**Estructura básica de un programa en Pascal**

program nombre\_ejemplo;

var

nombre\_variable: tipo; //puede ser cualquier tipo de dato

begin

// instrucciones del programa principal

end.

program → palabra clave que indica el nombre del programa.

var → sección para declarar variables.

begin end. → sección que delimita el programa principal. Es el único “end” que finaliza con punto.

begin end; → sección que delimita un conjunto de instrucciones.

**Tipos de Datos**

1. **¿Qué tipos de datos existen?**
   1. **boolean** → verdadero/falso
   2. **integer** → números enteros
   3. **real** → números con coma
   4. **char** → carácter (una letra o símbolo)
2. **¿Dónde se utilizan?**
   1. En la declaración de las variables en la sección **VAR**.

**program ejemplo;**

**var**

**num: integer;**

**letra: char;**

**fin: boolean;**

**precio: real;**

**Lectura de datos**

1. **¿Cuándo se utiliza?**
   1. La lectura de datos se realiza cuando se requiere que el usuario *ingrese datos* desde teclado.
2. **¿Cómo se utiliza?**
   1. read(variable); → no tiene en cuenta el salto de línea
   2. readln(variable); → tiene en cuenta el salto de línea
3. **¿Cómo funciona?**
   1. Cuando se ejecute la instrucción “read” el programa habilitará al usuario a ingresar un dato desde la consola.
   2. El dato se guardará en la variable que está dentro del read.

Tener en cuenta que el dato que se ingresa debe corresponder al tipo de dato de la variable

**Escritura de datos**

1. **¿Cuándo se utiliza?**
   1. La escritura de datos o de texto se utiliza cuando se quiere *informar* algo en pantalla.
2. **¿Cómo se utiliza?**
   1. write(variable); → imprime el contenido de la variable. no tiene en cuenta el salto de línea
   2. writeln(variable); → imprime el contenido de la variable. tiene en cuenta el salto de línea
   3. write(’texto’); → imprime el texto entre comillas. **No pone un “enter”** al final. El próximo texto que se imprima, lo hará en la **misma** línea
   4. writeln(’texto); → imprime el texto entre comillas. **Pone un “enter”** al final. El próximo texto que se imprima, lo hará en la **siguiente** línea
3. **¿Cómo funciona?**
   1. Cuando se ejecute la instrucción “read” el programa habilitará al usuario a ingresar un dato desde la consola.
   2. El dato se guardará en la variable que está dentro del read.

Se puede combinar el imprimir texto con el valor de una variable: **writeln(’Ganó ’, variable\_numero, ‘ pesos’)** → *“Ganó 500 pesos”*

**Operaciones**

* suma → +
* resta → -
* multiplicación → \*
* división → /
* división entera → DIV
* resto de división entera → MOD

**División Entera**

* Se utiliza la instrucción **DIV** o **MOD** según se necesite.
* **DIV** → se queda con el resultado de la división entera entre dos números.
* **MOD** → se queda con el resto de una división entera entre dos números.

**EJEMPLO**

* 15 div 2 = 7
* 15 mod 2 = 1
* 14 div 2 = 7
* 14 mod 2 = 0
* num1 = 10;
* num2 = 5;
* num1 div num2 = 2
* num1 mod num2 = 0

**Números pares e impares**

Un número es **PAR** cuando al aplicarle **MOD 2** el resultado **es igual a 0.**

Un número es **IMPAR** cuando al aplicarle **MOD 2** el resultado **es igual a 1.**

**EJEMPLO**

* 15 mod 2 = 1 -> IMPAR
* 16 mod 2 = 0 -> PAR

num1 = 10;  
num2 = 5;

* num1 mod 2 = 0 -> num1 es par
* num2 mod 2 = 1 -> num2 es impar

**Estructuras de Control**

**IF**

1. **¿Cuándo se utiliza?**
   1. Cuando se desea chequear una condición una sola vez.
2. **¿Cuántas veces se puede ejecutar?**
   1. 0 o 1 vez
3. **¿Cuándo se ejecuta?**
   1. Las instrucciones del IF \*\*\*\*se ejecutan cuando la **condición** da **VERDADERO**
4. **ELSE**
   1. Las instrucciones del ELSE se van a ejecutar si la instrucción del IF da **FALSO**

**EJEMPLO**

program ejemplo\_if;

var

num: integer;

begin

read(num); //leo un numero

if (num < 10) then

write('El numero es menor a 10');

else

write('El numero es mayor a 10');

end.

