Juan Menduiña

Trabajo Práctico N° 3:
Registros. Ejercicios con Corte de Control.

Ejercicio 1.

Dado el siguiente programa:

```
program TP3_E1;
{$codepage UTF8}
uses crt;
type
  str20=string[20];
  alumno=record
    codigo: integer;
    nombre: str20;
    promedio: real;
  end;
procedure leer(var alu: alumno);
begin
  textcolor(green); write('Introducir código del alumno: '); textcolor(yellow);
readln(alu.codigo);
  if (alu.codigo<>0) then
  begin
    textcolor(green); write('Introducir nombre del alumno: '); textcolor(yellow);
readln(alu.nombre);
    textcolor(green); write('Introducir promedio del alumno: '); textcolor(yellow);
readln(alu.promedio);
  end;
end;
var
  a: alumno;
begin

end.
```

(a) *Completar el programa principal para que lea información de alumnos (código, nombre, promedio) e informe la cantidad de alumnos leídos. La lectura finaliza cuando ingresa un alumno con código 0, que no debe procesarse. Nota: utilizar el módulo leer.*

```
program TP3_E1a;
{$codepage UTF8}
uses crt;
const
  alumno_salida=0;
type
  str20=string[20];
  alumno=record
    codigo: integer;
    nombre: str20;
    promedio: real;
  end;
procedure leer(var alu: alumno);
begin
  textcolor(green); write('Introducir código del alumno: '); textcolor(yellow);
readln(alu.codigo);
  if (alu.codigo<>alumno_salida) then
  begin
    textcolor(green); write('Introducir nombre del alumno: '); textcolor(yellow);
readln(alu.nombre);
```

```
    textcolor(green); write('Introducir promedio del alumno: '); textcolor(yellow);
readln(alu.promedio);
  end;
end;
var
  a: alumno;
  alumnos_leidos: integer;
begin
  alumnos_leidos:=0;
  leer(a);
  while (a.codigo<>alumno_salida) do
  begin
    alumnos_leidos:=alumnos_leidos+1;
    leer(a);
  end;
  textcolor(green); write('La cantidad de alumnos leídos es '); textcolor(red);
write(alumnos_leidos);
end.
```

(b) *Modificar al programa anterior para que, al finalizar la lectura de todos los alumnos, se informe también el nombre del alumno con mejor promedio.*

```
program TP3_E1b;
{$codepage UTF8}
uses crt;
const
  alumno_salida=0;
type
  str20=string[20];
  alumno=record
    codigo: integer;
    nombre: str20;
    promedio: real;
  end;
procedure leer(var alu: alumno);
begin
  textcolor(green); write('Introducir código del alumno: '); textcolor(yellow);
readln(alu.codigo);
  if (alu.codigo<>alumno_salida) then
  begin
    textcolor(green); write('Introducir nombre del alumno: '); textcolor(yellow);
readln(alu.nombre);
    textcolor(green); write('Introducir promedio del alumno: '); textcolor(yellow);
readln(alu.promedio);
  end;
end;
procedure promedios(alu: alumno; var promedio_max: real; var alumno_max: str20);
begin
  if (alu.promedio>promedio_max) then
  begin
    promedio_max:=alu.promedio;
    alumno_max:=alu.nombre;
  end;
end;
var
  a: alumno;
  alumnos_leidos: integer;
  promedio_max: real;
  alumno_max: str20;
begin
  alumnos_leidos:=0;
  leer(a);
```

```
  while (a.codigo<>alumno_salida) do
  begin
    alumnos_leidos:=alumnos_leidos+1;
    promedios(a,promedio_max,alumno_max);
    leer(a);
  end;
  textcolor(green); write('La cantidad de alumnos leidos es '); textcolor(red);
writeln(alumnos_leidos);
  textcolor(green); write('El nombre del alumno con mejor promedio es '); textcolor(red);
write(alumno_max);
end.
```

Ejercicio 2.

El registro civil de La Plata ha solicitado un programa para analizar la distribución de casamientos durante el año 2019. Para ello, cuenta con información de las fechas de todos los casamientos realizados durante ese año.

(a) *Analizar y definir un tipo de dato adecuado para almacenar la información de la fecha de cada casamiento.*

```
type
  t_dia=1..31;
  t_mes=1..12;
  t_registro_casamiento=record
    dia: t_dia;
    mes: t_mes;
    anio: int16;
  end;
```

(b) *Implementar un módulo que lea una fecha desde teclado y la retorne en un parámetro cuyo tipo es el definido en el inciso (a).*

```
procedure leer_casamientos(var casamientos_verano, casamientos_1a10: int16);
var
  registro_casamiento: t_registro_casamiento;
begin
  leer_casamiento(registro_casamiento);
  while (registro_casamiento.anio<>anio_salida) do
  begin
    if ((registro_casamiento.mes=1) or (registro_casamiento.mes=2) or
(registro_casamiento.mes=3)) then
      casamientos_verano:=casamientos_verano+1;
    if (registro_casamiento.dia<=dia_corte) then
      casamientos_1a10:=casamientos_1a10+1;
    leer_casamiento(registro_casamiento);
  end;
end;
```

(c) *Implementar un programa que lea la fecha de todos los casamientos realizados en 2019. La lectura finaliza al ingresar el año 2020, que no debe procesarse, e informe la cantidad de casamientos realizados durante los meses de verano (enero, febrero y marzo) y la cantidad de casamientos realizados en los primeros 10 días de cada mes. Nota: utilizar el módulo realizado en (b) para la lectura de fecha.*

```
program TP3_E2;
{$codepage UTF8}
uses crt;
const
  anio_salida=2020; dia_corte=10;
type
  t_dia=1..31;
  t_mes=1..12;
  t_registro_casamiento=record
    dia: t_dia;
```

```
    mes: t_mes;
    anio: int16;
  end;
procedure leer_casamiento(var registro_casamiento: t_registro_casamiento);
begin
  textcolor(green); write('Introducir año del casamiento: '); textcolor(yellow);
readln(registro_casamiento.anio);
  if (registro_casamiento.anio<>anio_salida) then
  begin
    textcolor(green); write('Introducir mes del casamiento: '); textcolor(yellow);
readln(registro_casamiento.mes);
    textcolor(green); write('Introducir día del casamiento: '); textcolor(yellow);
readln(registro_casamiento.dia);
  end;
end;
procedure leer_casamientos(var casamientos_verano, casamientos_1a10: int16);
var
  registro_casamiento: t_registro_casamiento;
begin
  leer_casamiento(registro_casamiento);
  while (registro_casamiento.anio<>anio_salida) do
  begin
    if ((registro_casamiento.mes=1) or (registro_casamiento.mes=2) or
(registro_casamiento.mes=3)) then
      casamientos_verano:=casamientos_verano+1;
    if (registro_casamiento.dia<=dia_corte) then
      casamientos_1a10:=casamientos_1a10+1;
    leer_casamiento(registro_casamiento);
  end;
end;
var
  casamientos_verano, casamientos_1a10: int16;
begin
  casamientos_verano:=0; casamientos_1a10:=0;
  leer_casamientos(casamientos_verano,casamientos_1a10);
  textcolor(green); write('La cantidad de casamientos realizados durante los meses de verano
(enero, febrero y marzo) es '); textcolor(red); writeln(casamientos_verano);
  textcolor(green); write('La cantidad de casamientos realizados en los primeros 10 días de
cada mes es '); textcolor(red); write(casamientos_1a10);
end.
```

Ejercicio 3.

El Ministerio de Educación desea realizar un relevamiento de las 2400 escuelas primarias de la provincia de Bs. As., con el objetivo de evaluar si se cumple la proporción de alumnos por docente calculada por la UNESCO para el año 2015 (1 docente cada 23,435 alumnos). Para ello, se cuenta con información de: CUE (código único de establecimiento), nombre del establecimiento, cantidad de docentes, cantidad de alumnos, localidad. Se pide implementar un programa que procese la información y determine:

- *Cantidad de escuelas de La Plata con una relación de alumnos por docente superior a la sugerida por UNESCO.*
- *CUE y nombre de las dos escuelas con mejor relación entre docentes y alumnos.*

El programa debe utilizar:

- *Un módulo para la lectura de la información de la escuela.*
- *Un módulo para determinar la relación docente-alumno (esa relación se obtiene del cociente entre la cantidad de alumnos y la cantidad de docentes).*

```
program TP3_E3;
{$codepage UTF8}
uses crt;
const
  escuelas_total=2400; ratio_unesco=23.435;
type
  registro_escuela=record
    cue: int16;
    nombre: string;
    docentes: int16;
    alumnos: int16;
    localidad: string;
  end;
procedure leer_escuela(var escuela: registro_escuela);
begin
  textcolor(green); write('Introducir CUE (Código Único de Establecimiento) de la escuela:
'); textcolor(yellow); readln(escuela.cue);
  textcolor(green); write('Introducir nombre de la escuela: '); textcolor(yellow);
readln(escuela.nombre);
  textcolor(green); write('Introducir cantidad de docentes de la escuela: ');
textcolor(yellow); readln(escuela.docentes);
  textcolor(green); write('Introducir cantidad de alumnos de la escuela: ');
textcolor(yellow); readln(escuela.alumnos);
  textcolor(green); write('Introducir localidad de la escuela: '); textcolor(yellow);
readln(escuela.localidad);
end;
function ratio_alumnos_docente(escuela: registro_escuela): real;
begin
    ratio_alumnos_docente:=escuela.alumnos/escuela.docentes;
end;
procedure actualizar_minimos(ratio: real; escuela: registro_escuela; var ratio_min1,
ratio_min2: real; var cue_min1, cue_min2: int16; var nombre_min1, nombre_min2: string);
begin
  if (ratio<ratio_min1) then
  begin
    ratio_min2:=ratio_min1;
    cue_min2:=cue_min1;
    nombre_min2:=nombre_min1;
    nombre_min1:=escuela.nombre;
    cue_min1:=escuela.cue;
  end
  else
```

```
    if (ratio<ratio_min2) then
    begin
      ratio_min2:=ratio;
      cue_min2:=escuela.cue;
      nombre_min2:=escuela.nombre;
  end;
end;
procedure leer_escuelas(var escuelas_lp, cue_min1, cue_min2: int16; var nombre_min1,
nombre_min2: string);
var
  escuela: registro_escuela;
  i: int16;
  ratio, ratio_min1, ratio_min2: real;
begin
  ratio:=0; ratio_min1:=999999; ratio_min2:=999999;
  for i:= 1 to escuelas_total do
  begin
    leer_escuela(escuela);
    ratio:=ratio_alumnos_docente(escuela);
    actualizar_minimos(ratio,escuela,ratio_min1,ratio_min2,cue_min1,cue_min2,nombre_min1,nom
bre_min2);
    if ((escuela.localidad='La Plata') and (ratio>ratio_unesco)) then
      escuelas_lp:=escuelas_lp+1;
  end;
end;
var
  escuelas_lp, cue_min1, cue_min2: int16;
  nombre_min1, nombre_min2: string;
begin
  escuelas_lp:=0; cue_min1:=0; cue_min2:=0;
  leer_escuelas(escuelas_lp,cue_min1,cue_min2,nombre_min1,nombre_min2);
  textcolor(green); write('La cantidad de escuelas de La Plata con una relación de alumnos
por docente superior a la sugerida por UNESCO es '); textcolor(red); writeln(escuelas_lp);
  textcolor(green); write('Los CUEs de las dos escuelas con mejor relación entre docentes y
alumnos son '); textcolor(red); write(cue_min1); textcolor(green); write(' y ');
textcolor(red); writeln(cue_min2);
  textcolor(green); write('Los nombres de las dos escuelas con mejor relación entre docentes
y alumnos son '); textcolor(red); write(nombre_min1); textcolor(green); write(' y ');
textcolor(red); writeln(nombre_min2);
end.
```

Ejercicio 4.

Una compañía de telefonía celular debe realizar la facturación mensual de sus 9300 clientes con planes de consumo ilimitados (clientes que pagan por lo que consumen). Para cada cliente, se conoce su código de cliente y cantidad de líneas a su nombre. De cada línea, se tiene el número de teléfono, la cantidad de minutos consumidos y la cantidad de MB consumidos en el mes. Se pide implementar un programa que lea los datos de los clientes de la compañía e informe el monto total a facturar para cada uno. Para ello, se requiere:

- *Realizar un módulo que lea la información de una línea de teléfono.*
- *Realizar un módulo que reciba los datos de un cliente, lea la información de todas sus líneas (utilizando el módulo desarrollado en el inciso (a) y retorne la cantidad total de minutos y la cantidad total de MB a facturar del cliente.*

Nota: para realizar los cálculos tener en cuenta que cada minuto cuesta $3,40 y cada MB consumido cuesta $1,35.

Ejercicio 5.

Realizar un programa que lea información de autos que están a la venta en una concesionaria. De cada auto, se lee: marca, modelo y precio. La lectura finaliza cuando se ingresa la marca "ZZZ" que no debe procesarse. La información se ingresa ordenada por marca. Se pide calcular e informar:

- *El precio promedio por marca.*
- *Marca y modelo del auto más caro.*

```
program TP3_E5;
{$codepage UTF8}
uses crt;
const
  marca_salida='ZZZ';
type
  t_registro_auto=record
    marca: string;
    modelo: string;
    precio: int32;
  end;
procedure leer_auto(var registro_auto: t_registro_auto);
begin
  textcolor(green); write('Introducir marca del auto: '); textcolor(yellow);
readln(registro_auto.marca);
  if (registro_auto.marca<>marca_salida) then
  begin
    textcolor(green); write('Introducir modelo del auto: '); textcolor(yellow);
readln(registro_auto.modelo);
    textcolor(green); write('Introducir precio del auto: '); textcolor(yellow);
readln(registro_auto.precio);
  end;
end;
procedure actualizar_maximos(registro_auto: t_registro_auto; var precio_max: int32; var
marca_max, modelo_max: string);
begin
  if (registro_auto.precio>precio_max) then
  begin
    precio_max:=registro_auto.precio;
    marca_max:=registro_auto.marca;
    modelo_max:=registro_auto.modelo;
  end;
end;
procedure leer_autos(var marca_max, modelo_max: string);
var
  registro_auto: t_registro_auto;
  precio_max, precio_sum, autos_total: int32;
  precio_prom: real;
  marca: string;
begin
  precio_max:=0;
  leer_auto(registro_auto);
  while (registro_auto.marca<>marca_salida) do
  begin
    marca:=registro_auto.marca; precio_sum:=0; autos_total:=0; precio_prom:=0;
    while ((registro_auto.marca=marca) and (registro_auto.marca<>marca_salida)) do
    begin
      precio_sum:=precio_sum+registro_auto.precio;
      autos_total:=autos_total+1;
      actualizar_maximos(registro_auto,precio_max,marca_max,modelo_max);
      leer_auto(registro_auto);
    end;
    precio_prom:=precio_sum/autos_total;
```

```
    textcolor(green); write('El promedio de la marca '); textcolor(red); write(marca);
textcolor(green); write(' es '); textcolor(red); writeln(precio_prom:0:2);
  end;
end;
var
  marca_max, modelo_max: string;
begin
  leer_autos(marca_max,modelo_max);
  textcolor(green); write('La marca y el modelo del auto más caro son '); textcolor(red);
write(marca_max); textcolor(green); write(' y '); textcolor(red); write(modelo_max);
textcolor(green); write(', respectivamente');
end.
```

Ejercicio 6.

Una empresa importadora de microprocesadores desea implementar un sistema de software para analizar la información de los productos que mantiene actualmente en stock. Para ello, se conoce la siguiente información de los microprocesadores: marca (Intel, AMD, NVidia, etc.), línea (Xeon, Core i7, Opteron, Atom, Centrino, etc.), cantidad de cores o núcleos de procesamiento (1, 2, 4, 8), velocidad del reloj (medida en Ghz) y tamaño en nanómetros (nm) de los transistores (14, 22, 32, 45, etc.). La información de los microprocesadores se lee de forma consecutiva por marca de procesador y la lectura finaliza al ingresar un procesador con 0 cores (que no debe procesarse). Se pide implementar un programa que lea información de los microprocesadores de la empresa importadora e informe:

- *Marca y línea de todos los procesadores de más de 2 cores con transistores de, a lo sumo, 22 nm.*
- *Las dos marcas con mayor cantidad de procesadores con transistores de 14 nm.*
- *Cantidad de procesadores multicore (de más de un core) de Intel o AMD, cuyos relojes alcancen velocidades de, al menos, 2 Ghz.*

```
program TP3_E6;
{$codepage UTF8}
uses crt;
const
  cores_salida=0; cores_obj=2; transistores_obj1=22; transistores_obj2=14;
velocidad_obj=2.0;
type
  t_registro_procesador=record
    cores: int16;
    marca: string;
    linea: string;
    velocidad: real;
    transistores: int16;
  end;
procedure leer_procesador(var registro_procesador: t_registro_procesador);
begin
  textcolor(green); write('Introducir cantidad de cores o núcleos de procesamiento del
procesador (1, 2, 4, 8): '); textcolor(yellow); readln(registro_procesador.cores);
  if (registro_procesador.cores<>cores_salida) then
  begin
    textcolor(green); write('Introducir marca del procesador (Intel, AMD, NVidia): ');
textcolor(yellow); readln(registro_procesador.marca);
    textcolor(green); write('Introducir línea del procesador (Xeon, Core i7, Opteron, Atom,
Centrino): '); textcolor(yellow); readln(registro_procesador.linea);
    textcolor(green); write('Introducir velocidad del reloj (medida en Ghz): ');
textcolor(yellow); readln(registro_procesador.velocidad);
    textcolor(green); write('Introducir tamaño en nanómetros (nm) de los transistores (14,
22, 32, 45): '); textcolor(yellow); readln(registro_procesador.transistores);
  end;
end;
procedure actualizar_maximos(transistores_marca: int16; marca: string; var
transistores_max1, transistores_max2: int16; var marca_max1, marca_max2: string);
begin
  if (transistores_marca>transistores_max1) then
  begin
    transistores_max2:=transistores_max1;
    marca_max2:=marca_max1;
    transistores_max1:=transistores_marca;
    marca_max1:=marca;
  end
  else
```

```
    if (transistores_marca>transistores_max2) then
    begin
      transistores_max2:=transistores_marca;
      marca_max2:=marca;
    end;
end;
procedure leer_procesadores(var procesadores_intel_amd: int16; var marca_max1, marca_max2:
string);
var
  registro_procesador: t_registro_procesador;
  transistores_marca, transistores_max1, transistores_max2: int16;
  marca: string;
begin
  transistores_max1:=0; transistores_max2:=0;
  leer_procesador(registro_procesador);
  while (registro_procesador.cores<>cores_salida) do
  begin
    marca:=registro_procesador.marca; transistores_marca:=0;
    while ((registro_procesador.marca=marca) and (registro_procesador.cores<>cores_salida))
do
    begin
      if ((registro_procesador.cores>cores_obj) and
(registro_procesador.transistores<=transistores_obj1)) then
      begin
        textcolor(green); write('La marca y la línea de este registro_procesador con más de
2 cores con transistores de, a lo sumo, 22 nm. son '); textcolor(red);
write(registro_procesador.marca); textcolor(green); write(' y '); textcolor(red);
writeln(registro_procesador.linea);
      end;
      if (registro_procesador.transistores=transistores_obj2) then
        transistores_marca:=transistores_marca+1;
      if ((registro_procesador.cores>=cores_obj) and ((registro_procesador.marca='Intel') or
(registro_procesador.marca='AMD')) and (registro_procesador.velocidad>=velocidad_obj)) then
        procesadores_intel_amd:=procesadores_intel_amd+1;
      leer_procesador(registro_procesador);
    end;
    actualizar_maximos(transistores_marca,marca,transistores_max1,transistores_max2,marca_ma
x1,marca_max2);
  end;
end;
var
  procesadores_intel_amd: int16;
  marca_max1, marca_max2: string;
begin
  procesadores_intel_amd:=0;
  leer_procesadores(procesadores_intel_amd,marca_max1,marca_max2);
  textcolor(green); write('Las dos marcas con mayor cantidad de registro_procesadores con
transistores de 14 nm. son '); textcolor(red); write(marca_max1); textcolor(green); write('
y '); textcolor(red); writeln(marca_max2);
  textcolor(green); write('La cantidad de registro_procesadores multicore (de más de un
core) de Intel o AMD, cuyos relojes alcancen velocidades de, al menos, 2 Ghz es ');
textcolor(red); write(procesadores_intel_amd);
end.
```

Ejercicio 7.

Realizar un programa que lea información de centros de investigación de Universidades Nacionales. De cada centro, se lee su nombre abreviado (ej., LIDI, LIFIA, LINTI), la universidad a la que pertenece, la cantidad de investigadores y la cantidad de becarios que poseen. La información se lee de forma consecutiva por universidad y la lectura finaliza al leer un centro con 0 investigadores, que no debe procesarse. Informar:

- *Cantidad total de centros para cada universidad.*
- *Universidad con mayor cantidad de investigadores en sus centros.*
- *Los dos centros con menor cantidad de becarios.*

Ejercicio 8.

La Comisión Provincial por la Memoria desea analizar la información de los proyectos presentados en el programa Jóvenes y Memoria durante la convocatoria 2020. Cada proyecto, posee un código único, un título, el docente coordinador (DNI, nombre y apellido, email), la cantidad de alumnos que participan del proyecto, el nombre de la escuela y la localidad a la que pertenecen. Cada escuela, puede presentar más de un proyecto. La información se ingresa ordenada consecutivamente por localidad y, para cada localidad, por escuela. Realizar un programa que lea la información de los proyectos hasta que se ingrese el proyecto con código -1 (que no debe procesarse), e informe:

- *Cantidad total de escuelas que participan en la convocatoria 2018 y cantidad de escuelas por cada localidad.*
- *Nombres de las dos escuelas con mayor cantidad de alumnos participantes.*
- *Título de los proyectos de la localidad de Daireaux cuyo código posee igual cantidad de dígitos pares e impares.*

Ejercicio 9.

Realizar un programa que lea información de los candidatos ganadores de las últimas elecciones a intendente de la provincia de Buenos Aires. Para cada candidato, se lee: localidad, apellido del candidato, cantidad de votos obtenidos y cantidad de votantes de la localidad. La lectura finaliza al leer la localidad 'Zárate', que debe procesarse. Informar:

- *El intendente que obtuvo la mayor cantidad de votos en la elección.*
- *El intendente que obtuvo el mayor porcentaje de votos de la elección.*

Ejercicio 10.

Un centro de investigación de la UNLP está organizando la información de las 320 especies de plantas con las que trabajan. Para cada especie, se ingresa su nombre científico, tiempo promedio de vida (en meses), tipo de planta (por ej., árbol, conífera, arbusto, helecho, musgo, etc.), clima (templado, continental, subtropical, desértico, etc.) y países en el mundo donde se las encuentra. La información de las plantas se ingresa ordenada por tipo de planta y, para cada planta, la lectura de países donde se las encuentra finaliza al ingresar el país 'zzz'. Al finalizar la lectura, informar:

- *El tipo de planta con menor cantidad de plantas.*
- *El tiempo promedio de vida de las plantas de cada tipo.*
- *El nombre científico de las dos plantas más longevas.*
- *Los nombres de las plantas nativas de Argentina que se encuentran en regiones con clima subtropical.*
- *El nombre de la planta que se encuentra en más países.*

Ejercicio 11.

Una compañía de vuelos internacionales está analizando la información de todos los vuelos realizados por sus aviones durante todo el año 2019. De cada vuelo, se conoce el código de avión, país de salida, país de llegada, cantidad de kilómetros recorridos y porcentaje de ocupación del avión. La información se ingresa ordenada por código de avión y, para cada avión, por país de salida. La lectura finaliza al ingresar el código 44. Informar:

- *Los dos aviones que más kilómetros recorrieron y los dos aviones que menos kilómetros recorrieron.*
- *El avión que salió desde más países diferentes.*
- *La cantidad de vuelos de más de 5.000 km que no alcanzaron el 60% de ocupación del avión.*
- *La cantidad de vuelos de menos de 10.000 km que llegaron a Australia o a Nueva Zelanda.*

Ejercicio 12.

En la "Práctica 1 - Ejercicios Adicionales", se resolvieron 3 problemas complejos sin utilizar módulos. Al carecer de herramientas para modularizar, esos programas resultaban difíciles de leer, de extender y de depurar. En la "Práctica 2 (parte 2) - Ejercicios Adicionales", se adaptaron los 3 problemas para utilizar módulos y, así, organizar mejor el programa. Ahora, podemos incluir los registros y, así, seguir mejorando nuestros programas. Para cada caso, analizar:

- *¿Qué entidades del programa conviene representar como registros?*
- *¿Qué atributos de cada entidad deben incluirse en los registros?*
- *¿Qué cambios deben realizarse en los módulos implementados en la práctica 2 para aprovechar los nuevos tipos de datos? ¿Conviene seguir utilizando los mismos módulos en todos los casos?*

Una vez realizado el análisis, modificar los 3 problemas, utilizando registros para representar los datos del programa. Al finalizar cada problema, comparar la solución usando registros y módulos con la solución sin registros y con módulos (Práctica 2) y con la solución sin registros ni módulos (Práctica 1).

- *¿Qué diferencias observa?*
- *¿Qué similitudes encuentra?*

Juan Menduiña

Trabajo Práctico N° 4.1:
Vectores (Parte 1).

Ejercicio 1.

```
program TP4_E1;
{$codepage UTF8}
uses crt;
type
  vnums=array[1..10] of integer;
var
  numeros: vnums;
  i: integer;
begin
  for i:=1 to 10 do
    numeros[i]:= i;
  for i:= 2 to 10 do
    numeros[i]:=numeros[i]+numeros[i-1]
end.
```

(a) *¿Qué valores toma la variable numeros al finalizar el primer bloque for?*

Los valores que toma la variable numeros al finalizar el primer bloque for son 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.

(b) *Al terminar el programa, ¿con qué valores finaliza la variable numeros?*

Al terminar el programa, la variable numeros finaliza con los valores 1, 3, 6, 10, 15, 21, 28, 36, 45, 55.

(c) *Si se desea cambiar la línea 11 por la sentencia: for i:=1 to 9 do, ¿cómo debe modificarse el código para que la variable números contenga los mismos valores que en (1.b)?*

```
program TP4_E1c;
{$codepage UTF8}
uses crt;
type
  vnums=array[1..10] of integer;
var
  numeros: vnums;
  i, j: integer;
begin
  for i:= 1 to 10 do
    numeros[i]:= i;
  for i:= 1 to 9 do
  begin
    j:=i+1;
    numeros[j]:=numeros[j]+numeros[j-1];
    writeln(numeros[j]);
  end;
end.
```

(d) *¿Qué valores están contenidos en la variable numeros si las líneas 11 y 12 se reemplazan por for i:= 1 to 9 do numeros[i+1]:=numeros[i];?*

```
program TP4_E1d;
{$codepage UTF8}
uses crt;
type
  vnums=array[1..10] of integer;
var
  numeros: vnums;
  i, j: integer;
begin
  for i:= 1 to 10 do
    numeros[i]:= i;
  for i:= 1 to 9 do
  begin
    numeros[i+1]:=numeros[i];
    writeln(numeros[i]);
  end;
end.
```

Ejercicio 2.

Dado el siguiente programa, completar las líneas indicadas, considerando que:

- *El módulo cargarVector debe leer números reales y almacenarlos en el vector que se pasa como parámetro. Al finalizar, debe retornar el vector y su dimensión lógica. La lectura finaliza cuando se ingresa el valor 0 (que no debe procesarse) o cuando el vector está completo.*
- *El módulo modificarVectorySumar debe devolver el vector con todos sus elementos incrementados con el valor n y también debe devolver la suma de todos los elementos del vector.*

```
program TP4_E2;
{$codepage UTF8}
uses crt;
const
  cant_datos=150;
type
  vdatos=array[1..cant_datos] of real;
procedure cargarVector(var v: vdatos; var dimL: integer);
var
  num_real: int16;
begin
  textcolor(green); write('Introducir número real: ');
  textcolor(yellow); readln(num_real);
  while ((num_real<>0) and (dimL<=cant_datos)) do
  begin
    dimL:=dimL+1;
    v[dimL]:=num_real;
    textcolor(green); write('Introducir número real: ');
    textcolor(yellow); readln(num_real);
  end;
end;
procedure modificarVectorySumar(var v: vdatos; dimL: integer; n: real; var suma: real);
var
  i: int16;
begin
  for i:= 1 to dimL do
  begin
    v[i]:=v[i]+n;
    suma:=suma+v[i];
  end;
end;
var
  datos: vdatos;
  i, dim: integer;
  num, suma: real;
begin
  dim:=0; suma:=0;
  cargarVector(datos,dim);
  textcolor(green); write('Introducir valor a sumar: ');
  textcolor(yellow); readln(num);
  modificarVectorySumar(datos,dim,num,suma);
  textcolor(green); write('La suma de los valores es '); textcolor(red); writeln(suma:0:2);
  textcolor(green); write('Se procesaron '); textcolor(red); write(dim); textcolor(green);
write(' números');
end.
```

Ejercicio 3.

Se dispone de un vector con números enteros, de dimensión física dimF y dimensión lógica dimL.

(a) *Realizar un módulo que imprima el vector desde la primera posición hasta la última.*

```
procedure imprimir_1adimL(vector: t_vector; dimL: int16);
var
  i: int16;
begin
  for i:= 1 to dimL do
    writeln(vector[i]);
end;
```

(b) *Realizar un módulo que imprima el vector desde la última posición hasta la primera.*

```
procedure imprimir_dimLa1(vector: t_vector; dimL: int16);
var
  i: int16;
begin
  for i:= dimL downto 1 do
    writeln(vector[i]);
end;
```

(c) *Realizar un módulo que imprima el vector desde la mitad (dimL DIV 2) hacia la primera posición, y desde la mitad más uno hacia la última posición.*

```
procedure imprimir_dimLdiv2(vector: t_vector; dimL: int16);
var
  i, dimLdiv2, dimLdiv2mas1: int16;
begin
  dimLdiv2:=dimL div 2; dimLdiv2mas1:=dimLdiv2+1;
  for i:= dimLdiv2 downto 1 do
    writeln(vector[i]);
  for i:= dimLdiv2mas1 to dimL do
    writeln(vector[i]);
end;
```

(d) *Realizar un módulo que reciba el vector, una posición X y otra posición Y, e imprima el vector desde la posición X hasta la Y. Asumir que tanto X como Y son menores o igual a la dimensión lógica. Y considerar que, dependiendo de los valores de X e Y, podría ser necesario recorrer hacia adelante o hacia atrás.*

```
procedure imprimir_general(vector: t_vector);
var
  i, numX, numY: int16;
begin
  textcolor(green); write('Introducir número entero X: ');
  textcolor(yellow); readln(numX);
```

```
  textcolor(green); write('Introducir número entero Y: ');
  textcolor(yellow); readln(numY);
  if (numX<=numY) then
  begin
    for i:= numX to numY do
      writeln(vector[i]);
  end
  else
    for i:= numX downto numY do
      writeln(vector[i]);
end;
```

(e) *Utilizando el módulo implementado en el inciso anterior, volver a realizar los incisos a, b y c.*

```
program TP4_E3;
{$codepage UTF8}
uses crt;
type
  t_vector=array of int16;
procedure crear_vector(var vector: t_vector; dimF: int16);
begin
  setLength(vector,dimF);
end;
procedure rellenar_vector(var vector: t_vector; limite_random, dimL: int16);
var
  i: int16;
begin
  for i:= 1 to dimL do
    vector[i]:=random(limite_random);
end;
procedure imprimir_1adimL(vector: t_vector; dimL: int16);
var
  i: int16;
begin
  for i:= 1 to dimL do
    writeln(vector[i]);
end;
procedure imprimir_dimLa1(vector: t_vector; dimL: int16);
var
  i: int16;
begin
  for i:= dimL downto 1 do
    writeln(vector[i]);
end;
procedure imprimir_dimLdiv2(vector: t_vector; dimL: int16);
var
  i, dimLdiv2, dimLdiv2mas1: int16;
begin
  dimLdiv2:=dimL div 2; dimLdiv2mas1:=dimLdiv2+1;
  for i:= dimLdiv2 downto 1 do
    writeln(vector[i]);
  for i:= dimLdiv2mas1 to dimL do
    writeln(vector[i]);
end;
procedure imprimir_general(vector: t_vector);
var
  i, numX, numY: int16;
begin
  textcolor(green); write('Introducir número entero X: ');
  textcolor(yellow); readln(numX);
  textcolor(green); write('Introducir número entero Y: ');
```

```
  textcolor(yellow); readln(numY);
  if (numX<=numY) then
  begin
    for i:= numX to numY do
      writeln(vector[i]);
  end
  else
    for i:= numX downto numY do
      writeln(vector[i]);
end;
var
  vector: t_vector;
  limite_random, dimF, dimL: int16;
begin
  randomize;
  textcolor(green); write('Introducir número entero como límite superior de una distribucion
aleatoria de la cual se extraerán elementos para el vector: ');
  textcolor(yellow); readln(limite_random);
  textcolor(green); write('Introducir número entero como dimensión física del vector: ');
  textcolor(yellow); readln(dimF);
  textcolor(green); write('Introducir número entero como dimensión lógica del vector: ');
  textcolor(yellow); readln(dimL);
  crear_vector(vector,dimF);
  rellenar_vector(vector,limite_random,dimL);
  imprimir_1adimL(vector,dimL);
  imprimir_dimLa1(vector,dimL);
  imprimir_dimLdiv2(vector,dimL);
  imprimir_general(vector);
end.
```

Ejercicio 4.

Se dispone de un vector con 100 números enteros. Implementar los siguientes módulos:

(a) *posicion: dado un número X y el vector de números, retorna la posición del número X en dicho vector, o el valor -1 en caso de no encontrarse.*

```
function posicion(vector: t_vector; dimL, numX: int16): int16;
var
  pos: int16;
begin
  pos:=0;
  while ((pos<=dimL) and (vector[pos]<>numX)) do
    pos:=pos+1;
  if (pos>dimL) then
    pos:=-1;
  posicion:=pos;
end;
```

(b) *intercambio: recibe dos valores x e y (entre 1 y 100) y el vector de números, y retorna el mismo vector, donde se intercambiaron los valores de las posiciones x e y.*

```
procedure intercambio(var vector: t_vector; numX, numY: int16);
var
  num_aux: int16;
begin
  num_aux:=vector[numX];
  vector[numX]:=vector[numY];
  vector[numY]:=num_aux;
end;
```

(c) *sumaVector: retorna la suma de todos los elementos del vector.*

```
function sumaVector(vector: t_vector; dimL: int16): int16;
var
  i, suma: int16;
begin
  suma:=0;
  for i:= 1 to dimL do
    suma:=suma+vector[i];
  sumaVector:=suma;
end;
```

(d) *promedio: devuelve el valor promedio de los elementos del vector.*

```
function promedio(vector: t_vector; dimL: int16): real;
begin
  promedio:=sumaVector(vector,dimL)/dimL;
end;
```

(e) *elementoMaximo: retorna la posición del mayor elemento del vector.*

```
function elementoMaximo(vector: t_vector; dimL: int16): int16;
var
  i, ele_max, pos_max: int16;
begin
  ele_max:=low(int16); pos_max:=low(int16);
  for i:= 1 to dimL do
  begin
    if (vector[i]>ele_max) then
    begin
      ele_max:=vector[i];
      pos_max:=i;
    end;
  end;
  elementoMaximo:=pos_max;
end;
```

