****

***Laboratorio #3***

***Estructuras de repetición***

**Objetivos:**

Conocer el funcionamiento de las estructuras de repetición.

Uso de las distintas estructuras de repetición.

**Pre laboratorio**

**Estructuras de repetición**

Se conocen también como bucles o lazos (loop).Se utilizan para repetir una secuencia de instrucciones un determinado número de veces. Este número de veces puede ser conocido de antemano o determinado por una condición lógica, dependiendo de esto, pueden ser:

1. **Repita mientras (while)**

2. **Repita hasta (repeat)**

3. **Repita para (for)**

Las estructuras de repetición están conformadas por:

**El cuerpo del bucle:** que contiene las sentencias que se repiten. Cada repetición del cuerpo del bucle se denomina iteración.

**La condición lógica:** que determinará el número de repeticiones a realizar.

**1.- La Sentencia WHILE (Repita mientras)**

 

Es aquella en la que el número de iteraciones no se conoce por anticipado y el cuerpo del bucle se repite mientras se cumple una determinada condición.

**Funcionamiento:**

1. Cuando la sentencia while se ejecuta, la primera cosa que sucede es la evaluación de la expresión lógica. Si resulta falso no se realiza ninguna acción y el programa prosigue en la siguiente sentencia después del bucle.

2. Si la expresión lógica se evalúa y resulta verdadero entonces se ejecuta el cuerpo del bucle y se evalúa nuevamente la expresión.

3. Mientras la condición sea verdadera el cuerpo del bucle se ejecutará, por lo que debe haber algo en el cuerpo del bucle que modifique la condición de verdadera a falsa. Si la condición nunca cambia de valor se denomina bucle infinito.

4. La condición lógica se evalúa antes y después de la ejecución del bucle.

5. Si al evaluar la condición la primera vez, ésta es falsa, el cuerpo del bucle nunca se ejecutará.

**Variables no Inicializadas**

1. La condición lógica en bucle debe tener un valor la primera vez que se evalúa; en caso contrario dará error.

2. Esta inicialización se hará con las sentencias de lectura o asignación.

**En PASCAL, sentencia simple:**

**while** expresión lógica **do**

sentencia1;

**Sentencia compuesta.**

**while** expresión lógica **do**

**begin**

sentencia1;

…

sentencia n;

**end;**

**Ejemplo de un programa en Pascal para sumar números enteros.(usando While)**

**Program** sumar\_numeros;

**uses** crt;

**Var**

n, contador, numero, suma: integer;

**Begin**

Writeln(' Este programa suma numeros');

Writeln('Cuantos numeros desea sumar?');

Readln(n);

Writeln('Por favor introduzca los ', n, ' numeros ');

Contador:=0;

Suma:=0;

**While** contador < n **do**

**begin**

Read(numero);

Suma:=suma+numero;

Contador:=contador+1;

**End;**

Writeln('La sumatoria de los numeros es ', suma);

Readkey;

**End.**

**2.- La Sentencia REPEAT (Repita Hasta)**

Una variante de la sentencia while es la sentencia repeat. La sentencia repeat especifica un bucle condicional que se repite hasta que la condición se hace verdadera, tal bucle se denomina bucle repeat-until.



**En Pascal, el repita hasta se hace de las siguiente forma :**

**Repeat**

Sentencia 1;

Sentencia 2;

. .

Sentencia n;

**Until** expresión lógica

**Funcionamiento:**

1. La condición se evalúa al final del bucle, después de ejecutarse todas las sentencias.

2. Si la condición es verdadera el bucle se termina y se sale de él, ejecutándose la siguiente sentencia.

3. Si la condición es falsa, el cuerpo del bucle se repite y se ejecutarán todas las sentencias.

4. La sintaxis no requiere begin y end;

**Ejemplo de un programa en Pascal para sumar números enteros (usando Repeat).**

**Program** sumar;

**uses** crt;

**Var**

total, contador, numero, suma: integer;

**Begin**

Writeln(' Este programa suma numeros');

Writeln('Cuantos numeros desea sumar?');

Readln(total);

Writeln('Introduzca los ’ ,total, ‘ numeros');

Contador:=0;

Suma:=0;

**Repeat**

Read(numero);

Suma:=suma+numero;

Contador:=contador+1;

**Until** contador=total;

Writeln('La sumatoria de los numeros es ', suma);

Readkey;

**End.**

**3.- La Sentencia FOR**

**3.1 Incrementos de contador de for (TO)**

En algunas ocasiones es posible que un bucle se ejecute un número determinado de veces y cuyo número se conozca por anticipado. Para aplicaciones de este tipo se utiliza la sentencia for, para ellos existe una variable llamada de control que es la que controla el número de repeticiones.

**Sentencia simple**

**For** variable:=valor\_inicial **to** valor:final **do**

Sentencia 1;

**Sentencia compuesta**

**For** variable:=valor\_inicial **to** valor:final **do**

**begin**

Sentencia 1;

Sentencia 2;

.

.

Sentencian;

**End;**

**Funcionamiento:**

1. Las variables de control, valor inicial y valor final deben ser todas del mismo tipo, el tipo real no está permitido. Los valores iniciales pueden ser tanto expresiones como constantes.

