
```

8. Matrices

Modelo Matemático:

$$crear\_matriz\_double : \mathbb{N} \times \mathbb{N} \to \mathbb{R}^{**}$$
$$(n, m) \mapsto x, \qquad \text{donde x} \in \mathbb{R}^{n*m} \; n \;\not\subseteq \mathbb{R}^{**}$$

matrices.h: linea 4
matrices.cpp: linea 4-10

Codificación C++:

```
double** crear_matriz_double(int n, int m){
    double** X = new double*[n];
    for(int i = 0; i < n; i++){
        X[i] = new double[m];
    };
    return X;
};
```

Problema 47. *Suma de dos matrices*
Desarrollar un algoritmo que permita sumar dos matrices de números reales (enteros).
matrices.h: linea 5
matrices.cpp: linea 12-20

Modelo Matemático:

$$suma\_matriz : \mathbb{R}^{**} \times \mathbb{R}^{**} \times \mathbb{N} \times \mathbb{N} \to \mathbb{R}^{**}$$
$$(x, y, n, m) \mapsto c, \qquad \forall_{i=0}^{n} \forall_{j=0}^{n} \; c_{i,j} = x_{i,j} + y_{i,j}.$$

Codificación C++:

```
double** suma_matriz(double** x, double** y, int n, int m){
    double** c = crear_matriz_double(n, m);
    for(int i = 0; i < n; i++){
        for(int j = 0; j < m; j++){
            c[i][j] = x[i][j] + y[i][j];
        };
    };
    return c;
};
```

Problema 48. *Multiplicacion de dos matrices* Desarrollar un algoritmo que permita multiplicar dos matrices de números reales (enteros).
matrices.h: linea 6
matrices.cpp: linea 22-30

Modelo Matemático:

$$multi\_matrices : \mathbb{R}^{**} \times \mathbb{R}^{**} \times \mathbb{N} \times \mathbb{N} \to \mathbb{R}^{**}$$
$$(x, y, n, m) \mapsto c, \qquad \forall_{i=0}^{n} \forall_{j=0}^{m} c_{i,j} = x_{i,j} * y_{i,j},$$

Codificación C++:

```
double** multi_matrices(double** x,double** y,int n, int m){
    double** z = crear_matriz_double(n,m);
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
                z[i][j] = x[i][j] * y[i][j];
        }
    }
    return z;
}
```

Problema 49. *Suma elementos de una columna*
Desarrollar un programa que sume los elementos de una columna dada de una matriz.
matrices.h: linea 7
matrices.cpp: linea 32-38

Modelo Matemático:

$$suma\_matriz\_columna : \mathbb{R}^{**} \times \mathbb{N} \times \mathbb{N} \times \mathbb{N} \to \mathbb{R}$$
$$(x, n, m, p) \mapsto \sum_{i=1}^{n} x_{i,p}$$

Codificación C++:

```
double suma_matriz_columna(double** x, int n, int m, int p){
    int ac = 0;
    for(int i = 0; i < n; i++){
        ac  = ac + x[i][p];
    };
    return ac;
};
```

Problema 50. *Suma elementos de una fila*
Desarrollar un programa que sume los elementos de una fila dada de una matriz.
matrices.h: linea 8
matrices.cpp: linea 40-46

Modelo Matemático:

$$suma\_matriz\_fila : \mathbb{R}^{**} \times \mathbb{N} \times \mathbb{N} \times \mathbb{N} \to \mathbb{R}$$
$$(x, n, m, p) \mapsto \sum_{j=1}^{m} x_{p,j}$$

Codificación C++:

```
double suma_matriz_columna(double** x, int n, int m, int p){
    int ac = 0;
    for(int i = 0; i < n; i++){
        ac  = ac + x[i][p];
    };
    return ac;
};
```

Problema 51. *Matriz magica* Desarrollar un algoritmo que determine si una matriz es mágica. Se dice que una matriz cuadrada es mágica si la suma de cada una de sus filas, de cada una de sus columnas y de cada diagonal es igual.
matrices.h: linea 12
matrices.cpp: linea 78-88

Modelo Matemático:

$$filas\_suma : \mathbb{R}^{**} \times \mathbb{N} \to \mathbb{B}$$
$$(x, n) \mapsto \begin{cases} V, & si \ \sum_{i=1}^{n} x_{i,p}, \text{es igual } \forall_{i=1}^{n} p_i \\ F, & \text{en otro caso;} \end{cases}$$

matrices.h: linea 9
matrices.cpp: linea 48-57

$$columnas\_suma : \mathbb{R}^{**} \times \mathbb{N} \to \mathbb{B}$$
$$(x,n) \mapsto \begin{cases} V, & si\ \sum_{i=1}^{n} x_{p,i}, \text{es igual } \forall_{i=1}^{n} p_i \\ F, & \text{en otro caso;} \end{cases}$$

matrices.h: linea 10
matrices.cpp: linea 59-68

$$suma\_matriz\_diagonal : \mathbb{R}^{**} \times \mathbb{N} \to \mathbb{R}$$
$$(x,n) \mapsto \sum_{i=1}^{n} x_{i,i}$$

matrices.h: linea 11
matrices.cpp: linea 70-76

$$matriz\_magica : \mathbb{R}^{**} \times \mathbb{N} \to \mathbb{B}$$
$$(x,n) \mapsto \begin{cases} V, & si\ (filas\_suma(x,n) = V \wedge columnas\_suma(x,n)) \wedge \\ & (suma\_matriz\_fila(x,n,n,0) = suma\_matriz\_columna(x,n,n,0) \\ & = suma\_matriz\_diagonal(x,n)) \\ F, & \text{en otro caso;} \end{cases}$$

Codificación C++:

```
bool filas_suma(double** x,int n){
    double p = suma_matriz_fila(x,n,n,0);
    bool c = true;
    for(int i=1;i<n;i++){
        if(suma_matriz_fila(x,n,n,i)!=p){
            c = false;
        }
    }
    return c;
}

bool columnas_suma(double** x,int n){
```

```
    double p = suma_matriz_columna(x,n,n,0);
    bool c = true;
    for(int i=1;i<n;i++){
        if(suma_matriz_columna(x,n,n,i)!=p){
            c = false;
        }
    }
    return c;
}

double suma_matriz_diagonal(double** x, int n){
    int ac=0;
    for(int i=0;i<n;i++){
        ac = ac + x[i][i];
    }
    return ac;
}

bool matriz_magica(double** x,int n){
    if(filas_suma(x,n)==1 && columnas_suma(x,n)==1){
        double a = suma_matriz_fila(x,n,n,0);
        double b = suma_matriz_columna(x,n,n,0);
        double c = suma_matriz_diagonal(x,n);
        if(a==b && b==c){
            return true;
        }
    }
    return false;
}
```

Problema 52. *Reemplazar respecto a un numero*
Desarrollar un algoritmo que dado un entero, reemplace en una matriz todos los números mayores a un número dado por un uno y todos los menores o iguales por un cero.
matrices.h: linea 13
matrices.cpp: linea 90-101

Modelo Matemático:

$$may\_men\_num : \mathbb{R}^{**} \times \mathbb{N} \times \mathbb{N} \times \mathbb{N} \to \mathbb{R}$$

$$(x, n, m, p) \mapsto \begin{cases} 1, & \forall_{i=0}^{n} \forall_{j=0}^{m}\ x_{i,j} > p; \\ 0, & \text{en otro caso;} \end{cases}$$

Codificación C++:

```
double** may_men_num(double** x, int n, int m, int p){
    for(int i = 0; i < n; i++){
        for(int j = 0; j < m; j++){
            if(x[i][j]>p){
                x[i][j] = 1;
            }else{
                x[i][j] = 0;
            }
        };
    };
    return x;
}
```

Problema 53.
matrices.h: linea 6
matrices.cpp: linea 25-31

Modelo Matemático:

$$: \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}$$
$$(A, n) \mapsto$$

Codificación C++:

Problema 54.
matrices.h: linea 6
matrices.cpp: linea 25-31

Modelo Matemático:

$$: \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}$$
$$(A, n) \mapsto$$

Codificación C++:

Problema 55.
matrices.h: linea 6
matrices.cpp: linea 25-31

Modelo Matemático:

$$: \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}$$
$$(A, n) \mapsto$$

Codificación C++:

Problema 56.
matrices.h: linea 6
matrices.cpp: linea 25-31

Modelo Matemático:

$$: \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}$$
$$(A, n) \mapsto$$

Codificación C++:

9. Relaciones binarias como matrices

Una matriz se puede usar para representar una relación entre dos conjuntos A y B. Esta representación es como sigue: Si A = x0, x1, x2, ... , xn-1 y B = y0, y1, y2, ... , ym-1 una relación R de A en B se representa mediante una matriz de unos y ceros, donde A[i][j] = 1 si el elemento xi se relaciona con el elemento yj.

$$crear\_matriz\_int : \mathbb{N} \times \mathbb{N} \to \mathbb{Z}^{**}$$
$$(n, m) \mapsto x, \qquad \text{donde x} \in \mathbb{Z}^{n*m} \; n \;\not\subseteq \mathbb{Z}^{**}$$

matrices.h: linea 4
matrices.cpp: linea 3-9

Codificación C++: \

```
int** crear_matriz_int(int n, int m){
    int** X = new int*[n];
    for(int i = 0; i < n; i++){
        X[i] = new int[m];
    };
    return X;
};
```

Problema 57. *Union*
Calcula e imprime la relación unión.
rela_bina.h: linea 5
rela_bina.cpp: linea 11-19

Modelo Matemático:

$$union\_rel : \mathbb{N}^* \times \mathbb{N}^* \times \mathbb{N} \times \mathbb{N} \to \mathbb{N}^*$$
$$(r, s, n, m) \mapsto T | \ \forall_{i=0}^{n-1} \forall_{j=0}^{m-1} \ T_{i,j} = r_{i,j} \vee s_{i,j}$$

Codificación C++:

```
int** union_rel(int** r, int** s, int n, int m){
    int** T = crear_matriz_int(n,m);
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            T[i][j] = r[i][j] || s[i][j];
        }
    }
    return T;
}
```

Problema 58. *Interseccion*
Calcula e imprime la relación intersección.
rela_bina.h: linea 6
rela_bina.cpp: linea 21-29

Modelo Matemático:

$$interseccion\_rel : \mathbb{N}^* \times \mathbb{N}^* \times \mathbb{N} \times \mathbb{N} \to \mathbb{N}^*$$
$$(r, s, n, m) \mapsto T | \ \forall_{i=0}^{n-1} \forall_{j=0}^{m-1} \ T_{i,j} = r_{i,j} \wedge s_{i,j}$$

Codificación C++:

```
int** interseccion_rel(int** r, int** s, int n, int m){
    int** T = crear_matriz_int(n,m);
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            T[i][j] = r[i][j] && s[i][j];
        }
    }
    return T;
}
```

Problema 59.

matrices.h: linea 6
matrices.cpp: linea 25-31

Modelo Matemático:

$$: \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}$$
$$(A, n) \mapsto$$

Codificación C++:

Problema 60.

matrices.h: linea 6
matrices.cpp: linea 25-31

Modelo Matemático:

$$: \mathbb{Z}^* \times \mathbb{N} \to \mathbb{R}$$
$$(A, n) \mapsto$$

Codificación C++: