

Algoritmia

Por: Elliott Gibranny Mis Ramírez

Maestro: Jorge Alberto Ríos Martínez

Buenas, en este curso de algoritmia llevamos muchos temas fundamentales para poder comprender los algoritmos, estos sus funciones más comunes y como llevar su lógica para poder resolver problemas con dichos algoritmos, estos pueden ser tanto matemáticos como lógicos o incluso de ordenamiento. Vimos en total 4 temas cuales son algoritmos secuenciales, selectivos, iterativos y con arreglos, cuales cada uno se mostraban ejercicios pertinentes para su resolución en cada contexto. A continuación, se presentará los problemas que resuelven cada uno de los algoritmos, estos incluyen resoluciones por parte de la clase y otros 5 que serán separados de este archivo de estos 25 problemas, dichos 5 ejercicios son hechos personalmente, es decir, no se hizo en clase. Continuemos con los algoritmos:

Indice

Algoritmos secuenciales	2
Algoritmos Selectivos.....	9
Algoritmos con estructuras iterativas.....	19
Ejercicios con arreglos: Vectores.	32
Algoritmos con arreglos: Matrices.....	40

Algoritmos secuenciales

1. Escribe un algoritmo que permita obtener el área de un triángulo a partir de la longitud de su base y de su altura.

- Entradas: Altura y Base.
- Salidas: AreadelTriangulo.
- Foto del código:

Algoritmo Arealtriangulo

//Entradas: Altura y Base.

//Salida: AreadelTriangulo

//Caso de prueba: altura <- 5 Base <- 4 da como resultado AreadelTriangulo<- 10

Definir Altura, Base, AreadelTriangulo **Como Real**;

altura **<-** 0.0;

base **<-** 0.0;

AreadelTriangulo **<-** 0.0;

Escribir "Mencioname la altura del triangulo";

leer Altura;

Escribir "Mencioname la base del triangulo";

leer base;

AreadelTriangulo **<-** (altura * base)/2;

escribir "el area del triangulo es: ", AreadelTriangulo;

- Foto de ejecución:

*** Ejecución Iniciada. ***

Mencioname la altura del triangulo

> 10

Mencioname la base del triangulo

> 15

el area del triangulo es: 75

*** Ejecución Finalizada. ***

- Este algoritmo fue añadido porque me pareció increíble empezar la asignatura con esto, es el primer ejercicio del primer bloque de ejercicios de la asignatura

2. Se trata de un algoritmo que tiene de entrada el tiempo y te devolverá cual sería la distancia recorrida con ese tiempo si vas a la velocidad del sonido

- entradas: tiempo
- salidas: distancia, distanciarRestante
- Foto de código:

Algoritmo velocidadsonido

//entradas: tiempo.

//salidas: distancia, distanciarRestante.

//proceso de prueba: 5

//= la distancia recorrida a la velocidad de la luz en metros es: 1720

//= la distancia restante en metros para llegar a Tokio desde merida es: 11841283

definir tiempo, distancia, distanciaRestante **como** real;

tiempo \leftarrow 0.0;

distancia \leftarrow 0.0;

distanciaRestante \leftarrow 0.0;

escribir "dime el tiempo en segundos";

leer tiempo;

distancia \leftarrow tiempo * 344;

distanciaRestante \leftarrow 11843000 - distancia;

escribir "la distancia recorrida a la velocidad de la luz en metros es: ", distancia;

escribir "la distancia restante en metros para llegar a Tokio desde merida es: ", distanciaRestante;

FinAlgoritmo

- Foto de ejecución:

*** Ejecución Iniciada. ***

dime el tiempo en segundos

> 100

la distancia recorrida a la velocidad de la luz en metros es: 34400

la distancia restante en metros para llegar a Tokio desde merida es: 11808600

*** Ejecución Finalizada. ***

- Este algoritmo fue añadido meramente porque me fue curioso saber ese dato, un poco gracioso

3. Este es un problema un poco más largo, tiene como instrucción: a CONAGUA requiere determinar el pago trimestral que debe realizar una persona por los metros cúbicos que consume de agua. Para realizar el cálculo se toman los consumos de los tres últimos meses y se saca el promedio. Realice el algoritmo que permita determinar ese pago.

- entradas: metrosCubicos
- salida: promedioGasto
- Foto del código:

Algoritmo CONAGUA

```
//entradas: metrosCubicos
//salida: promedioGasto
//caso de prueba: 30, 30, 30, 30

definir metrcub1, metrcub2, metrcub3, promedioGasto como real;

metrcub1 ← 0.0;
metrcub2 ← 0.0;
metrcub3 ← 0.0;
promedioGasto ← 0.0;

escribir "¿cuántos metros cubicos se consumieron de agua en el primer, segundo y tercer mes consecutivos?";
leer metrcub1, metrcub2, metrcub3;

promedioGasto ← (metrcub1 + metrcub2 + metrcub3)/3;

escribir "el consumo total de agua en promedio fue de: ", promedioGasto " metros cubicos";
```

FinAlgoritmo

- Foto de ejecución:

```
*** Ejecución Iniciada. ***
¿cuántos metros cubicos se consumieron de agua en el primer, se
gundo y tercer mes consecutivos?
> 100
> 100
> 100
el consumo total de agua en promedio fue de: 100 metros cubicos

*** Ejecución Finalizada. ***
```
- Este ejercicio lo añadí porque me parece muy limpio el código

4. algoritmo para determinar cuánto dinero ahorra una persona en un año si considera que cada semana ahorra 15% de su sueldo (excepción: cuatro semanas por mes y no cambia el sueldo).

- entradas: sueldo
- salidas: ahorroSemanal
- Foto de código:

Algoritmo ahorros

```
//entradas: sueldo  
//salidas: ahorroSemanal  
//proceso de prueba: 200 es su salario, semanal ahorrará 30 pesos (15%)  
//mensual ahorrará 120 considerando 4 semanas, en un año ahorrará 1440 pesos
```

Definir sueldo, ahorroSemanal, ahorroAnual **Como Real**;

```
sueldo ← 0.0;  
ahorroSemanal ← 0.0;  
ahorroAnual ← 0.0;
```

```
escribir "¿Cuál es tu sueldo?";  
leer sueldo;
```

```
ahorroSemanal ← sueldo * .15;
```

```
escribir "Semanalmente se ahorrará: ", ahorroSemanal " pesos";
```

```
ahorroSemanal ← ahorroSemanal * 4;
```

```
Escribir "Mensualmente se ahorrará: ", ahorroSemanal " pesos";
```

```
ahorroAnual ← ahorroSemanal * 12;
```

```
Escribir "En un año se ahorra: ", ahorroAnual " pesos";
```

FinAlgoritmo

- Foto de ejecución:
*** Ejecución Iniciada. ***
¿Cuál es tu sueldo?
> 300
Semanalmente se ahorrará: 45 pesos
Mensualmente se ahorrará: 180 pesos
En un año se ahorra: 2160 pesos
*** Ejecución Finalizada. ***

5. Se trata de un Algoritmo para determinar el promedio que obtendrá un alumno considerando que realiza tres exámenes, de los cuales el primero y el segundo tienen una ponderación de 25%, mientras que el tercero es de 50%.

- entradas: exam1, exam2, exam3
- salidas: promedio
- Foto de código:

```
Algoritmo Calificacionsexamenes
//entradas: exam1, exam2, exam3
//salidas: promedio
//proceso de prueba: 70, 70, 100 promedio <- 67.5+25 <- 92.5

definir exam1, exam2, exam3, promedio como real;

exam1 <- 0.0;
exam2 <- 0.0;
exam3 <- 0.0;
promedio <- 0.0;

escribir "sobre 100, ¿cuánto sacó el primer y segundo examen?";
leer exam1, exam2;

escribir "sobre 100, ¿cuánto saco en el tercer examen?";
leer exam3;

promedio <- (((exam1 + exam2)/2)*.25) + ((exam3 *.50))+25;

Escribir "El promedio total del alumno es de: ", promedio;
FinAlgoritmo
```

- Foto de ejecución:
- ```
sobre 100, ¿cuánto sacó el primer y segundo examen?
> 70
> 70
sobre 100, ¿cuánto saco en el tercer examen?
> 100
El promedio total del alumno es de: 92.5
*** Ejecución Finalizada. ***
```

- Este pseudocódigo fue añadido porque se me hizo un poco complejo entender las instrucciones, me parecían un poco ambiguas, sin embargo, lo realice como pensaba que era.

6. Algoritmo que calcula la equivalencia en meses, semanas, días y horas de la edad de una persona proporcionada en años.

- entradas: edad
- salidas; meses, semanas, días, horas como equivalencias de su edad
- foto de pseudocódigo:

**Algoritmo** equivalenciasEdad

*//entradas: edad*

*//salidas; meses, semanas, días, horas como equivalencias de su edad*

*//proceso de prueba: 18, equivalen a 216 meses, 864 semanas, 6048 días, 145,152 horas.*

**definir** edad, meses, semanas, días, horas **como entero**;

edad  $\leftarrow$  0.0;

meses  $\leftarrow$  0.0;

semanas  $\leftarrow$  0.0;

días  $\leftarrow$  0.0;

horas  $\leftarrow$  0.0;

**escribir** "Mencioname tu edad: ";

**leer** edad;

meses  $\leftarrow$  edad \* 12;

semanas  $\leftarrow$  edad \* (12\*4);

días  $\leftarrow$  edad \* (12\*4\*7);

horas  $\leftarrow$  edad \* (12\*4\*7\*24);

**escribir** "Tu edad en meses equivale a: ", meses " meses";

**escribir** "en días semanas equivale a: ", semanas " semanas";

**escribir** "en días equivale a: ", días " días";

**escribir** "en horas, equivale a: ", horas " horas";

**FinAlgoritmo**

- Foto de ejecución:

\*\*\* Ejecución Iniciada. \*\*\*

Mencioname tu edad:

> 18

Tu edad en meses equivale a: 216 meses

en días semanas equivale a: 864 semanas

en días equivale a: 6048 días

en horas, equivale a: 145152 horas

\*\*\* Ejecución Finalizada. \*\*\*

- Añadí este pseudocódigo porque me pareció divertido de realizar, sobre todo con las ecuaciones para calcular el tiempo

## 7. Algoritmo que pide los valores X, Y, Z de dos puntos tridimensionales y de salida imprime las coordenadas del punto medio entre esos puntos

- Entradas: x1, y1, z1, x2, y2, z2. como coordenadas del primer y segundo punto
- salidas: puntoMedio, como el punto medio de los dos puntos tridimensionales
- Foto de pseudocódigo:

### Algoritmo puntoMedioEnEjes

*//entradas x1, y1, z1, x2, y2, z2. como coordenadas del primer y segundo punto*  
*//salidas: puntoMedio, como el punto medio de los dos puntos tridimensionales*  
*//caso de prueba: (4,5,6), (7,8,2) el punto medio es: (5.5, 6.5, 4)*

**definir** x1,y1,z1,x2,y2,z2,puntoMedio1, puntoMedio2, puntoMedio3 **como** real;

x1 ← 0.0;  
 y1 ← 0.0;  
 z1 ← 0.0;  
 x2 ← 0.0;  
 y2 ← 0.0;  
 z2 ← 0.0;  
 puntoMedio1 ← 0.0;  
 puntoMedio2 ← 0.0;  
 puntoMedio3 ← 0.0;

**escribir** "mencioname las coordenadas x, y, z del primer punto tridimensional: ";  
**leer** x1, y1, z1;

**escribir** "mencioname las otras coordenadas x, y, z, del segundo punto tridimensional: ";  
**leer** x2, y2, z2;

puntoMedio1 ← (x1 + x2)/2;  
 puntoMedio2 ← (y1 + y2)/2;  
 puntoMedio3 ← (z1 + z2)/2;

**escribir** "El punto medio se encuentra en las coordenadas: ";  
**escribir** puntoMedio1 " ", puntoMedio2 " ", puntoMedio3;

**FinAlgoritmo**

- Foto de ejecución:

```
*** Ejecución Iniciada. ***
mencioname las coordenadas x, y, z del primer punto tridimensional:
al:
> 4
> 5
> 6
mencioname las otras coordenadas x, y, z, del segundo punto tridimensional:
> 7
> 8
> 2
El punto medio se encuentra en las coordenadas:
5.5, 6.5, 4
*** Ejecución Finalizada. ***
```

- Este pseudocódigo fue añadido pues porque me parecía interesante como se puede expresar el código para hallar el punto medio entre todos los puntos.



## Algoritmos Selectivos

1. Escribe un algoritmo que pida un número y que imprima si el número proporcionado es negativo, positivo o cero según sea el caso.

- entradas: numero como el numero proporcionado
- salida: la condición que si es negativo, positivo o cero, se imprima que diga que es.
- Foto de pseudocódigo:

### Algoritmo numeros

*//entradas: numero como el numero proporcionado*  
*//salida: la condicion que si es negativo, positivo o cero, se imprima que diga que es.*  
*//proceso de prueba -1, El numero proporcionado es negativo*

*//ENTRADA*

**definir** num como real;

num ← 0.0;

**Escribir** "escribe el numero que piensas";

**leer** num;

*//SALIDA*

**si** num < 0 **entonces**;

**escribir** "El numero proporcionado es negativo";

**sino si** num > 0 **entonces**;

**escribir** "El numero proporcionado es positivo";

**sino**;

**Escribir** "El numero propocionado es cero";

**FinSi**

**FinSi**

**FinAlgoritmo**

- Foto de ejecución:

\*\*\* Ejecución Iniciada. \*\*\*

escribe el numero que piensas

> -1

El numero proporcionado es negativo

\*\*\* Ejecución Finalizada. \*\*\*

- Este algoritmo lo elegí porque en mi opinión, fue el primero para empezar con las estructuras selectivas

2. Un empleado trabaja 8 horas diario, si trabaja más horas se consideran extras y se pagan doble. El pago por hora es de 150 pesos. Escribe un algoritmo que reciba un número de horas trabajadas y que devuelva el sueldo total que le corresponde tomando en cuenta horas extra. Las horas se capturan enteras, valida que no sean más de 12 horas.

- entradas: horas como las horas trabajadas
- salidas: la suma total de las horas pagadas, cada hora a partir de 8, son 150, más de 8, el excedente vale 300, no se puede más de 12 horas.
- Foto de pseudocódigo:

```

Algoritmo horasTrabajadas
 //entradas: horas como las horas trabajadas
 //salidas: la suma total de las horas pagadas, cada hora a partir de 8, son 150, mas de 8, el excedente vale 300
 //no se puede mas de 12 horas.
 //proceso de prueba: 8, el pago total será de: 1200 pesos

 //ENTRADAS
 definir horas, pagoTotal como entero;

 horas \leftarrow 0.0;

 escribir "¿Cuántas horas trabajó? No se puede mas de 12 horas";
 leer horas;

 //PROCESO Y SALIDA
 si horas \leq 8 entonces;
 pagoTotal \leftarrow horas * 150;
 escribir "El pago total es de: ", pagoTotal " pesos";
 sino si horas > 8 y horas \leq 12 entonces;
 pagoTotal \leftarrow (horas - 8) * 300 + 1200;
 escribir "El pago total es de: ", pagoTotal " pesos, esto contemplando las horas extra por 300 cada una";
 sino;
 escribir "no se puede trabajar mas de 12 horas";
 finsi
FinSi

FinAlgoritmo

```

- Foto de ejecución:

\*\*\* Ejecución Iniciada. \*\*\*

¿Cuántas horas trabajó? No se puede mas de 12 horas

> 11

El pago total es de: 2100 pesos, esto contemplando las horas extra por 300 cada una

\*\*\* Ejecución Finalizada. \*\*\*

- Este algoritmo lo añadí porque es parte de los que pensaba que iban a ser complicados de plantear

3. Algoritmo que permite determinar cuántos productos se pueden comprar con una cantidad de dinero. Sólo se puede comprar de un tipo de productos a la vez: teléfonos (1000 pesos), cargadores (350 pesos) y estuches (200 pesos).

- entradas: dinero
- salidas: sobrante, teléfonos, cargadores, estuches como cantidad de los productos y el dinero sobrante
- Foto de pseudocódigo:

```
Algoritmo compras
//entradas: dinero
//salidas: sobrante, telefonos, cargadores, estuches como cantidad de los productos y el dinero sobrante
//entradas
definir dinero, sobrante, telefonos, cargadores, estuches como real;

//inicialización
dinero ← 0.0;
sobrante ← 0.0;
telefonos ← 0.0;
cargadores ← 0.0;
estuches ← 0.0;

escribir "¿Cuánto dinero tienes para gastar?";
leer dinero;

//proceso
si dinero ≥ 1000 y telefonos ≤ 1 entonces;
 telefonos ← 1;
 sobrante ← dinero - 1000;
sino;
 telefonos ← 0.0;
FinSi

si sobrante ≥ 350 y cargadores ≤ 1 entonces;
 cargadores ← 1;
 sobrante ← sobrante - 350;
sino;
 cargadores ← 0.0;
FinSi

si sobrante ≥ 200 y estuches ≤ 1 entonces;
 estuches ← 1;
 sobrante ← sobrante - 200;
sino;
 estuches ← 0.0;
FinSi

//salida
escribir "Lo que puedes comprar es: ", telefonos " telefonos, ", cargadores " cargadores";
Escribir estuches " estuches, todo esto y tienes un resto de: ", sobrante " pesos";
FinAlgoritmo
```

- Foto de ejecución:

```
*** Ejecución Iniciada. ***
¿Cuánto dinero tienes para gastar?
> 10000
Lo que puedes comprar es: 1 telefonos, 1 carga
dores
1 estuches, todo esto y tienes un resto de: 84
50 pesos
*** Ejecución Finalizada. ***
```

- Este algoritmo lo elegí porque la primera vez me salió mal, pero después entendí bien lo que tenía que hacer.

4. Escriba un algoritmo que reciba las medidas de los tres lados de un triángulo y que imprima si el triángulo es “escaleno”, “isósceles” o “equilátero”.

- entradas: lado, lado1, lado2 como los vertices del triangulo.
- salidas: dependiendo de los valores de los angulos, definirá si es escaleno, isosceles o equilatero.
- Foto de código:

**Algoritmo** Triangulos

```
//entradas: lado, lado1, lado2 como los vertices del triangulo.
//salidas: dependiendo de los valores de los angulos, definirá si es escaleno, isosceles o equilatero.
//proceso de prueba: 90, 90, 90, el triangulo es equilatero
definir lado, lado1, lado2 Como Real;
//inicialización
lado ← 0.0;
lado1 ← 0.0;
lado2 ← 0.0;
//entradas
escribir "mencioname el valor de los angulos de cada uno de los vertices del triangulo";
leer lado, lado1, lado2;

//Proceso y salida
si lado = lado1 y lado1 = lado2 entonces;
| escribir "el triangulo es equilatero";
sino si lado = lado1 y lado1 ≠ lado2 o lado = lado2 y lado ≠ lado1 o lado1 = lado2 y lado1 ≠ lado entonces;
| | escribir "El triangulo es isosceles";
| sino;
| Escribir "El triangulo es escaleno";
| finSi
FinSi
FinAlgoritmo
```

- Foto de ejecución:  
mencioname el valor de los angulos de cada uno  
de los vertices del triangulo

> 60

> 60

> 60

el triangulo es equilatero

\*\*\* Ejecución Finalizada. \*\*\*

- Este algoritmo lo elegí porque me gustaba el hecho de identificar cual tipo era cada uno de los triángulos, me pareció fantástico

5. El dueño de un estacionamiento requiere un algoritmo que le permita determinar cuánto debe cobrar por el uso del estacionamiento a sus clientes. Las tarifas que se tienen son las siguientes: Las dos primeras horas a \$5.00 c/u, las siguientes tres a \$4.00 c/u, las cinco siguientes a \$3.00 c/u y después de diez horas el costo por cada una es de dos pesos.

- entrada: horas
- salida: el costo por las horas hechas, dependiendo de la cantidad de horas.

- Foto de código:

**Algoritmo** Estacionamiento

```
//entrada: horas
//salida: el costo por las horas hechas, dependiendo de la cantidad de horas.
//Proceso de prueba: 5, el costo será de 22 pesos
```

```
definir horas, precio como real;
```

```
horas ← 0.0;
```

```
escribir "¿Cuántas horas paso en el estacionamiento?";
leer horas;
```

```
si horas ≤ 2 entonces;
```

```
 precio ← trunc((horas + 0.9)) * 5;
```

```
sino si horas > 2 y horas ≤ 5 entonces;
```

```
 precio ← (trunc(horas + 0.9) - 2) * 4 + 10;
```

```
sino si horas > 5 y horas ≤ 10 entonces;
```

```
 precio ← (trunc(horas + 0.9) - 5) * 3 + 22;
```

```
sino;
```

```
 precio ← (trunc(horas + 0.9) - 10) * 2 + 37;
```

```
FinSi
```

```
finSi
```

```
FinSi
```

```
escribir "El precio es de: ", precio " pesos";
```

**FinAlgoritmo**

- Foto de ejecución:

```
*** Ejecución Iniciada. ***
```

```
¿Cuántas horas paso en el estacionamiento?
```

```
> 10
```

```
El precio es de: 37 pesos
```

```
*** Ejecución Finalizada. ***
```

- Este pseudocódigo lo añadí porque me parecía muy limpia su estructura

6. Se tiene el nombre y la edad de tres personas. Se desea saber el nombre y la edad de la persona de menor edad. Realice el algoritmo correspondiente.

- entradas: nombre, nombre1, nombre2, edad, edad1, edad2, como los datos de las personas
- salida: el menor de las tres personas
- foto del código:

```
Algoritmo edadNombre
//entradas: nombre, nombre1, nombre2, edad, edad1, edad2, como los datos de las personas
//salida: el menor de las tres personas
//proceso de prueba: Juan, pablo, pedro, 10, 11, 9, El menor es pedro con 9 años.

definir nombre, nombre1, nombre2 Como Caracter;
definir edad, edad1, edad2 como enteros;

//Inicialización
edad ← 0.0;
edad1 ← 0.0;
edad2 ← 0.0;

//Entradas
escribir "¿Cómo se llama la primera persona?";
leer nombre;
escribir "¿Cómo se llama la segunda persona?";
leer nombre1;
escribir "¿Cómo se llama la tercera persona?";
leer nombre2;

escribir "¿Cual es la edad de ", nombre "?";
leer edad;
escribir "¿Cual es la edad de ", nombre1 "?";
leer edad1;
escribir "¿Cual es la edad de ", nombre2 "?";
leer edad2;

//Proceso y Salidas.
si edad < edad1 y edad < edad2 entonces;
| escribir nombre " Es el menor de los tres";
sino si edad1 < edad y edad1 < edad2 entonces;
| escribir nombre1 " Es el menor de los tres";
sino si edad2 < edad y edad2 < edad1 entonces;
| escribir nombre2 " Es el menor de los tres";
| finSi
| finSi
FinSi
FinAlgoritmo
```

- Foto de ejecución:

```
*** Ejecución Iniciada. ***
¿Cómo se llama la primera persona?
> Juan
¿Cómo se llama la segunda persona?
> pablo
¿Cómo se llama la tercera persona?
> pedro
¿Cual es la edad de Juan?
> 10
¿Cual es la edad de pablo?
> 11
¿Cual es la edad de pedro?
> 15
Juan Es el menor de los tres
```

- Este lo añadí porque era parte del bloque, además porque se me hizo un poco largo de hacer, cual pensaba que iba a ser menos por lo que pedía el ejercicio.

7. El presidente de la república ha decidido estimular a todos los estudiantes de una universidad mediante la asignación de becas mensuales, para esto se tomarán en consideración los siguientes criterios: Para alumnos mayores de 18 años con promedio mayor o igual a 9, la beca será de \$2000.00; con promedio mayor o igual a 7.5, de \$1000.00; para los promedios menores de 7.5 pero mayores o iguales a 6.0, de \$500.00; a los demás se les enviará una carta de invitación incitándolos a que estudien más en el próximo ciclo escolar. A los alumnos de 18 años o menores de esta edad, con promedios mayores o iguales a 9, se les dará \$3000; con promedios menores a 9 pero mayores o iguales a 8, \$2000; para los alumnos con promedios menores a 8 pero mayores o iguales a 6, se les dará \$100, y a los alumnos que tengan promedios menores a 6 se les enviará carta de invitación. Realice el algoritmo correspondiente.

- entrada: edad, promedio como valores importantes para definir la beca
- salida: La ganancia que tendrá respecto a su edad y promedio.
- Foto de código:

#### Algoritmo Beca

//entrada: edad, promedio como valores importantes para definir la beca  
//salida: La ganancia que tendrá respecto a su edad y promedio.  
//Proceso de prueba: 18, 9, la beca será de 3000 mensuales.

definir edad como entero;  
definir promedio, pago como real;

edad ← 0.0;  
promedio ← 0.0;  
pago ← 0.0;

escribir "¿Cuál es tu edad?";  
leer edad;

escribir "¿Cuál es tu promedio?";  
leer promedio;

```
si edad > 18 entonces;
 si promedio ≥ 9 entonces;
 pago ← 2000;
 sino si promedio < 9 y promedio ≥ 7.5 entonces;
 pago ← 1000;
 sino si promedio < 7.5 y promedio ≥ 6 entonces;
 pago ← 500;
 sino;
 Escribir "¡Estudia más para el siguiente ciclo escolar!";
 FinSi
FinSi
FinSi
```

```
sino;
 si promedio ≥ 9 entonces;
 pago ← 3000;
 sino si promedio < 9 y promedio ≥ 8 entonces;
 pago ← 2000;
 sino si promedio < 8 y promedio ≥ 6 entonces;
 pago ← 100;
 sino;
 Escribir "¡Estudia más para el siguiente ciclo escolar!";
 FinSi
FinSi
FinSi

si pago ≠ 0 entonces;
 escribir "El pago de la beca será de: ", pago " pesos Mensuales";
FinSi
```

FinAlgoritmo

- Foto de ejecución:

```
*** Ejecución Iniciada. ***
¿Cuál es tu edad?
> 10
¿Cuál es tu promedio?
> 3
¡Estudia más para el siguiente ciclo escolar!
*** Ejecución Finalizada. ***
```

- Este algoritmo lo elegí porque tenía una estructura curiosa, además porque me parecía que su funcionamiento era muy limpio, las salidas igual se me hacían muy bien



8. Este es un algoritmo donde da permisos del “material clasificado” al usuario, sin embargo, para que lo logre debe cumplir algunas condiciones, debe ser un usuario activo, de este, si es honorario tiene acceso al material, sino es honorario pero es un miembro veterano y tiene toda la cuota pagada, tiene todo el permiso, de lo contrario ante todas las condiciones, entonces no tiene acceso.

- Entradas: nombre, activo, antigüedad1, cuotapagada, honorario.
- salidas: Si tiene acceso al material clasificado
- foto de código:

**Algoritmo** Biblioteca

```
//entradas: nombre, activo, antigüedad1, cuotapagada, honorario.
//salidas: Si tiene acceso al material clasificado
//caso de prueba: Juan, si, 9, no, si, tiene acceso al material clasificado.
definir nombre Como Caracter; Definir antigüedad, activo, cuotapagada, honorario como entero;
//inicialización
antigüedad ← 0.0;
activo ← 0.0;
cuotapagada ← 0.0;
honorario ← 0.0;

//entradas
escribir "¿Cual es su nombre?";
leer nombre;
escribir "¿Usted es un usuario activo?";
escribir "1- si, 2- no";
leer activo;
escribir "¿Esta al día con la cuota?";
escribir "1- si, 2- no";
leer cuotapagada;
escribir "¿Tiene 10 o más años de antigüedad?";
escribir "1- si, 2- no";
leer antigüedad;
escribir "¿Usted es un miembro honorario?";
escribir "1- si, 2- no";
leer honorario;

//proceso y salida.
si activo = 1 entonces;
 si honorario == 1 entonces;
 escribir "tiene acceso al material clasificado";
 sino si antigüedad == 1 y cuotapagada == 1 entonces;
 escribir "tiene acceso al material clasificado";
 sino;
 escribir " no tiene acceso al material clasificado";
 FinSi
FinSi
sino;
 escribir "no tiene acceso al material clasificado";
FinSi
FinAlgoritmo
```

- Foto de ejecución:

```
*** Ejecución Iniciada. ***
¿Cual es su nombre?
> Bani
¿Usted es un usuario activo?
1- si, 2- no
> 1
¿Esta al día con la cuota?
1- si, 2- no
> 2
¿Tiene 10 o más años de antigüedad?
1- si, 2- no
> 2
¿Usted es un miembro honorario?
1- si, 2- no
> 1
tiene acceso al material clasificado
*** Ejecución Finalizada. ***
```

- Elegí este algoritmo, porque cuando lo realizamos en su momento, que fue en una tarea en clase, lo terminé de escribir rápido, en ese momento me había sorprendido

## Algoritmos con estructuras iterativas

1. Este algoritmo trata de un simple acumulador, suma números conforme pasa cada iteración, es decir  $1+2+3+4+5...$

- Entradas: valor, limite
- Salida: Valor, pero con la suma total de iteraciones
- Foto de código:

**Algoritmo** Acumulador

**Definir** valor **Como Entero**;

**definir** limite, indice **Como Entero**;

limite  $\leftarrow 0$ ;

valor  $\leftarrow 0$ ;

**Escribir** "Cuantos primeros números quieres que se sumen?";

**leer** limite;

**para** indice  $\leftarrow 1$  **hasta** limite **Hacer**

    valor  $\leftarrow$  valor + indice;

**FinPara**

**escribir** "El total es: ", valor;

**FinAlgoritmo**

- Foto de ejecución:

Cuantos primeros números quieres que se sumen?

> 5

El total es: 15

- Este algoritmo elegí porque era el primer ejercicio que se realizó para estructuras iterativas, fue curioso de llevar y fue con lo primero de las ideas que se fueron dando.

## 2. Algoritmo que dibuja un rectángulo de largo\*ancho con asteriscos

- Entradas: largo, ancho.
- Salida: el rectángulo con largo y ancho con asteriscos.
- Foto:

**Algoritmo** Rectangulo

**definir** largo, ancho, indiceLargo, indiceAncho **Como Entero**;

**escribir** "Mencioname el largo y luego el ancho para hacerlo rectangulo";

**leer** largo, ancho;

**para** indiceLargo  $\leftarrow$  1 **hasta** largo **hacer**

**para** indiceAncho  $\leftarrow$  1 **hasta** ancho **Hacer**

**escribir** "\*" **sin saltar**;

**FinPara**

**Escribir** " ";

**FinPara**

**FinAlgoritmo**

- Foto de ejecución:

\*\*\* Ejecución Iniciada. \*\*\*

Mencioname el largo y luego el ancho para hacerlo rectangulo

> 2

> 3

\*\*\*

\*\*\*

\*\*\* Ejecución Finalizada. \*\*\*

- Este algoritmo lo elegí, porque en su momento me fascinó el hecho de dibujar el rectángulo con ello, además de que en el momento de hacerlo no sabía para nada como hacerlo.

### 3. Algoritmo que saca el promedio de la calificación de n estudiantes en el salón

- entrada: cantidad, calificación.
- salida: sumatotal, promedio.
- Foto de código:

**Algoritmo** promediosalon

*//entrada: cantidad, calificación.*

*//salida: sumatotal, promedio.*

*//Caso de prueba: cantidad <- 3, calificaciones <- 10, 10, 10 suma total<- 30, promedio<- 10.*

*//Este algoritmo sirve para sacar el promedio de un grupo de estudiantes con calificaciones sobre 100*

**definir** cantidad, cal, sumaTotal, indice **Como Entero**;

**definir** promedio **Como Real**;

cantidad  $\leftarrow$  0.0;

cal  $\leftarrow$  0.0;

sumaTotal  $\leftarrow$  0.0;

promedio  $\leftarrow$  0.0;

**escribir** "Cuantos estudiantes son?";

**leer** cantidad;

**para** indice  $\leftarrow$  1 **hasta** cantidad **Hacer**

**escribir** "Cual fue la calificación sobre 100 del estudiante numero: " indice " ?";

**leer** cal;

    sumaTotal  $\leftarrow$  sumaTotal + cal;

**FinPara**

promedio  $\leftarrow$  sumaTotal/cantidad;

**escribir** "El promediode los estudiantes es de: ", promedio;

**FinAlgoritmo**

- Foto de ejecución:

\*\*\* Ejecución Iniciada. \*\*\*

Cuantos estudiantes son?

> 2

Cual fue la calificación sobre 100 del estudiante numero: 1 ?

> 70

Cual fue la calificación sobre 100 del estudiante numero: 2 ?

> 70

El promediode los estudiantes es de: 70

\*\*\* Ejecución Finalizada. \*\*\*

- Este algoritmo lo elegí, porque ví la evolución de este ejercicio que me pareció increíble, porque anteriormente ya lo habíamos hecho pero la definición de variables era demasiada, ya acá son valores que se van sobrescribiendo.

#### 4. Algoritmo que dibuja una tabla de las funciones seno, coseno, arcotang de 1 a -1

- Entradas ninguna solamente imprime una tabla de las funciones
- Salida: Imprime una tabla de las funciones seno, coseno y arcotangente.
- Foto de código:

**Algoritmo** senoxcosxarctangx

*//entradas y salidas: ninguna solamente imprime una tabla de las funciones  
//seno, coseno y arcotangente de x de valores de x desde -1 hasta 1  
//con intervalos de 0.2.*

**definir** x, seno, coseno, arcotang **como** real;

**definir** indice **como** entero;

x ← 0.0;

seno ← 0.0;

coseno ← 0.0;

arcotang ← 0.0;

**escribir** "[ x ][ seno x ][ cosenox ][ arco Tangente X ]" **sin saltar**;  
**escribir** "";

**para** indice ← -1 **hasta** 1 **con** paso 0.2 **Hacer**

x ← indice;

seno ← sen(x);

coseno ← cos(x);

arcotang ← atan(x);

**escribir** "| ",x " || ",seno " || ",coseno " || ",arcotang " |";

**FinPara**

**FinAlgoritmo**

- Foto de ejecución:

\*\*\* Ejecución Iniciada. \*\*\*

```
[x][seno x][cosenox][arco Tangente X]
-1		-0.8414709848		0.5403023059		-0.7853981634
-0.8		-0.7173560909		0.6967067093		-0.6747409422
-0.6		-0.5646424734		0.8253356149		-0.5404195003
-0.4		-0.3894183423		0.921060994		-0.3805063771
-0.2		-0.1986693308		0.9800665778		-0.1973955598
0		0		1		0
0.2		0.1986693308		0.9800665778		0.1973955598
0.4		0.3894183423		0.921060994		0.3805063771
0.6		0.5646424734		0.8253356149		0.5404195003
0.8		0.7173560909		0.6967067093		0.6747409422
1		0.8414709848		0.5403023059		0.7853981634
```

\*\*\* Ejecución Finalizada. \*\*\*

- Este algoritmo lo añadí porque cuando lo realicé pensé que iba a ser imposible de hacerse, sin embargo, fue mucho más fácil de lo que pensaba.

5. Algoritmo que suma la cantidad de múltiplos de 5 hasta limite, es decir  
5+10+15+20...

- entrada: limite este debe estar en múltiplo de 5.
- salida: suma como la suma de los enteros múltiplos de 5 que hay entre 0 y limite
- Foto de código:

**Algoritmo** multiplos

*//entrada: limite*

*//salida: suma como la suma de los enteros múltiplos de 5 que hay entre 0 y limite*

*//caso de prueba: 10, la suma es: 15.*

**definir** limite, indice, suma **Como Entero**;

limite  $\leftarrow$  0.0;

suma  $\leftarrow$  0.0;

**escribir** "¿Cuál será el limite?";

**leer** limite;

**para** indice  $\leftarrow$  0 **hasta** limite **con paso** 5 **Hacer**

    suma  $\leftarrow$  suma + indice;

**FinPara**

**escribir** "La suma de todos los múltiplos de 0 hasta el límite es de: ", suma;

**FinAlgoritmo**

- Foto de ejecución:

\*\*\* Ejecución Iniciada. \*\*\*

¿Cuál será el limite?

> 10

La suma de todos los múltiplos de 0 hasta el l

ímite es de: 15

\*\*\* Eiecución Finalizada. \*\*\*

- La razón de haber agregado este ejercicio es por lo super fácil que fue de hacer jajaj, en realidad simplemente le cambie un valor diferente al contador.

## 6. Algoritmo que imprime la secuencia de Fibonacci hasta n iteraciones

- entrada: limite como el limite de numeros de fibonacci a imprimir.
- salida: nums como numeros de la secuencia fibonacci de 0 hasta limite
- Foto de código:

### Algoritmo fibonacci

*//entrada: limite como el limite de numeros de fibonacci a imprimir.*

*//salida: nums como numeros de la secuencia fibonacci de 0 hasta limite*

*//caso de prueba: limite <- 5, numeros <- 1, 1, 2, 3, 5;*

**definir** limite, numanterior, numnuevo, suma, indice **como entero**;

limite  $\leftarrow$  0.0;

numanterior  $\leftarrow$  0;

numnuevo  $\leftarrow$  0.0;

indice  $\leftarrow$  0.0;

suma  $\leftarrow$  0.0;

*//entrada*

**escribir** "¿Hasta qué posición de numeros de la secuencia fibonacci quieres imprimir?";

**leer** limite;

### Repetir

*//proceso.*

numnuevo  $\leftarrow$  numnuevo + numanterior;

numanterior  $\leftarrow$  numnuevo - numanterior;

**si** numnuevo = 0 **entonces**;

numnuevo  $\leftarrow$  1;

**FinSi**

*//salida*

**escribir** numnuevo;

indice = indice + 1;

**Hasta que** indice = limite

### FinAlgoritmo

- Foto de ejecución:

¿Hasta qué posición de numeros de la secuencia  
fibonacci quieres imprimir?

> 5

1

1

2

3

5

- Este ejercicio lo elegí porque en su momento que lo hice, tardé como 30 minutos para poder pensar como plantear el código, aunque en realidad era mas sencillo de lo esperado



## 7. Algoritmo que imprime la factorial de un valor insertado

- Entradas: n como el numero al que se le aplicará la factorial
- Salida: la factorial de ese numero.
- Foto de código:

### **Algoritmo** Factorial

*//Entradas: n como el numero al que se le aplicará la factorial*  
*//Salida: la factorial de ese numero.*

```
definir indice, n, factori como entero;
escribir "dime el limite";
factori ← 1;
n ← 0.0;
leer n;
para indice ← 1 hasta n hacer;
 factori ← factori *indice;
FinPara
escribir "El factorial de: ", n " Es igual a: ", factori;
```

### **FinAlgoritmo**

- Foto de ejecución:  
dime el limite  
> 10  
El factorial de: 10 Es igual a: 3628800
- La razón de haber agarrado este código, es porque sentía increíble el hecho de que se puede hacer algo que te simplifica todo el trabajo, como ir multiplicando uno por uno los valores para hacer la factorial.

8. Este algoritmo trata de una empresa donde se producen sillas, donde se evalúan la cantidad de sillas que produjo 3 trabajadores de la misma empresa.

- entradas: nombre1, nombre2, nombre3 como los nombres de los trabajadores. después preguntar la cantidad de sillas que produjeron cada día de los 5 días de la semana.
- salidas: nombres, la suma de su produccion de toda la semana, quien produjo más entre los tres y la suma de todas las sillas hechas por los tres trabajadores
- Foto de código:

```

Algoritmo Sillas
 //entradas: nombre1, nombre2, nombre3 como los nombres de los trabajadores. Despues preguntar la cantidad de sillas que produjeron
 //cada dia de los 5 días de la semana.
 //salidas: nombres, la suma de su produccion de toda la semana, quien produjo más entre los tres y la suma de todas las sillas hechas
 //por los tres trabajadores

 definir nombre1, nombre2, nombre3, mejor como caracter;
 definir sillasHechas, total, semanaTrabajador1, semanaTrabajador2, semanaTrabajador3, produjoMas, totalDelosTres, indice como real;

 sillasHechas ← 0.0;
 total ← 0.0;
 produjoMas ← 0.0;
 totalDelosTres ← 0.0;
 semanaTrabajador1 ← 0.0;
 semanaTrabajador2 ← 0.0;
 semanaTrabajador3 ← 0.0;

 //Entrada:
 escribir "Dime el nombre del trabajador numero 1";
 leer nombre1;
 escribir "Dime el nombre del trabajador numero 2";
 leer nombre2;
 escribir "Dime el nombre del trabajador numero 3";
 leer nombre3;

 //Entrada
 //Proceso
 para indice ← 1 hasta 5 hacer;
 escribir "¿Cuántas sillas produjo el día: " indice " el trabajador ", nombre1 " ?";
 leer total;
 semanaTrabajador1 ← semanaTrabajador1 + total;
 FinPara
 para indice ← 1 hasta 5 hacer;
 escribir "¿Cuántas sillas produjo el día: " indice " el trabajador ", nombre2 " ?";
 leer total;
 semanaTrabajador2 ← semanaTrabajador2 + total;
 FinPara
 para indice ← 1 hasta 5 hacer;
 escribir "¿Cuántas sillas produjo el día: " indice " el trabajador ", nombre3 " ?";
 leer total;
 semanaTrabajador3 ← semanaTrabajador3 + total;
 FinPara

 //Salida
 totalDelosTres ← semanaTrabajador1 + semanaTrabajador2 + semanaTrabajador3;
 escribir "El trabajador: ", nombre1 " produjo: ", semanaTrabajador1 " Sillas";
 escribir "El trabajador: ", nombre2 " produjo: ", semanaTrabajador2 " Sillas";
 escribir "El trabajador: ", nombre3 " produjo: ", semanaTrabajador3 " Sillas";
 escribir "-----";

```

```

Escribir "El total de todas las sillas producidas entre los tres trabajadores es de: ", totalDelosTres " sillas";
si semanaTrabajador1 = semanaTrabajador2 y semanaTrabajador1 > semanaTrabajador3 entonces
 escribir nombre1 " y " nombre2 " produjeron lo mismo y más que ", nombre3 " con una cantidad de ", semanaTrabajador1 " Sillas";
sino si semanaTrabajador3 = semanaTrabajador2 y semanaTrabajador3 > semanaTrabajador1 Entonces
 escribir nombre3 " y " nombre2 " produjeron lo mismo y más que ", nombre1 " con una cantidad de ", semanaTrabajador3 " Sillas";
sino si semanaTrabajador1 = semanaTrabajador3 y semanaTrabajador1 > semanaTrabajador2 Entonces
 escribir nombre1 " y " nombre3 " produjeron lo mismo y más que ", nombre2 " con una cantidad de ", semanaTrabajador3 " Sillas";
sino si semanaTrabajador1 = semanaTrabajador2 y semanaTrabajador2 = semanaTrabajador3 Entonces
 escribir "Los tres produjeron lo mismo con una cantidad de ", semanaTrabajador1 " Sillas";
sino si semanaTrabajador1 > semanaTrabajador2 y semanaTrabajador1 > semanaTrabajador3 Entonces
 escribir "El trabajador: " nombre1 " fué el que produjo más entre los tres con una cantidad de: " semanaTrabajador1 " sillas";
sino si semanaTrabajador2 > semanaTrabajador1 y semanaTrabajador2 > semanaTrabajador3 Entonces
 escribir "El trabajador: " nombre2 " fué el que produjo más entre los tres con una cantidad de: " semanaTrabajador2 " sillas";
sino si semanaTrabajador3 > semanaTrabajador1 y semanaTrabajador3 > semanaTrabajador2 Entonces
 escribir "El trabajador: " nombre3 " fué el que produjo más entre los tres con una cantidad de: " semanaTrabajador3 " sillas";
 FinSi
 FinSi
finsi
FinSi
finsi
FinSi
finsi
FinSi

```

FinAlgoritmo

- Foto de ejecución:

```

*** Ejecución Iniciada. ***
Dime el nombre del trabajador numero 1
> Juan
Dime el nombre del trabajador numero 2
> Pablo
Dime el nombre del trabajador numero 3
> Pedro
¿Cuántas sillas produjo el día: 1 el trabajador Juan ?
> 100
¿Cuántas sillas produjo el día: 2 el trabajador Juan ?
> 1001
¿Cuántas sillas produjo el día: 3 el trabajador Juan ?
> 10
¿Cuántas sillas produjo el día: 4 el trabajador Juan ?
> 100
¿Cuántas sillas produjo el día: 5 el trabajador Juan ?
> 10
¿Cuántas sillas produjo el día: 1 el trabajador Pablo ?
> 100
¿Cuántas sillas produjo el día: 2 el trabajador Pablo ?
> 100001
¿Cuántas sillas produjo el día: 3 el trabajador Pablo ?
> 10
¿Cuántas sillas produjo el día: 4 el trabajador Pablo ?
> 1000
¿Cuántas sillas produjo el día: 5 el trabajador Pablo ?
> 100
¿Cuántas sillas produjo el día: 1 el trabajador Pedro ?
> 100
¿Cuántas sillas produjo el día: 2 el trabajador Pedro ?
> 10000
¿Cuántas sillas produjo el día: 3 el trabajador Pedro ?
> 100
¿Cuántas sillas produjo el día: 4 el trabajador Pedro ?
> 100
¿Cuántas sillas produjo el día: 5 el trabajador Pedro ?
> 1000
El trabajador: Juan produjo: 1221 Sillas
El trabajador: Pablo produjo: 101211 Sillas
El trabajador: Pedro produjo: 11300 Sillas

El total de todas las sillas producidas entre los tres trabajadores es de: 113732 sillas
El trabajador: Pablo fué el que produjo más entre los tres con una cantidad de: 101211 sillas
*** Ejecución Finalizada. ***

```

- Elegí este algoritmo porque se me hizo muy gracioso lo ineficiente que es para sacar unos simples resultados, el código es muy largo.

9. Algoritmo que imprime el valor de un dinero insertado que son 1500 que va aumentando con el interés por el banco, este se imprime desde el año 1961 hasta el año n donde  $n > 1961$ .

- Entradas: añoFinal como el año limite que se calcula para ir revisando el interés compuesto
- Salidas: cantidadFinal como la cantidad de dinero final que se tenia cada año hasta añoFinal
- Foto de código:

```

Algoritmo Mis1
 //Entradas: añoFinal como el año limite que se calcula para ir revisando el interés compuesto
 //Salidas: cantidadFinal como la cantidad de dinero final que se tenia cada año hasta añoFinal
 //Caso de prueba: Sabemos que la cantidad inicial es de 1500, y desde 1961 añoFinal<- 1963 cantidadFinal<- 1500, 1,725, 1983.75

 definir anoInicial, anoFinal, cantidadInicial como entero;
 definir cantidadFinal Como Real;

 anoInicial \leftarrow 1961;
 anoFinal \leftarrow 0.0;
 cantidadInicial \leftarrow 1500;
 cantidadFinal \leftarrow cantidadInicial;

 //Entrada
 escribir "¿Hasta qué año limite quieres revisar el progreso anual del interés? (más de 1961)";
 leer anoFinal;
 si anoFinal < anoInicial Entonces
 escribir "No se puede hasta ese año, es menor que el inicial";
 sino;
 //Proceso y Salida
 para anoInicial \leftarrow anoInicial hasta anoFinal Hacer
 Escribir "En el año ", anoInicial " Se tendrá: ", cantidadFinal " pesos";
 cantidadFinal \leftarrow cantidadFinal *1.15;
 FinPara
 FinSi

FinAlgoritmo

```

- Foto de ejecución:

```

*** Ejecución Iniciada. ***
¿Hasta qué año limite quieres revisar el progreso anual
del interés? (más de 1961)
> 1965
En el año 1961 Se tendrá: 1500 pesos
En el año 1962 Se tendrá: 1725 pesos
En el año 1963 Se tendrá: 1983.75 pesos
En el año 1964 Se tendrá: 2281.3125 pesos
En el año 1965 Se tendrá: 2623.509375 pesos

```

- Este algoritmo lo elijo porque forma parte de una de mis pruebas de desempeño que fueron clave para mi aprendizaje.

## 10. Algoritmo que saca la desviación estándar de un conjunto de valores.

- Entrada: dato;
- Salida: La desviación estándar de los datos
- Foto de código:

### Algoritmo Mis3

```
//Entrada: dato;
//Salida: La desviación estandar de los datos
//Caso de prueba: dato(1-10) <- 1,1,1,1,1,2,2,2,2,2 desviacion <- 0.527

definir dato, desviacion, indice, promedio, suma, varianza, varianza2 como real;
dato ← 0.0;
desviacion ← 0.0;
promedio ← 0.0;
suma ← 0.0;
indice ← 1;
varianza ← 1;
varianza2 ← 0.0;

//entrada
mientras indice ≤ 10 Hacer
 escribir "Dime el valor del dato numero: ", indice;
 leer dato;
 suma ← dato + suma;
 indice ← indice + 1;
FinMientras
promedio ← suma/10;
indice ← 1;
mientras indice ≤ 10 Hacer
 escribir "Repiteme el dato numero: ", indice;
 leer dato;

 varianza ← (dato - promedio)2;
 varianza2 ← varianza2 + varianza;
 indice ← indice + 1;
FinMientras

desviacion ← raiz((varianza2)/9);
escribir "la desviacion estandar es de: ", desviacion;
FinAlgoritmo
```

- Foto de ejecución:

```
*** Ejecución Iniciada. ***
Dime el valor del dato numero: 1
> 1
Dime el valor del dato numero: 2
> 1
Dime el valor del dato numero: 3
> 1
Dime el valor del dato numero: 4
> 1
Dime el valor del dato numero: 5
> 1
Dime el valor del dato numero: 6
> 2
Dime el valor del dato numero: 7
> 2
Dime el valor del dato numero: 8
> 2
Dime el valor del dato numero: 9
> 2
Dime el valor del dato numero: 10
> 2
Repíteme el dato numero: 1
> 1
Repíteme el dato numero: 2
> 1
Repíteme el dato numero: 3
> 1
Repíteme el dato numero: 4
> 1
Repíteme el dato numero: 5
> 1
Repíteme el dato numero: 6
> 2
Repíteme el dato numero: 7
> 2
Repíteme el dato numero: 8
> 2
Repíteme el dato numero: 9
> 2
Repíteme el dato numero: 10
> 2
la desviación estandar es de: 0.5270462767
*** Ejecución Finalizada. ***
```

- Elegí este pseudocódigo porque fue parte de la misma prueba de desempeño, pero básicamente representa a la introducción de los vectores, pues en ese momento no podíamos escribir vectores, sin embargo, esto nos demuestra su importancia.

## Ejercicios con arreglos: Vectores.

1. Se tiene un vector de estaturas de n alumnos, determinar cuál es la estatura promedio y cuántos alumnos tienen una estatura arriba del promedio.

- Entradas: estudiantes, altura el primero como el array de 20, el segundo la altura de cada uno
- salida: como el promedio como promedio de estatura y fueraDelpromedio como la cantidad de estudiantes fuera del promedio
- Foto de código:

```
Algoritmo estaturaPromedio
//Entradas: estudiantes, altura el primero como el array de 20, el segundo la altura de c
//salida: como el promedio como promedio de estatura y fueraDelpromedio como la cantidad
//caso de prueba: 1.70, 1.70, 1.90 promedio <- 1.7666, fueraDelpromedio <-1.

definir estudiantes, cantidad, altura, fueraDelpromedio, suma, indice Como Entero;
definir promedio Como real;

altura ← 0;
fueraDelpromedio ← 0;
suma ← 0;
promedio ← 0.0;
cantidad ← 0;

escribir "Mencioname la cantidad de estudiantes";
leer cantidad;

dimensionar estudiantes(cantidad);

escribir "Menciona las alturas en centimetros";
para indice ← 0 hasta cantidad - 1 Hacer
 escribir "Mencioname la altura del estudiante numero: ", indice;
 leer altura;
 estudiantes(indice) ← altura;
 suma ← suma + altura;
FinPara

promedio ← suma/cantidad;

para indice ← 0 hasta cantidad -1 hacer
 si estudiantes(indice) > promedio Entonces
 fueraDelpromedio = fueraDelpromedio +1;
 FinSi
FinPara

escribir "El promedio de altura es de: ", promedio;
escribir fueraDelpromedio " estudiantes estan arriba del promedio";
FinAlgoritmo
```

- Foto de ejecución:

```
Mencioname la cantidad de estudiantes
> 2
Menciona las alturas en centimetros
Mencioname la altura del estudiante numero: 0
> 180
Mencioname la altura del estudiante numero: 1
> 50
El promedio de altura es de: 115
1 estudiantes estan arriba del promedio
```

- Seleccioné este ejercicio porque me pareció cómodo de hacer, aunque anteriormente pensaba que sería muy difícil de hacer, pero ahí entendí que los vectores si te facilitan las cosas mucho más.



2. Escriba un algoritmo que permita insertar números en un arreglo de 5 elementos (al inicio todos son cero) pero siempre en la primera posición. Cada vez que se inserta un valor se desplazan todos los números una posición hacia atrás el valor.

- entradas: num
- salida: vector como el vector de numeros que se irá corriendo
- Foto de código:

```
Algoritmo ejercicio7
//entradas: num
//salida: vector como el vector de numeros que se irá corriendo
//caso de prueba: 0,0,0,0,0 num <- 1 vector <- 1,0,0,0,0

definir num, vector, indice, limite Como Entero;

num ← 0;
limite ← 5;
Dimensionar vector(5+1);

para indice ← 1 hasta limite Hacer
 vector(indice) ← 0;
FinPara

escribir "Inserta un valor entero positivo, escribe -1 para finalizar";
mientras num ≠ -1 hacer
 para indice ← 1 hasta limite Hacer
 escribir vector(indice) " " sin saltar;
 FinPara
 escribir "";
 leer num;

 vector(5) ← vector(4);
 vector(4) ← vector(3);
 vector(3) ← vector(2);
 vector(2) ← vector(1);
 vector(1) ← num;
FinMientras
FinAlgoritmo
```

- Foto de ejecución:

```
0 0 0 0 0
> 3
3 0 0 0 0
> 2
2 3 0 0 0
> 1
1 2 3 0 0
> 4
```

línea

- Pongo este ejercicio porque me pareció divertido de probar al final porque se ve muy atractivo en su funcionamiento y lógica.

3. Escriba un algoritmo que reciba un arreglo de n valores y que los ordene de menor a mayor.

- Entradas: nums como el vector de los numeros desordenado, num como el numero a insertar en el vector y espacio como el espacio del vector nums
- Salidas: el vector nums pero ordenado de manera ascendente
- Foto de código:

```

definir nums, num, espacio, indice, aux Como Entero;
definir ordenado, ordenadoProceso Como Logico;

ordenado \leftarrow falso;
ordenadoProceso \leftarrow Verdadero;
aux \leftarrow 0;
espacio \leftarrow 0;

escribir "¿De cuantos espacios es tu vector?";
leer espacio;

Dimensionar nums[espacio];

escribir "Mencioname uno por uno los valores de tu vector";
para indice \leftarrow 0 hasta espacio-1 Hacer
 leer num;
 nums[indice] \leftarrow num;
FinPara

escribir "Vector inicial: ";
para indice \leftarrow 0 hasta espacio-1 Hacer
 escribir sin saltar nums[indice] ", ";
FinPara

mientras ordenado == falso hacer
 ordenadoProceso \leftarrow verdadero;
 indice \leftarrow 0;

 Repetir
 si indice < espacio-1 entonces
 si nums[indice] > nums[indice+1] Entonces
 ordenadoProceso \leftarrow falso;
 FinSi
 FinSi

 indice \leftarrow indice+1;

```

```

Hasta Que ordenadoProceso == falso o indice == espacio;
ordenado ← ordenadoProceso;

si ordenado == falso Entonces
 para indice ← 0 hasta espacio -1 Hacer
 si indice < espacio-1 entonces
 si nums[indice] > nums[indice+1] Entonces
 aux ← nums[indice+1];
 nums[indice+1] ← nums[indice];
 nums[indice] ← aux;
 FinSi
 FinSi
 FinPara
FinSi
FinMientras

escribir "";
escribir "Vector final: ";
para indice←0 hasta espacio-1 Hacer
 escribir sin saltar nums[indice] ", ";
FinPara
escribir "";
inAlgoritmo

```

- Foto de ejecución:

```

> 3
Mencioname uno por uno los valores de tu vector
> 3
> 4
> 2
Vector inicial:
3, 4, 2,
Vector final:
2, 3, 4,

```

- Este lo elijo, porque en ese momento nunca supe como ordenar un vector, ese fue como mi “manera de ordenarlo” aunque en el momento que se dio la clase de ordenar vectores con métodos, ya empecé ser más eficiente.

4. Escriba un algoritmo que permita capturar un arreglo, que calcule la media, mediana, moda y rango de los valores de ese arreglo. Considere únicamente moda unimodal.