**FOR**

1. **¿Cuándo se utiliza?**
   1. Cuando se sabe **EXACTAMENTE** la cantidad de veces que se desean ejecutar ciertas instrucciones
2. **¿Cuántas veces se puede ejecutar?**
   1. 0 o X veces
3. **¿Cuándo se ejecuta?**
   1. Las instrucciones del for se van a ejecutar hasta que el índice llegue al valor final (**TO**)
   2. Las instrucciones del for se van a ejecutar hasta que el índice llegue al valor inicial (**DOWNTO**)
4. **TO**
   1. Cuando se usa este formato, el índice se va autoincrementando.
5. **DOWNTO**
   1. Cuando se usa este formato, el índice en vez de autoincrementarse, va disminuyendo de a uno.

Tener en cuenta que la variable **i** (o cualquiera sea su nombre), no debe ser modificada y generalmente se utiliza únicamente dentro del bucle.

**EJEMPLO**

program ejemplo\_for;

var

num: integer;

i: integer;

begin

for (i:=1 to 10) do begin // va a leer e imprimir 10 numeros

read(num);

write(num);

end;

end.

//------------------------------------------------------------ //

program ejemplo\_for;

var

indice: integer;

begin

for (indice:=1 to 10) do begin // va a imprimir numeros del 1 al 10

write(indice)

end;

end.

//------------------------------------------------------------ //

program ejemplo\_for;

var

indice: integer;

begin

for (indice:=10 downto 1) do begin // va a imprimir numeros del 10 al 1

write(indice)

end;

end.

**WHILE**

1. **¿Cuándo se utiliza?**
   1. Cuando no se sabe la cantidad de veces que se procesan los datos
   2. Cuando el enunciado dice “**no se procesa”.**
2. **¿Cuántas veces se puede ejecutar?**
   1. 0, 1 o más veces
3. **¿Cuándo se ejecuta?**
   1. Las instrucciones del **WHILE** se ejecutan mientras la **condición** da **VERDADERO**
   2. Cada vez que se ejecutan las instrucciones se vuelve a chequear la condición

Tener en cuenta que dentro del while deberá haber algo que cambie en algún momento el estado de la condición para evitar un **bucle infinito**.

**EJEMPLO**

program ejemplo\_while;

var

num: integer;

begin

read(num); //leo un numero

while (num < 10) do begin //mientras que el numero que leo sea menor a 10, sigo leyendo

writeln('Ingrese otro numero');

read(num);

end;

writeln('El numero es mayor o igual a 10');

end.

**REPEAT UNTIL**

1. **¿Cuándo se utiliza?**
   1. Cuando no se sabe la cantidad de veces que se procesan los datos
   2. Cuando el enunciado dice “**se procesa”.**
2. **¿Cuántas veces se puede ejecutar?**
   1. 1 o muchas veces
3. **¿Cuándo se ejecuta?**
   1. Las instrucciones del **REPEAT UNTIL** se ejecutan **HASTA QUE** la condición sea **VERDADERA**
   2. Cada vez que se ejecutan las instrucciones se vuelve a chequear la condición

Tener en cuenta que dentro del repeat until deberá haber algo que cambie en algún momento el estado de la condición para evitar un **bucle infinito**.

**EJEMPLO**

program ejemplo\_repeat\_until;

var

num: integer;

begin

repeat // lee numeros hasta que aparezca uno menor a 10

writeln('Ingrese numero');

read(num);

until (num < 10);

writeln('El numero es menor o igual a 10');

end.