

¡Empecemos programando con unos ejercicios! El primer programa que haremos recoge los datos de un usuario, y hace unas operaciones básicas sobre ellos.

1. Pidamos el nombre del usuario, y despleguemos un mensaje de bienvenida con su nombre.

```
1  escribir("Ingrese su nombre por favor:")
2  tex nombre = leer()
3  escribir("Hola " + nombre + "!")
```

2. Pidamos datos numéricos básicos como la edad, el peso y la estatura.

```
4  escribir("Ingrese su edad:")
5  ent edad = convertirEnt(leer())
6  escribir("Ingrese su peso en kilogramos:")
7  dec peso = convertirDec(leer())
8  escribir("Ingrese su estatura en centímetros:")
9  dec estatura = convertirDec(leer())
```

3. Preguntemos si el usuario es hombre o mujer. El programa debe pedir este dato hasta que se haya ingresado de la forma correcta. Se debe guardar el resultado en un variable de tipo lógico (booleano).

```
11 hacer:
12     escribir("Ingrese M si es mujer, o H si es hombre:")
13     género = convertirCar(leer())
14     log esMujer
15     si (género == 'M' | género == 'm'):
16         esMujer = verdad
17     fin
18     sinosi(género == 'H' | género == 'h'):
19         esMujer = falso
20     fin
21     sino:
22         escribir("Opción no reconocida")
23     fin
24 fin mientras (género != 'M' & género != 'm' & género != 'H' & género != 'h')
```

4. Indiquemos al usuario su estado sobre el sufragio.

```
26 si (edad < 16):
27     escribir("No tiene edad suficiente para sufragar")
28 fin
29 sinosi(edad < 18):
30     escribir("En su caso, el voto es opcional")
31 fin
32 sino:
33     escribir("El sufragio es obligatorio en su caso")
34 fin
```

5. Indiquemos al usuario su índice de masa corporal (peso en kg / (estatura en metros)^2).

```
36 escribir("Su índice de masa corporal es:")
37 escribir(peso / ((estatura / 100) ^ 2))
```

6. Escribamos el nombre del usuario al revés. (Es posible que el editor muestre un error semántico, porque no reconoce al variable de tipo texto como un arreglo, pero el código sí ejecutará)

```
39 para(ent i = longitud(nombre) - 1; i >= 0; i--):  
40     escribir(nombre[i])  
41 fin
```

7. Cerremos el programa con un comando de leer (para detener la ejecución del programa hasta leer los resultados) y fin (para indicar la finalización el programa).

```
43 leer()  
44 fin
```

Si redactamos el código y ejecutamos el programa, podríamos tener una salida de consola así:

```
Ingrese su nombre por favor:  
James  
Hola James!  
Ingrese su edad:  
26  
Ingrese su peso en kilogramos:  
65.9  
Ingrese su estatura en centímetros:  
172.72  
Ingrese M si es mujer, o H si es hombre:  
H  
El sufragio es obligatorio en su caso  
Su indice de masa corporal es:  
22.0902258413513  
s  
e  
m  
a  
J
```

En este sencillo ejercicio, se ha visto muchos fundamentos de la programación, como el manejo de consola (escribir y leer texto), diferentes tipos de datos (texto, entero, decimal, carácter y lógico), conversiones de tipo texto a numérico, bucles (hacer-mientras y para), estructuras de control (si-sinosisi-sino), operadores relacionales (<, ==, != y >=), operadores lógicos (|), cambios unitarios (i--), acceso a arreglos (nombre[i]), etc.

Ejercicio para resolver: Pida al usuario la longitud de tres lados de triángulo. Haga lo siguiente: determine la suma de los lados, el promedio de su longitud y si el triángulo es escaleno, isósceles o equilátero. Use un bucle (el bucle de su preferencia) para recopilar los datos del triángulo.

Para el siguiente ejercicio, vamos a usar arreglos y funciones. Crearemos un programa donde se puede ingresar notas de un estudiante, escribirlos a un archivo de texto y determinar si el estudiante pasa (o no).

1. Creemos una función de tipo "nada" (no tiene retorno) que indique si el usuario pasa o no, de nombre "nada".

```
1 función nada pasa(dec promedio):  
2     si (promedio >= 7):  
3         escribir("Sí pasa")  
4     fin  
5     sino si (promedio >= 5):  
6         escribir("No pasa pero sí llega a supletorio")  
7     fin  
8     sino:  
9         escribir("No pasa ni llega a supletorio")  
10    fin  
11    leer()  
12    fin
```

2. Pidamos la cantidad de notas que el usuario va a ingresar. Inicialicemos un arreglo que aloje las notas y variables para suma y promedio, y un dato textual que usaremos para guardar la información.

```
14 escribir("Ingrese la cantidad de notas a calcular:")  
15 ent cantidadNotas = convertirEnt(leer())  
16 dec notas[cantidadNotas]  
17 dec suma = 0, promedio = 0  
18 tex notasTexto = ""
```

3. Escribamos un bucle que recorra las notas: va a pedir la nota, guardarla en el arreglo, incrementar la suma y agregar el valor a la variable de texto.

```
19 para(ent i = 0; i < cantidadNotas; i++):  
20     escribir("Ingrese la nota #" + (i + 1) + ":")  
21     notas[i] = convertirDec(leer())  
22     suma = suma + notas[i]  
23     notasTexto = notasTexto + notas[i] + "\n"  
24 fin
```

4. Calculemos el promedio y mostremos la suma y el promedio en la pantalla.

```
25 promedio = suma / cantidadNotas  
26 escribir("Suma: " + suma)  
27 escribir("Promedio: " + promedio)
```

5. Invoquemos la función "pasa". Escribamos un archivo de texto que contenga las notas. Que el archivo se titule "misNotas.txt". Finalicemos el código con un "fin".