- entradas: nums, como el numero en un espacio indice del vector nums
- salidas: media, mediana, moda y rango
- Foto de código:

**Algoritmo** estadisticaVector

```
//entradas: nums, como el numero en un espacio indice del vector nums
//salidas: media, mediana, moda y rango
//restricciones: unicamente unimodal
//caso de prueba: nums <- 2, 2, 0, 4, media <- 2, mediana <- 2, moda <- 2 rango<- 4;
```

```
definir nums, num, espacio, indice, indice2, aux Como Entero;
//definimos los datos que se sacarán de los datos
definir media, mediana Como Real;
definir moda, rango, repeticiones, repeticionesDefinido, suma Como Entero;
definir ordenado, ordenadoProceso Como Logico;
```

```
media <- 0.0;
mediana <- 0.0;
moda <- 0;
rango <- 0;
repeticiones <- 0;
repeticionesDefinido <- 0;
suma <- 0;
aux <- 0;
espacio <- 0;
ordenado <- falso;
ordenadoProceso <- Verdadero;
```

```
escribir "¿Cuántos numeros insertarás?";
leer espacio;
```

```
Dimensionar nums[espacio];
```

```
escribir "Mencioname uno por uno los valores";
para indice <- 0 hasta espacio-1 Hacer
 leer num;
 nums[indice] <- num;
FinPara
```

```
Repetir
 si indice < espacio-1 entonces
 si nums[indice] > nums[indice+1] Entonces
 ordenadoProceso ← falso;
 FinSi
 FinSi

 indice ← indice+1;
Hasta Que ordenadoProceso == falso o indice == espacio;
ordenado ← ordenadoProceso;

si ordenado == falso Entonces
 para indice ← 0 hasta espacio -1 Hacer
 si indice < espacio-1 entonces
 si nums[indice] > nums[indice+1] Entonces

 aux ← nums[indice+1];
 nums[indice+1] ← nums[indice];
 nums[indice] ← aux;

 FinSi
 FinSi
 FinPara
FinSi
FinMientras
escribir "Los valores ordenados son: ";
para indice←0 hasta espacio-1 Hacer
 escribir sin saltar nums[indice] ", ";
FinPara
escribir "";
```

```
//hallamos los valores
para indice \leftarrow 0 hasta espacio-1 Hacer
 suma \leftarrow suma + nums[indice];
FinPara

para indice \leftarrow nums[0] hasta nums[espacio-1] Hacer
 repeticiones \leftarrow 0;

 para indice2 \leftarrow 0 hasta espacio-1 Hacer

 si indice2 < espacio-1 entonces
 si indice == nums[indice2] Entonces
 repeticiones \leftarrow repeticiones+1;
 FinSi
 FinSi

 FinPara

 si repeticiones > repeticionesDefinido Entonces
 repeticionesDefinido \leftarrow repeticiones;
 moda \leftarrow indice;
 FinSi
FinPara

si espacio es par Entonces
 mediana \leftarrow (nums[((espacio)/2)-1]+(nums[(espacio/2)]))/2;
SiNo
 mediana \leftarrow nums[((espacio+1)/2)-1];
FinSi

media \leftarrow suma/espacio;
rango \leftarrow nums[espacio-1] - nums[0];

escribir "La media aritmetica es: ", media;
escribir "La mediana es: ", mediana;
escribir "La moda es: ", moda;
escribir "El rango es: ", rango;
FinAlgoritmo
```

- Foto de ejecución:

```
¿Cuántos numeros insertarás?
> 3
Mencioname uno por uno los valores
> 10
> 20
> 30
Los valores ordenados son:
10, 20, 30,
La media aritmetica es: 20
La mediana es: 20
La moda es: 10
El rango es: 20
```

- En este algoritmo va a la par con el anterior, porque necesitaba ordenar el vector y había utilizado el mismo patrón, aunque de igual manera me daba el resultado.

5. Este algoritmo suma dos vectores de la misma longitud e imprime uno nuevo.

- Entradas: Ninguna, son valores al azar de cada elemento de los vectores A y B que tienen una longitud de 10 cada uno
- Salida: Un nuevo vector con la suma de cada valor entre los vectores A Y B
- Foto de código:

**Algoritmo** actividadSumavectores

**definir** vectorA, vectorB, vectorC **como** entero;

**definir** indice, limite **Como** Entero;

limite  $\leftarrow$  9;

**dimensionar** vectorA(10), vectorB(10), vectorC(10);

**para** indice  $\leftarrow$  0 **hasta** limite **Hacer**

vectorA(indice)  $\leftarrow$  azar(100);

vectorB(indice)  $\leftarrow$  azar(100);

vectorC(indice)  $\leftarrow$  vectorA(indice) + vectorB(indice);

**FinPara**

**para** indice  $\leftarrow$  0 **hasta** limite **Hacer**

**escribir** vectorA(indice), " ", vectorB(indice), " ", vectorC(indice);

**FinPara**

**FinAlgoritmo**

- Foto de ejecución:

```
14 75 89
46 87 133
77 13 90
43 62 105
53 11 64
17 65 82
70 0 70
76 55 131
13 43 56
79 24 103
```

- Este lo adjunto porque forma parte de la introducción a los arreglos en nuestra unidad, y me había parecido interesante imprimir los valores así.

## Algoritmos con arreglos: Matrices

1. Se tiene un par de matrices cuadradas A y B de NxN. Escribir un algoritmo que permita intercambiar toda una fila de A por toda una columna de B, designadas por el usuario. Por ejemplo, si en las siguientes dos matrices el usuario pide intercambiar la fila 1 de A por la columna 3 de B, se tendría el siguiente resultado.

- entrada: filas, columnas como las dimensiones de matrizA y matrizB, después Seleccion1, Seleccion2 como la fila y la columna que quiere que se intercambien
- Salida: las matrices intercambiadas
- Foto de código:

```
definir filas, columnas, matrizA, matrizB, seleccion1, seleccion2, indice1, indice2, aux como entero;
filas ← 0;
columnas ← 0;
seleccion1 ← 0;
seleccion2 ← 0;
aux ← 0;

//inicializamos las matrices
escribir "Menciona el espacio cuadrado de la matriz";
leer filas;
columnas ← filas;

dimensionar matrizA[filas,columnas], matrizB[filas,columnas];

//pedimos valores a matrizA
escribir "Menciona uno por uno los valores de la matriz A";
para indice1←0 hasta filas-1 Hacer
 para indice2 ←0 hasta columnas-1 Hacer
 leer matrizA[indice1,indice2];
 FinPara
FinPara

//Pedimos valores a matrizB
escribir "Menciona uno por uno los valores de la matriz B";
para indice1←0 hasta filas-1 Hacer
 para indice2 ←0 hasta columnas-1 Hacer
 leer matrizB[indice1,indice2];
 FinPara
FinPara

//Imprimimos inicialmente las matrices
escribir "Las matrices son:";
escribir "A: ";
para indice1←0 hasta filas-1 Hacer
 para indice2 ←0 hasta columnas-1 Hacer
 escribir sin saltar matrizA[indice1,indice2] " ";
 FinPara
 escribir "\n";
FinPara
escribir "-----";
escribir "B: ";
para indice1←0 hasta filas-1 Hacer
 para indice2 ←0 hasta columnas-1 Hacer
```



```

para indice1←0 hasta filas-1 Hacer
 para indice2 ←0 hasta columnas-1 Hacer
 escribir sin saltar matrizB[indice1, indice2] " ";
 FinPara
 escribir " ";
FinPara

//seccion para seleccionar fila y columna que se cambiara
escribir "¿Cual fila de la matriz A seleccionas para intercambiar? 0-",filas-1;
leer seleccion1;
escribir "¿Cual columna de la matriz B seleccionas para intercambiar? 0-",columnas-1;
leer seleccion2;

para indice1 ← 0 hasta filas-1 Hacer
 aux ← matrizA[seleccion1,indice1];
 matrizA[seleccion1,indice1] ← matrizB[indice1,seleccion2];
 matrizB[indice1,seleccion2] ← aux;
FinPara

//-----
//imprimos matrices
//-----
escribir "Las matrices quedan:";
escribir "A: ";
para indice1←0 hasta filas-1 Hacer
 para indice2 ←0 hasta columnas-1 Hacer
 escribir sin saltar matrizA[indice1,indice2] " ";
 FinPara
 escribir "";
FinPara
escribir "-----";
escribir "B: ";
para indice1←0 hasta filas-1 Hacer
 para indice2 ←0 hasta columnas-1 Hacer
 escribir sin saltar matrizB[indice1, indice2] " ";
 FinPara
 escribir " ";
FinPara
FinAlgoritmo

```

- Foto de ejecución:

```

Las matrices son:
A:
3 1 2
3 4 5
2 3 4

B:
1 2 3
4 5 6
3 2 4
¿Cual fila de la matriz A seleccionas para intercambiar? 0-2
> 0
¿Cual columna de la matriz B seleccionas para intercambiar? 0-2
> 1
Las matrices quedan:
A:
2 5 2
3 4 5
2 3 4

B:
1 3 3
4 1 6
3 2 4

```

- Elegí este algoritmo porque me parecía muy atractivo de usar, sobre todo porque anteriormente ya lo había escuchado que lo mencione el profesor.

2. Hacer un algoritmo que llene una matriz 10x10 de enteros con números aleatorios del 0 al 99, luego de eso pida al usuario un número e imprima las posiciones en las que se encuentra ese número dentro de la matriz.

