CADP 2022
Práctica 5 – Punteros

PARTE CONCEPTUAL

1) ¿Qué se define como memoria estática?
2) ¿Qué se define como memoria dinámica?
3) ¿Qué es una variable de tipo puntero?
4) ¿Qué hace la operación "NEW" aplicada en una variable del tipo puntero?
5) ¿Qué hace la operación "DISPOSE" aplicada en una variable del tipo puntero?

PARTE PRÁCTICA

Para algunos ejercicios de la parte práctica, se utilizará la función de Pascal "**sizeof"**, que recibe como parámetro una variable de cualquier tipo y retorna la cantidad de bytes que dicha variable ocupa en la memoria principal.
Se presenta la siguiente tabla, que indica la cantidad de bytes que ocupa la representación interna de distintos tipos de datos en un compilador de Pascal típico.
Se recomienda **graficar** cada una de las situaciones planteadas a partir de una prueba de escritorio.

TIPO	CANTIDAD DE BYTES
Entero	2 bytes
Real	4 bytes
Char	1 byte
String	Tantos bytes como indique la longitud del String + 1
Record	La suma de las longitudes de los campos del registro
Puntero	4 bytes
Boolean	1 byte

Tabla de referencia de tamaño de los tipos de datos de Pascal (estos valores pueden variar entre diferentes implementaciones del compilador)

1) Indicar los valores que imprime el siguiente programa en Pascal. Justificar mediante prueba de escritorio.

```
program punteros;
type
  cadena = string[50];
  puntero_cadena = ^cadena;
var
  pc: puntero_cadena;
begin
  writeln(sizeof(pc), ' bytes');
  new(pc);
  writeln(sizeof(pc), ' bytes');
  pc^:= 'un nuevo nombre';
  writeln(sizeof(pc), ' bytes');
  writeln(sizeof(pc^), ' bytes');
  pc^:= 'otro nuevo nombre distinto al anterior';
  writeln(sizeof(pc^), ' bytes');
end.
```

2) Indicar los valores que imprime el siguiente programa en Pascal. Justificar mediante prueba de escritorio.

```
program punteros;
type
  cadena = string[9];
  producto = record
    codigo: integer;     2
    descripcion: cadena; 10
    precio: real;        4
  end;

  puntero_producto = ^producto;

var
  p: puntero_producto;
  prod: producto;
begin
  writeln(sizeof(p), ' bytes');          4 bytes
  writeln(sizeof(prod), ' bytes');      16 bytes
  new(p);
  writeln(sizeof(p), ' bytes');          4 bytes
  p^.codigo      := 1;
  p^.descripcion := 'nuevo producto';
  writeln(sizeof(p^), ' bytes');        16 bytes
  p^.precio := 200;
  writeln(sizeof(p^), ' bytes');        16 bytes
  prod.codigo := 2;
  prod.descripcion := 'otro nuevo producto';
  writeln(sizeof(prod), ' bytes');      16 bytes
end.
```

3) Indicar los valores que imprime el siguiente programa en Pascal. Justificar mediante prueba de escritorio.

```
program punteros;
type
  numeros = array[1..10000] of integer;
  puntero_numeros = ^numeros;

var
  n: puntero_numeros;
  num: numeros;
  i:integer;
begin
  writeln(sizeof(n), ' bytes');         4 bytes
  writeln(sizeof(num), ' bytes');       20000 bytes
  new(n);
  writeln(sizeof(n^), ' bytes');        20000 bytes
  for i:= 1 to 5000 do
    n^[i]:= i;
  writeln(sizeof(n^), ' bytes');        20000 bytes
end.
```

4) Indicar los valores que imprimen los siguientes programas en Pascal. Justificar mediante prueba de escritorio.

a) `program punteros;`

```
type
  cadena = string[50];
  puntero_cadena = ^cadena;
var
  pc: puntero_cadena;
begin
  pc^:= 'un nuevo texto';
  new(pc);
  writeln(pc^);
end.
```

b)
```
program punteros;
type
  cadena = string[50];
  puntero_cadena = ^cadena;
var
  pc: puntero_cadena;
begin
  new(pc);
  pc^:= 'un nuevo nombre';
  writeln(sizeof(pc^), ' bytes');
  writeln(pc^);
  dispose(pc);
  pc^:= 'otro nuevo nombre';
end.
```

c)
```
program punteros;
type
  cadena = string[50];
  puntero_cadena = ^cadena;

procedure cambiarTexto(pun: puntero_cadena);
begin
  pun^:= 'Otro texto';
end;

var
  pc: puntero_cadena;
begin
  new(pc);
  pc^:= 'Un texto';
  writeln(pc^);
  cambiarTexto(pc);
  writeln(pc^);
end.
```

d)
```
program punteros;
type
  cadena = string[50];
  puntero_cadena = ^cadena;

procedure cambiarTexto(pun: puntero_cadena);
begin
  new(pun);
  pun^:= 'Otro texto';
end;

var
  pc: puntero_cadena;
```

```
begin
  new(pc);
  pc^:= 'Un texto';
  writeln(pc^);
  cambiarTexto(pc);
  writeln(pc^);
end.
```

5) De acuerdo a los valores de la tabla que indica la cantidad de bytes que ocupa la representación interna de cada tipo de dato en Pascal, calcular el tamaño de memoria en los puntos señalados a partir de **(I)**, suponiendo que las variables del programa ya están declaradas y se cuenta con 400.000 bytes libres. Justificar mediante prueba de escritorio.

```
Program Alocacion_Dinamica;
Type
  TEmpleado = record
        sucursal: char;
        apellido: string[25];
        correoElectrónico: string[40];
        sueldo: real;
  end;
  Str50 = string[50];

Var
  alguien: TEmpleado;
  PtrEmpleado: ^TEmpleado;
Begin
  {Suponer que en este punto se cuenta con 400.000 bytes de memoria disponible  (I)}
  Readln( alguien.apellido );
  {Pensar si la lectura anterior ha hecho variar la cantidad de memoria         (II)}
  New( PtrEmpleado );
  {¿Cuánta memoria disponible hay ahora?                                       (III)}
  Readln( PtrEmpleado^.Sucursal, PtrEmpleado^.apellido );
  Readln( PtrEmpleado^.correoElectrónico, PtrEmpleado^.sueldo );
  {¿Cuánta memoria disponible hay ahora?                                        (IV)}
  Dispose( PtrEmpleado );
  {¿Cuánta memoria disponible hay ahora?                                         (V)}
end.
```

6) Realizar un programa que ocupe 50 KB de memoria en total. Para ello:

a) El programa debe utilizar sólo memoria estática
b) El programa debe utilizar el 50% de memoria estática y el 50% de memoria dinámica
c) El programa debe minimizar tanto como sea posible el uso de la memoria estática (a lo sumo, 4 bytes)

7) Se desea almacenar en memoria una secuencia de 2500 nombres de ciudades, cada nombre podrá tener una longitud máxima de 50 caracteres.

a) Definir una estructura de datos estática que permita guardar la información leída. Calcular el tamaño de memoria que requiere la estructura.

b) Dado que un compilador de Pascal típico no permite manejar estructuras de datos estáticas que superen los 64 Kb, pensar en utilizar un vector de punteros a palabras, como se indica en la siguiente estructura:

```
Type  Nombre = String[50];
      Puntero = ^Nombre;
      ArrPunteros = array[1..2500] of Puntero;
Var
      Punteros: ArrPunteros;
```

b.1) Indicar cuál es el tamaño de la variable Punteros al comenzar el programa.
b.2) Escribir un módulo que permita reservar memoria para los 2500 nombres. ¿Cuál es la cantidad de memoria reservada después de ejecutar el módulo? ¿La misma corresponde a memoria estática o dinámica?
b.3) Escribir un módulo para leer los nombres y almacenarlos en la estructura de la variable Punteros.

8) Analice el siguiente programa:

```
1.  program memoria;
2.  type
3.    datos = array [1..20] of integer;
4.    punt = ^datos;
5.
6.  procedure procesarDatos(v : datos; var v2 : datos);
7.  var
8.    i, j : integer;
9.  begin
10.    for i := 1 to 20 do
11.       v2[21 - i] := v[i];
12. end;
13.
14. var
15.    vect : datos;
16.    pvect : punt;
17.    i : integer;
18. begin
19.    for i:= 1 to 20 do
20.      vect[i] := i;
21.    new(pvect);
22.    for i:= 20 downto 1 do
23.      pvect^[i] := 0;
24.   procesarDatos(pvect^, vect);
25.   writeln('fin');
26. end.
```

Responda: ¿cuánta memoria en total ocupa el programa al ejecutarse? Considere tanto variables estáticas como dinámicas, parámetros y variables locales de los módulos.

a) Hasta la sentencia de la línea 18
b) Hasta la sentencia de la línea 20
c) Hasta la sentencia de la línea 23
d) Hasta la sentencia de la línea 11
e) Hasta la sentencia de la línea 25