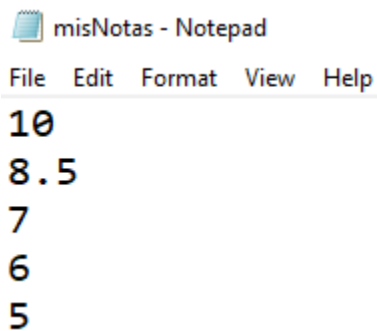
```
28 pasa(promedio)  
29 escribirtexto(notasTexto, "misNotas.txt")  
30
```

6. 31 fin

Si ejecutamos el código, podríamos tener el siguiente ejemplo de salida:

```
Ingrese la cantidad de notas a calcular:
5
Ingrese la nota #1:
10
Ingrese la nota #2:
8.5
Ingrese la nota #3:
7
Ingrese la nota #4:
6
Ingrese la nota #5:
5
Suma: 36.5
Promedio: 7.3
Si pasa
```

Se puede visualizar también las notas en un archivo de texto:



misNotas - Notepad

File Edit Format View Help

10
8.5
7
6
5

Ejercicio para resolver: ¡Cree un programa que permite componer música! Debe crear dos arreglos (una para frecuencia y otra para duración) de tipo entero, o un arreglo bidimensional. Debe ingresar los datos requeridos mediante la consola (o incluso podría usar números aleatorios con azar(mínimo, máximo)) . Debe tener un mínimo de quince notas. Puede tratar de componer su propia canción, o basarse en una ya existente. Para reproducir la canción, se debe usar un bucle (el bucle de su preferencia) y la función sonido(frecuencia, duración).

Para este último ejercicio práctico, vamos a usar la ventana gráfica. Vamos a dibujar un carro (se puede usar cualquier otro gráfico .png) que se va a ir moviendo por la pantalla. Se deja a su libertad y creatividad sobre el movimiento, pero aquí se presentará un ejemplo. (Puede usar “car1.png” para este ejercicio – el gráfico viene incluido)

1. Creamos una función de movimiento.

```
1  funcion ent sumarVelocidad(ent coordenada, ent direccion):
2      si (direccion == 1 | direccion == 2):
3          retornar(coordenada + azar(1, 35))
4      fin
5      sino:
6          retornar(coordenada - azar(10, 15))
7      fin
8  fin
```

2. Inicializamos las variables necesarias, recordando que es imprescindible declarar variables sin asignación para que se las pueda usar en el espacio “dibujar”.

```
10  ras auto
11  ent x, y, min, max, direccion
12  auto = leerras("car1.png")
13  x = 25
14  y = 25
15  min = 25
16  max = 400
17  direccion = 1
18  fin
```

3. Escribimos el bloque dibujar, que tenga las condiciones de movimiento y que dibuje el carro en la pantalla.

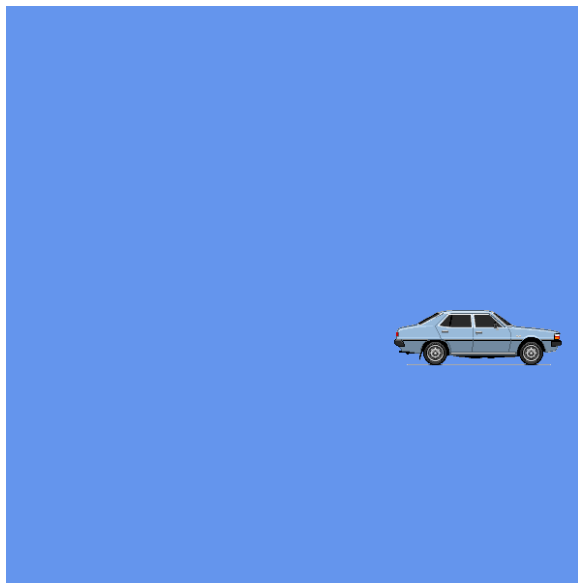
```
21  dibujar:
22      dibujarras(auto, x, y, blanco)
23      si (direccion == 1):
24          x = sumarVelocidad(x, direccion)
25          si(x >= max):
26              direccion = 2
27          fin
28      fin
29      sinosi(direccion == 2):
30          y++
31          y = sumarVelocidad(y, direccion)
32          si(y >= max):
33              direccion = 3
34          fin
35      fin
```

```

36  sinosi(direccion == 3):
37      x--
38      x = sumarVelocidad(x, direccion)
39      si (x <= min):
40          direccion = 4
41      fin
42  fin
43  sino:
44      y--
45      y = sumarVelocidad(y, direccion)
46      si(y <= min):
47          direccion = 1
48      fin
49  fin
50  fin

```

Siguiendo este ejemplo, se puede visualizar la imagen de un automóvil moviendo por la pantalla en líneas rectas, a una velocidad variable.



Ejercicio a resolver: Dibuje un gráfico en la pantalla que pueda moverse de un lado a otro, pero que también se cambie la gráfica usada dependiendo de alguna condición. Por ejemplo, en el caso del carro, que cambie la gráfica dependiendo de la dirección. (Puede usar “car1.png”, “car2.png”, “car3.png” y “car4.png” para este ejercicio, u otro archivo gráfico si desea)