(f) *elementoMinimo: retorna la posicion del menor elemento del vector.*

```
function elementoMinimo(vector: t_vector; dimL: int16): int16;
var
  i, ele_min, pos_min: int16;
begin
  ele_min:=high(int16); pos_min:=high(int16);
  for i:= 1 to dimL do
  begin
    if (vector[i]<ele_min) then
    begin
      ele_min:=vector[i];
      pos_min:=i;
    end;
  end;
  elementoMinimo:=pos_min;
end;
```

```
program TP4_E4;
{$codepage UTF8}
uses crt;
type
  t_vector=array of int16;
procedure crear_vector(var vector: t_vector; dimF: int16);
begin
  setLength(vector,dimF);
end;
procedure rellenar_vector(var vector: t_vector; dimL: int16);
var
  i: int16;
begin
  for i:= 1 to dimL do
    vector[i]:=random(maxint);
end;
function posicion(vector: t_vector; dimL, numX: int16): int16;
var
  pos: int16;
begin
  pos:=0;
  while ((pos<=dimL) and (vector[pos]<>numX)) do
    pos:=pos+1;
  if (pos>dimL) then
```

```
    pos:=-1;
  posicion:=pos;
end;
procedure intercambio(var vector: t_vector; numX, numY: int16);
var
  num_aux: int16;
begin
  num_aux:=vector[numX];
  vector[numX]:=vector[numY];
  vector[numY]:=num_aux;
end;
function sumaVector(vector: t_vector; dimL: int16): int16;
var
  i, suma: int16;
begin
  suma:=0;
  for i:= 1 to dimL do
    suma:=suma+vector[i];
  sumaVector:=suma;
end;
function promedio(vector: t_vector; dimL: int16): real;
begin
  promedio:=sumaVector(vector,dimL)/dimL;
end;
function elementoMaximo(vector: t_vector; dimL: int16): int16;
var
  i, ele_max, pos_max: int16;
begin
  ele_max:=low(int16); pos_max:=low(int16);
  for i:= 1 to dimL do
  begin
    if (vector[i]>ele_max) then
    begin
      ele_max:=vector[i];
      pos_max:=i;
    end;
  end;
  elementoMaximo:=pos_max;
end;
function elementoMinimo(vector: t_vector; dimL: int16): int16;
var
  i, ele_min, pos_min: int16;
begin
  ele_min:=high(int16); pos_min:=high(int16);
  for i:= 1 to dimL do
  begin
    if (vector[i]<ele_min) then
    begin
      ele_min:=vector[i];
      pos_min:=i;
    end;
  end;
  elementoMinimo:=pos_min;
end;
var
  vector: t_vector;
  dimF, dimL, numX, numY: int16;
begin
  textcolor(green); write('Introducir número entero como dimensión física del vector: ');
  textcolor(yellow); readln(dimF);
  textcolor(green); write('Introducir número entero como dimensión lógica del vector: ');
  textcolor(yellow); readln(dimL);
  crear_vector(vector,dimF);
  rellenar_vector(vector,dimL);
  textcolor(green); write('Introducir número entero X para buscar su posición (si existe) en
el vector: ');
```

```
  textcolor(yellow); readln(numX);
  textcolor(green); write('La posición del número '); textcolor(red); write(numX);
textcolor(green); write(' en el vector es '); textcolor(red);
writeln(posicion(vector,dimL,numX));
  textcolor(green); write('Introducir número entero X (entre 1 y 100): ');
  textcolor(yellow); readln(numX);
  textcolor(green); write('Introducir número entero Y (entre 1 y 100): ');
  textcolor(yellow); readln(numY);
  textcolor(green); write('Pre-intercambio, en las posiciones '); textcolor(red);
write(numX); textcolor(green); write(', '); textcolor(red); write(numY); textcolor(green);
write(', se tienen los valores '); textcolor(red); write(vector[numX]); textcolor(green);
write(' y '); textcolor(red); write(vector[numY]); textcolor(green); writeln(',
respectivamente');
  intercambio(vector,numX,numY);
  textcolor(green); write('Post-intercambio, en las posiciones '); textcolor(red);
write(numX); textcolor(green); write(', '); textcolor(red); write(numY); textcolor(green);
write(', se tienen los valores '); textcolor(red); write(vector[numX]); textcolor(green);
write(' y '); textcolor(red); write(vector[numY]); textcolor(green); writeln(',
respectivamente');
  textcolor(green); write('La suma de todos los elementos del vector es '); textcolor(red);
writeln(sumaVector(vector,dimL));
  textcolor(green); write('El valor promedio de los elementos del vector es ');
textcolor(red); writeln(promedio(vector,dimL):0:2);
  textcolor(green); write('La posición del mayor elemento del vector es '); textcolor(red);
writeln(elementoMaximo(vector,dimL));
  textcolor(green); write('La posición del menor elemento del vector es '); textcolor(red);
write(elementoMinimo(vector,dimL));
end.
```

Ejercicio 5.

Utilizando los módulos implementados en el ejercicio 4, realizar un programa que lea números enteros desde teclado (a lo sumo, 100) y los almacene en un vector. La carga finaliza al leer el número 0. Al finalizar la carga, se debe intercambiar la posición del mayor elemento por la del menor elemento e informar la operación realizada de la siguiente manera: "El elemento máximo ... que se encontraba en la posición ... fue intercambiado con el elemento mínimo ... que se encontraba en la posición ...".

```
program TP4_E5;
{$codepage UTF8}
uses crt;
const
  dimF=100; vector_salida=0;
type
  t_vector=array[1..dimF] of int16;
procedure rellenar_vector(var vector: t_vector; var dimL: int16; dimF: int16);
var
  num: int16;
begin
  textcolor(green); write('Introducir número entero para la posición ', dimL+1, ' del
vector: ');
  textcolor(yellow); readln(num);
  while ((num<>vector_salida) and (dimL<dimF)) do
  begin
    dimL:=dimL+1;
    vector[dimL]:=num;
    textcolor(green); write('Introducir número entero para la posición ', dimL+1, ' del
vector: ');
    textcolor(yellow); readln(num);
  end;
end;
procedure intercambio(var vector: t_vector; pos_max, pos_min: int16);
var
  num_aux: int16;
begin
  num_aux:=vector[pos_max];
  vector[pos_max]:=vector[pos_min];
  vector[pos_min]:=num_aux;
end;
procedure elementoMaximo(vector: t_vector; dimL: int16; var ele_max, pos_max: int16);
var
  i: int16;
begin
  for i:= 1 to dimL do
  begin
    if (vector[i]>ele_max) then
    begin
      ele_max:=vector[i];
      pos_max:=i;
    end;
  end;
end;
procedure elementoMinimo(vector: t_vector; dimL: int16; var ele_min, pos_min: int16);
var
  i: int16;
begin
  for i:= 1 to dimL do
  begin
    if (vector[i]<ele_min) then
    begin
      ele_min:=vector[i];
```

```
      pos_min:=i;
    end;
  end;
end;
var
  vector: t_vector;
  dimL, ele_max, ele_min, pos_max, pos_min: int16;
begin
  dimL:=0; ele_max:=low(int16); ele_min:=high(int16); pos_max:=0; pos_min:=0;
  rellenar_vector(vector,dimL,dimF);
  elementoMaximo(vector,dimL,ele_max,pos_max);
  elementoMinimo(vector,dimL,ele_min,pos_min);
  intercambio(vector,pos_max,pos_min);
  textcolor(green); write('El elemnento máximo '); textcolor(red); write(ele_max);
textcolor(green); write(', que se encontraba en la posición '); textcolor(red);
write(pos_max); textcolor(green); write(', fue intercambiado con el elemento mínimo ');
textcolor(red); write(ele_min); textcolor(green); write(', que se encontraba en la posición
'); textcolor(red); write(pos_min);
end.
```

Ejercicio 6.