2. Antes de la primera ejecución del bucle a la variable de control se asigna el valor inicial.

3. La última ejecución del bucle ocurre cuando la variable de control es igual al valor final.

4. Es ilegal modificar el valor de la variable de control, el valor inicial y el valor final dentro del bucle.

**Ejemplo de un programa en Pascal para sumar números enteros (usando FOR).**

**Program** ciclo\_for;

**uses** crt;

**Var**

Max, i, numero,suma:integer;

**Begin**

Writeln(' Este programa suma numeros');

Writeln('Cuantos numeros desea sumar?');

Readln(max);

Writeln('Introduzca los numeros');

**For** i:=1 **to** max **do**

**Begin**

Read(numero);

Suma:=suma+numero;

**End;**

Writeln('La sumatoria de los numeros es ', suma);

Readkey;

**End.**

**3.2 Decremento de contador de for (DOWNTO)**

El contador del bucle se puede decrementar de uno en uno en lugar de incrementar como en el caso anterior.

**Funcionamiento:**

1. Las variables de control, valor inicial y valor final deben ser todas del mismo tipo, el tipo real no está permitido. Los valores iniciales pueden ser tanto expresiones como constantes.

2. Antes de la primera ejecución del bucle a la variable de control se asigna el valor inicial.

3. La última ejecución del bucle ocurre cuando la variable de control es igual al valor final.

4. Cuando se utiliza la palabra reservada **to** la variable de control se incrementó en casa iteración, si se utiliza **downto,** la variable se decrementa.

5. Es ilegal modificar el valor de la variable de control, el valor inicial y el valor final dentro del bucle.

**Sentencia simple**

**For** variable:=valor\_inicial **downto** valor:final **do**

Sentencia 1;

**Sentencia compuesta**

**For** variable:=valor\_inicial **downto** valor:final **do**

**begin**

Sentencia 1;

Sentencia 2;

.

.

Sentencia n;

**End;**

**Ejemplo de un programa en Pascal para mostrar números en forma descendente a partir de un número dado (usando FOR).**

**Program** for\_decremento;

**uses** crt;

**Var**

Max,i:integer;

**Begin**

Writeln(' Este programa muestra los números de forma descendente');

Writeln('Introduzca n');

readln(max);

Writeln('Numeros del ', max, ' al 1');

**For** i:=max **downto** 1 do

Writeln(i);

Readkey;

**End.**

**4.- Bucles ANIDADOS**

El cuerpo de un bucle puede contener cualquier tipo de sentencias (simples o compuestas), selectivas (if-then-else-case) o repetitivas (for, while, repeat).

Cuando un bucle está contenido en otro bucle, se denomina bucle anidado.

Cuando se anidan bucles, se debe tener la seguridad que el bucle interior esté completamente dentro del bucle exterior.

**Recuerde elaborar los algoritmos de los problemas planteados para la actividad de laboratorio, antes del día de la práctica.**

**Laboratorio**

1. Construya un programa que calcule y muestre el promedio de una secuencia de números reales leída por el teclado, hasta que se introduzca el cero “0”.
2. Dados la categoría y el sueldo de N trabajadores, realice un programa, que calcule el aumento correspondiente teniendo en cuenta la siguiente tabla dada. Debe imprimir la categoría y el nuevo sueldo del trabajador.

|  |  |
| --- | --- |
| **Categoría** | **% aumento** |
| Instructor | 40 |
| Asistente | 35 |
| Agregado | 25 |
| Asociado | 20 |
| Titular | 15 |

1. Escribir un programa que dado el grado de una matriz cuadrada, generar la secuencia de sus celdas. Ejemplo. Si introduzco el grado 4, me debe generar por pantalla la secuencia:

|  |  |  |  |
| --- | --- | --- | --- |
| 1,1 | 1,2 | 1,3 | 1,4 |
| 2,1 | 2,2 | 2,3 | 2,4 |
| 3,1 | 3,2 | 3,3 | 3,4 |
| 4,1 | 4,2 | 4,3 | 4,4 |

**Post Laboratorio**

**Valor de cada ejercicio ( 5 ptos c/u)**

1.- Dada una secuencia de enteros positivos terminada en cero, elabore un algoritmo que :

b) Calcule el porcentaje de números impares y el porcentaje de números pares.

c) Calcule la cantidad de valores iguales a un valor N dado por el usuario.

d) El número mayor y el menor de la secuencia.

e) Calcular la cantidad de números primos.

2.-Construya un algoritmo que calculen el factorial de un número entero positivo n. La función factorial, representada por n!, es ampliamente utilizada, y se especifica que n!= n\*(n-1)!, y además 0!=1. Implemente el mismo programa utilizando repita mientras, repita hasta y repita para.

3.- La expresión ex para un número x cualquiera se puede estimar haciendo uso de la siguiente serie:

ex = 1 + x +

La cantidad de términos a sumar depende de la precisión que se desee, que se especifica cuando el valor del ultimo termino generado sea menor que un valor cualquiera. Escriba un algoritmo que permita leer un valor x y la precisión deseada.

4.-Elabore un algoritmo que determine si un número dado por pantalla es automórfico. Un número automórfico es aquel que al elevarlo sistemáticamente a una potencia mayor que cero, las últimas cifras del resultado corresponden al mismo número. Hacer el programa, para probar que un número es automórfico con un máximo de una potencia N dada. Ejemplo: 76 es automorfico para una potencia N= 4 porque: 761= 76; 762 = 5776; 763 = 438976; 764 = 33362176. Fíjese que al elevar el 76 a las 4 potencias sucesivas su resultado teminan en 76.

**Instrucciones para el envío de la tarea** :

Crear una carpeta comprimida que incluya:

* Un archivo .doc con los algoritmos de los problemas propuesto en el Postlaboratorio
* Los archivos .pas de los problemas resuelto (un archivo .pas por cada programa)
* Un archivo .doc con las capturas de pantalla de las corridas de los programas.

Además debe enviar la HOJA DE CONTROL DE PRACTICAS con la autoevaluación de la práctica de la semana.

Adjuntar archivos a la tarea correspondiente en Módulo 7