- entradas: num como el número que se busca las repeticiones dentro de la matriz
- salida: ubicación y repeticiones de ese número dentro de la matriz
- Foto de código:

```

definir matriz, num, ubicacion, repeticiones, indice1, indice2 Como Entero;

repeticiones ← 0;
ubicacion ← 0;
num ← 0;

dimensionar matriz[10,10];

para indice1 ← 0 hasta 9 Hacer
 para indice2 ← 0 hasta 9 Hacer
 matriz[indice1, indice2] ← aleatorio[0,99];
 FinPara
FinPara

escribir "La matriz es: ";
para indice1 ← 0 hasta 9 Hacer
 para indice2 ← 0 hasta 9 Hacer
 escribir sin saltar matriz[indice1, indice2] " ";
 FinPara
 escribir "\n";
FinPara

escribir "¿Que numero quieres averiguar sus repeticiones y lugares?";
leer num;

escribir "El numero se encuentra en las posiciones: ";
para indice1 ← 0 hasta 9 Hacer
 para indice2 ← 0 hasta 9 Hacer
 si num == matriz[indice1, indice2] Entonces
 repeticiones ← repeticiones + 1;
 escribir sin saltar "(" , indice1 " , " , indice2 ") ";
 FinSi
 FinPara
FinPara
escribir "\n";

escribir "La cantidad de repeticiones es de: ", repeticiones;
inAlgoritmo

```

- Foto de ejecución:

```

La matriz es:
37 46 65 66 23 19 81 86 50 59
64 60 93 19 67 1 80 85 9 33
64 6 90 39 11 20 30 49 28 23
25 92 51 6 34 60 61 28 41 23
87 80 16 34 63 61 25 9 58 46
10 64 75 41 93 81 65 27 26 89
13 96 22 65 6 78 69 26 33 51
32 36 74 70 87 54 64 51 8 87
85 36 25 94 65 42 6 12 97 96
27 98 13 44 33 65 71 44 14 11
¿Que numero quieres averiguar sus repeticiones y lugares?
> 10
El numero se encuentra en las posiciones:
(5,0)
La cantidad de repeticiones es de: 1
*** Ejecución Finalizada. ***

```

- Elegí este algoritmo porque me gustaba el hecho de generar los números aleatorios y pensar en la probabilidad que se repitan.

3. Hacer un algoritmo que llene una matriz de 5x3 reales y que calcule la desviación estándar de ese conjunto de valores.

- entrada: rango de los valores de la matriz
- salida: la desviación estándar de la matriz 5x3
- foto de código:

```
definir indice1, indice2, indice3 Como Entero;
definir desviacion, matriz, min, max, vector, suma, promedio Como Real;
```

```
min ← 0.0;
max ← 0.0;
promedio ← 0.0;
suma ← 0.0;
desviacion ← 0.0;
indice3 ← 0;
```

```
escribir "Cual es el valor minimo que puede aparecer";
leer min;
escribir "Ahora maximo";
leer max;
```

```
Dimensionar matriz[5,3];
Dimensionar vector[15];
para indice1 ← 0 hasta 4 Hacer
 para indice2 ← 0 hasta 2 Hacer
 matriz[indice1, indice2] ← aleatorio(min,max);
 suma ← suma + matriz[indice1, indice2];

 //Hacemos un vector para aligerar
 vector[indice3] ← matriz[indice1, indice2];
 indice3 ← indice3 + 1;
 FinPara
FinPara
```

```
promedio ← suma/15;
```

```
//hallamos la suma de la resta de cada valor del vector hecho a base de la i
suma ← 0.0;
```

```
para indice1 ← 0 hasta 14 Hacer
```

```
 suma ← suma + (vector[indice1]-promedio)2;
```

```
FinPara
```

```
desviacion ← raiz((suma)/15);
```

```
//imprimos la matriz
```

```
para indice1 ← 0 hasta 4 Hacer
```

```
 para indice2 ← 0 hasta 2 Hacer
```

```
 escribir sin saltar matriz[indice1, indice2] " ";
```

```
 FinPara
```

```
 escribir " ";
```

```
FinPara
```

```
escribir "La desviacion estandar de todo esos numeros es de: ", desviacion;
```

**FinAlgoritmo**

- Foto de ejecución:

Cual es el valor minimo que puede aparecer

> 0

Ahora maximo

> 10

10 4 3

0 9 2

1 7 8

10 0 0

9 5 0

La desviacion estandar de todo esos numeros es de: 3.8447655614

\*\*\* Ejecución Finalizada. \*\*\*

- Elegí poner acá este algoritmo porque se me había ocurrido algo super interesante para hacer, es volver las matrices en vectores, eso te agiliza un montón las funciones para poder trabajar y cuando lo implementé, fue mucho más fácil de hacer. Sobre todo porque en ese momento ya había aprendido un método para poder ordenar

4. Dada una matriz cuadrada A, construya un algoritmo que permita determinar si dicha matriz es simétrica. Se considera que una matriz es simétrica si  $A[i,j] = A[j,i]$  y esto se cumple para todos los elementos i, j de la matriz.

- entradas: amplitud como el cuadrado de los espacios que habran en la matriz, num como el valor a insertar
- salida: si es simetrica la matriz tanto en matriz[i,j] sea igual que matriz[j,i], tanto en posiciones
- Foto de codigo;

**Algoritmo Simetria**

*//entradas: amplitud como el cuadrado de los espacios que habran en la matriz, num como el valor a insertar  
//salida: si es simetrica la matriz tanto en matriz[i,j] sea igual que matriz[j,i], tanto en posiciones*

```

definir amplitud, matriz, num, indice1, indice2 Como Entero;
definir simetrico Como Logico;

amplitud \leftarrow 0;
simetrico \leftarrow verdadero;
num \leftarrow 0;

escribir "Mencioname de que espacio al cuadrado quieres que sea tu matriz";
leer amplitud;

Dimensionar matriz[amplitud,amplitud];

escribir "Mencioname uno por uno los valores de la matriz";
para indice1 \leftarrow 0 hasta amplitud - 1 Hacer
 para indice2 \leftarrow 0 hasta amplitud-1 Hacer
 leer matriz[indice1,indice2];
 FinPara
FinPara

para indice1 \leftarrow 0 hasta amplitud - 1 Hacer
 para indice2 \leftarrow 0 hasta amplitud-1 Hacer
 si matriz[indice1,indice2] == matriz[indice2, indice1] y simetrico==verdadero Entonces
 simetrico \leftarrow Verdadero;
 SiNo
 simetrico \leftarrow falso;
 FinSi
 FinPara
FinPara

```

```

para indice1←0 hasta amplitud - 1 Hacer
 para indice2←0 hasta amplitud-1 Hacer
 escribir sin saltar matriz[indice1,indice2] " ";
 FinPara
 escribir " ";
FinPara
escribir "-----";
para indice1←0 hasta amplitud - 1 Hacer
 para indice2←0 hasta amplitud-1 Hacer
 escribir sin saltar matriz[indice2,indice1] " ";
 FinPara
 escribir " ";
FinPara

si simetrico == Verdadero Entonces
 escribir "Si cumple simetria";
SiNo
 escribir "No cumple simetría";
FinSi

```

#### FinAlgoritmo

- Foto de ejecución:  
Mencioname de que espacio al cuadrado quieres que sea tu matriz:

> 2

Mencioname uno por uno los valores de la matriz

> 1

> 0

> 0

> 1

1 0

0 1

-----

1 0

0 1

Si cumple simetria

- Este lo elijo porque me aprendí como identificar las matrices simétricas y sobre todo porque se veía muy bonita la salida.

5. Escriba un algoritmo que genere una matriz de 3x4 de valores enteros que la imprima y después que la ordene de mayor a menor. El mayor estará en la posición [0,0] y el menor en la posición [2,3].

- entradas: nums como los valores a insertarse en la matriz 3x4 llamado matriz
- salida: la misma matriz, pero ordenada
- Foto de código:

```
definir nums, matriz, indice1, indice2, indice3, indice4, esMenor1, esMenor2, aux Como Entero;
```

```
nums ← 0;
```

```
aux ← 0;
```

```
esMenor1 ← 0;
```

```
esMenor2 ← 0;
```

```
Dimensionar matriz[3,4];
```

```
escribir "Mencioname los valores de la matriz";
```

```
para indice1 ← 0 hasta 2 Hacer
```

```
 para indice2 ← 0 hasta 3 Hacer
```

```
 leer nums;
```

```
 matriz[indice1, indice2] ← nums;
```

```
 FinPara
```

```
FinPara
```

```
escribir "La matriz es: ";
```

```
para indice1 ← 0 hasta 2 Hacer
```

```
 para indice2 ← 0 hasta 3 Hacer
```

```
 escribir sin saltar matriz[indice1, indice2] " ";
```

```
 FinPara
```

```
 escribir " ";
```

```
FinPara
```

```
//ordenamiento por seleccion
```

```
para indice1 ← 0 hasta 2 Hacer
```

```
 para indice2 ← 0 hasta 3 Hacer
```

```
 esMenor1 ← indice1;
```

```
 esMenor2 ← indice2;
```

```
 para indice3 ← indice1 hasta 2 Hacer
```

```
 para indice4 ← indice2 hasta 3 Hacer
```

```
 si matriz[esMenor1, esMenor2] > matriz[indice3, indice4] Entonces
```

```
 esMenor1 ← indice3;
```

```
 esMenor2 ← indice4;
```

```
 FinSi
```

```
 FinPara
```

```
 FinPara
```

```

escribir "-----";
escribir "La matriz queda: ";
para indice1 ← 0 hasta 2 Hacer
 para indice2 ← 0 hasta 3 Hacer
 escribir sin saltar matriz[indice1, indice2] " ";
 FinPara
 escribir "\n";
FinPara
FinAlgoritmo

```

- Foto de ejecución:

Mencioname los valores de la matriz

```

> 3
> 2
> 1
> 3
> 4
> 5
> 1
> 2
> 3
> 4
> 5
> 3

```

La matriz es:

```

3 2 1 3
4 5 1 2
3 4 5 3

```

-----

La matriz queda:

```

1 1 2 2
3 3 3 4
3 4 5 5

```

- Este es el último de los 25, sinceramente este se me hizo difícil, porque estaba pensando ordenarlo sin utilizar matrices y con el método de selección, y sí me funcionó, es increíble.