Dado que en la solución anterior se recorre dos veces el vector (una para calcular el elemento máximo y otra para el mínimo), implementar un único módulo que recorra una única vez el vector y devuelva ambas posiciones.

```
program TP4_E6;
{$codepage UTF8}
uses crt;
const
  dimF=100; vector_salida=0;
type
  t_vector=array[1..dimF] of int16;
procedure rellenar_vector(var vector: t_vector; var dimL: int16; dimF: int16);
var
  num: int16;
begin
  textcolor(green); write('Introducir número entero para la posición ', dimL+1, ' del
vector: ');
  textcolor(yellow); readln(num);
  while ((num<>vector_salida) and (dimL<dimF)) do
  begin
    dimL:=dimL+1;
    vector[dimL]:=num;
    textcolor(green); write('Introducir número entero para la posición ', dimL+1, ' del
vector: ');
    textcolor(yellow); readln(num);
  end;
end;
procedure intercambio(var vector: t_vector; pos_max, pos_min: int16);
var
  num_aux: int16;
begin
  num_aux:=vector[pos_max];
  vector[pos_max]:=vector[pos_min];
  vector[pos_min]:=num_aux;
end;
procedure elementosMaximoYMinimo(vector: t_vector; dimL: int16; var ele_max, ele_min,
pos_max, pos_min: int16);
var
  i: int16;
begin
  for i:= 1 to dimL do
  begin
    if (vector[i]>ele_max) then
    begin
      ele_max:=vector[i];
      pos_max:=i;
    end;
    if (vector[i]<ele_min) then
    begin
      ele_min:=vector[i];
      pos_min:=i;
    end;
  end;
end;
var
  vector: t_vector;
  dimL, ele_max, ele_min, pos_max, pos_min: int16;
begin
  dimL:=0; ele_max:=low(int16); ele_min:=high(int16); pos_max:=0; pos_min:=0;
  rellenar_vector(vector,dimL,dimF);
  elementosMaximoYMinimo(vector,dimL,ele_max,ele_min,pos_max,pos_min);
```

```
  intercambio(vector,pos_max,pos_min);
  textcolor(green); write('El elemnento máximo '); textcolor(red); write(ele_max);
textcolor(green); write(', que se encontraba en la posición '); textcolor(red);
write(pos_max); textcolor(green); write(', fue intercambiado con el elemento mínimo ');
textcolor(red); write(ele_min); textcolor(green); write(', que se encontraba en la posición
'); textcolor(red); write(pos_min);
end.
```

Ejercicio 7.

Juan Menduiña

Trabajo Práctico N° 5:
Punteros.

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)

Ejercicio 1.

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

```
program TP5_E1;
{$codepage UTF8}
uses crt;
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.
```

Instrucciones	pc	pc^	write
`writeln(sizeof(pc), ' bytes');`			4 bytes
`new(pc);`	XXX		
`writeln(sizeof(pc), ' bytes');`			4 bytes
`pc^:='un nuevo nombre';`		‘un nuevo nombre’	
`writeln(sizeof(pc), ' bytes');`			4 bytes
`writeln(sizeof(pc^), ' bytes');`			51 bytes
`pc^:='otro nuevo nombre distinto al anterior';`		‘otro nuevo nombre distinto al anterior’	
`writeln(sizeof(pc^), ' bytes');`			51 bytes

Ejercicio 2.

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

```
program TP5_E2;
{$codepage UTF8}
uses crt;
type
  cadena=string[9];
  producto=record
    codigo: integer;
    descripcion: cadena;
    precio: real;
  end;
  puntero_producto=^producto;
var
  p: puntero_producto;
  prod: producto;
begin
  writeln(sizeof(p), ' bytes');
  writeln(sizeof(prod), ' bytes');
  new(p);
  writeln(sizeof(p), ' bytes');
  p^.codigo:=1;
  p^.descripcion:='nuevo producto';
  writeln(sizeof(p^), ' bytes');
  p^.precio:=200;
  writeln(sizeof(p^), ' bytes');
  prod.codigo:=2;
  prod.descripcion:='otro nuevo producto';
  writeln(sizeof(prod), ' bytes');
end.
```

Instrucciones	p	p^	prod	write
`writeln(sizeof(p), ' bytes');`				4 bytes
`writeln(sizeof(prod), ' bytes');`				24 bytes
`new(p);`	XXX			
`writeln(sizeof(p), ' bytes');`				4 bytes
`p^.codigo:=1;`		.codigo=1		
`p^.descripcion:='nuevo producto';`		.descripcion=’nuevo pro’		warming
`writeln(sizeof(p^), ' bytes');`				24 bytes
`p^.precio:=200;`		.precio=200		
`writeln(sizeof(p^), ' bytes');`				24 bytes
`prod.codigo:=2;`			.codigo=2	
`prod.descripcion:='otro nuevo producto';`			.descripcion=’otro nuev’	warning
`writeln(sizeof(prod), ' bytes');`				24 bytes

Ejercicio 3.

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

```
program TP5_E3;
{$codepage UTF8}
uses crt;
type
  numeros=array[1..10000] of integer;
  puntero_numeros=^numeros;
var
  n: puntero_numeros;
  num: numeros;
  i: integer;
begin
  writeln(sizeof(n), ' bytes');
  writeln(sizeof(num), ' bytes');
  new(n);
  writeln(sizeof(n^), ' bytes');
  for i:= 1 to 5000 do
    n^[i]:=i;
  writeln(sizeof(n^), ' bytes');
end.
```

Instrucciones	n	n^	num	i	write
`writeln(sizeof(n), ' bytes');`					4 bytes
`writeln(sizeof(num), ' bytes');`					20000 bytes
`new(n);`	XXX				
`writeln(sizeof(n^), ' bytes');`					20000 bytes
`for i:= 1 to 5000 do n^[i]:=i;`			n^[1..5000]=[1..5000]	[1..5000]	
`writeln(sizeof(n^), ' bytes');`					20000 bytes

Ejercicio 4.

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

(a)

```
program TP5_E4a;
{$codepage UTF8}
uses crt;
type
  cadena=string[50];
  puntero_cadena=^cadena;
var
  pc: puntero_cadena;
begin
  pc^:='un nuevo texto';
  new(pc);
  writeln(pc^);
end.
```

Instrucciones	pc	pc^	write
`pc^:='un nuevo texto';`		error	
`new(pc);`	XXX		
`writeln(pc^);`			basura

(b)

```
program TP5_E4b;
{$codepage UTF8}
uses crt;
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.
```

Instrucciones	pc	pc^	write
`new(pc);`	XXX		
`pc^:='un nuevo nombre';`		‘un nuevo nombre’	
`writeln(sizeof(pc^), ' bytes');`			51 bytes
`writeln(pc^);`			un nuevo nombre
`dispose(pc);`	elimina puntero y libera memoria		
`pc^:='otro nuevo nombre';`		error	

(c)

```
program TP5_E4c;
{$codepage UTF8}
uses crt;
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.
```

Instrucciones	pc	pc^	write
`new(pc);`	XXX		
`pc^:='Un texto';`		‘Un texto’	
`writeln(pc^);`			Un texto
`cambiarTexto(pc);`		‘Otro texto’	
`writeln(pc^);`			Otro texto

(d)

```
program TP5_E4d;
{$codepage UTF8}
uses crt;
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.
```

Instrucciones	pc	pc^	copia pc	copia pc^	write
`new(pc);`	XXX				
`pc^:='Un texto';`		‘Un texto’			

`writeln(pc^);`					Un texto
`cambiarTexto(pc);`			YYY	‘Otro texto’	
`writeln(pc^);`					Un texto

Ejercicio 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 TP5_E5;
{$codepage UTF8}
uses crt;
type
  TEmpleado=record
    sucursal: char;
    apellido: string[25];
    correoElectronico: 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^.correoElectronico,PtrEmpleado^.sueldo);
  {¿Cuánta memoria disponible hay ahora? (IV)}
  dispose(PtrEmpleado);
  {¿Cuánta memoria disponible hay ahora? (V)}
end.
```

Instrucciones	Memoria
`readln(alguien.apellido);`	400.000 bytes
`new(PtrEmpleado);`	399.924 bytes (400.000 - 76)
`readln(PtrEmpleado^.Sucursal,PtrEmpleado^.apellido);`	399.924 bytes
`readln(PtrEmpleado^.correoElectrónico,PtrEmpleado^.sueldo);`	399.924 bytes
`dispose(PtrEmpleado);`	400.000 bytes (399.924 + 76)

Ejercicio 6.

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

(a) *El programa debe utilizar sólo memoria estática.*

```
program TP5_E6a;
{$codepage UTF8}
uses crt;
const
  KB=50; bytes=51200; vector_total=12800;
type
  t_vector=array[1..vector_total] of int32;
var
  vector: t_vector;
begin
  textcolor(green); write('La memoria estática ocupada por vector es '); textcolor(red);
write(sizeof(vector)/1024:0:2); textcolor(green); writeln(' KB');
end.
```

(b) *El programa debe utilizar el 50% de memoria estática y el 50% de memoria dinámica.*

```
program TP5_E6b;
{$codepage UTF8}
uses crt;
const
  KB=50; bytes=51200; vector_total=6399;
type
  t_vector=array[1..vector_total] of int32;
  t_registro_vector=record
    completa: int32;
    vector: t_vector;
  end;
  t_puntero_registro=^t_registro_vector;
var
  vector: t_vector;
  puntero_registro: t_puntero_registro;
begin
  new(puntero_registro);
  textcolor(green); write('La memoria estática ocupada por vector es '); textcolor(red);
write(sizeof(vector)/1024:0:2); textcolor(green); writeln(' KB');
  textcolor(green); write('La memoria estática ocupada por puntero_registro es ');
textcolor(red); write(sizeof(puntero_registro)/1024:0:2); textcolor(green); writeln(' KB');
  textcolor(green); write('La memoria dinámica ocupada por contenido puntero_registro es ');
textcolor(red); write(sizeof(puntero_registro^)/1024:0:2); textcolor(green); write(' KB');
end.
```

(c) *El programa debe minimizar tanto como sea posible el uso de la memoria estática (a lo sumo, 4 bytes).*

```
program TP5_E6c;
{$codepage UTF8}
uses crt;
const
  KB=50; bytes=51200; vector_total=12799;
```

```
type
  t_vector=array[1..vector_total] of int32;
  t_puntero_vector=^t_vector;
var
  puntero_vector: t_puntero_vector;
begin
  new(puntero_vector);
  textcolor(green); write('La memoria estática ocupada por puntero_vector es ');
textcolor(red); write(sizeof(puntero_vector)/1024:0:2); textcolor(green); writeln(' KB');
  textcolor(green); write('La memoria dinámica ocupada por contenido puntero_vector es ');
textcolor(red); write(sizeof(puntero_vector^)/1024:0:2); textcolor(green); write(' KB');
end.
```

Ejercicio 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.*

```
program TP5_E7a;
{$codepage UTF8}
uses crt;
const
  longitud_ciudad=50; ciudades_total=2500;
type
  t_ciudad=string[longitud_ciudad];
  t_vector_ciudad=array[1..ciudades_total] of t_ciudad;
var
  memoria: int32;
begin
  memoria:=sizeof(t_vector_ciudad);
  textcolor(green); write('El tamaño de memoria que requiere la estructura es ');
textcolor(red); write(memoria); textcolor(green); write(' bytes');
end.
```

(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;

(i) *Indicar cuál es el tamaño de la variable Punteros al comenzar el programa.*

El tamaño de la variable Punteros al comenzar el programa es 10.000 bytes.

(ii) *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?*

```
procedure reservar_memoria(var punteros: t_vector_ciudad);
var
  i: int16;
begin
  for i:= 1 to ciudades_total do
    new(punteros[i]);
end;
```

La cantidad de memoria reservada después de ejecutar el módulo es 127.500 bytes, la cual corresponde a memoria dinámica.

(iii) *Escribir un módulo para leer los nombres y almacenarlos en la estructura de la variable Punteros.*

```
procedure leer_ciudades(var vector_ciudad: t_vector_ciudad);
var
  i: int16;
begin
  for i:= 1 to ciudades_total do
  begin
    textcolor(green); write('Introducir nombre de ciudad ', i, ': ');
    textcolor(yellow); readln(vector_ciudad[i]^);
  end;
end;
```

```
program TP5_E7b;
{$codepage UTF8}
uses crt;
const
  longitud_ciudad=50; ciudades_total=2500;
type
  t_ciudad=string[longitud_ciudad];
  t_puntero_ciudad=^t_ciudad;
  t_vector_ciudad=array[1..ciudades_total] of t_puntero_ciudad;
{ii}
procedure reservar_memoria(var vector_ciudad: t_vector_ciudad);
var
  i: int16;
begin
  for i:= 1 to ciudades_total do
    new(vector_ciudad[i]);
end;
{iii}
procedure leer_ciudades(var vector_ciudad: t_vector_ciudad);
var
  i: int16;
begin
  for i:= 1 to ciudades_total do
  begin
    textcolor(green); write('Introducir nombre de ciudad ', i, ': ');
    textcolor(yellow); readln(vector_ciudad[i]^);
  end;
end;
var
  vector_ciudad: t_vector_ciudad;
  i: int16;
begin
  for i:= 1 to ciudades_total do
    vector_ciudad[i]:=nil;
  textcolor(green); write('El tamaño de la variable vector_ciudad es '); textcolor(red);
write(sizeof(vector_ciudad)); textcolor(green); writeln(' bytes');
  textcolor(green); write('El tamaño del contenido de la variable vector_ciudad es ');
textcolor(red); write(sizeof(vector_ciudad[1]^)*length(vector_ciudad)); textcolor(green);
writeln(' bytes');
  reservar_memoria(vector_ciudad);
  textcolor(green); write('El tamaño de la variable vector_ciudad es '); textcolor(red);
write(sizeof(vector_ciudad)); textcolor(green); writeln(' bytes');
```

```
  textcolor(green); write('El tamaño del contenido de la variable vector_ciudad es ');
textcolor(red); write(sizeof(vector_ciudad[1]^)*length(vector_ciudad)); textcolor(green);
writeln(' bytes');
  leer_ciudades(vector_ciudad);
  textcolor(green); write('El tamaño de la variable vector_ciudad es '); textcolor(red);
write(sizeof(vector_ciudad)); textcolor(green); writeln(' bytes');
  textcolor(green); write('El tamaño del contenido de la variable vector_ciudad es ');
textcolor(red); write(sizeof(vector_ciudad[1]^)*length(vector_ciudad)); textcolor(green);
write(' bytes');
end.
```

Ejercicio 8.

Analizar el siguiente programa:

```
program TP5_E8;
{$codepage UTF8}
uses crt;
type
  datos=array[1..20] of integer;
  punt=^datos;
procedure procesarDatos(v: datos; var v2: datos);
var
  i, j: integer;
begin
  for i:= 1 to 20 do
    v2[21-i]:=v[i];
end;
var
  vect: datos;
  pvect: punt;
  i: integer;
begin
  for i:= 1 to 20 do
    vect[i]:=i;
    new(pvect);
  for i:= 20 downto 1 do
    pvect^[i]:=0;
  procesarDatos(pvect^,vect);
  writeln('fin');
end.
```

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

Hasta sentencia de la línea	Memoria estática	Memoria dinámica	Memoria total
(a) 18	46 bytes	0 bytes	46 bytes
(b) 20	46 bytes	0 bytes	46 bytes
(c) 23	46 bytes	40 bytes	86 bytes
(d) 11	0 bytes	0 bytes	0 bytes
(e) 25	46 bytes	40 bytes	86 bytes

Aclaración: Hasta la sentencia de la línea 24, tenemos 88 bytes en memoria dinámica, ya que se suman 40 bytes por el parámetro por valor, 4 bytes por el parámetro por referencia y 4 bytes por las variables locales al procedure “procesarDatos”.

Juan Menduiña

Trabajo Práctico N° 7:
Repaso.

Ejercicio 1.

Una productora nacional realiza un casting de personas para la selección de actores extras de una nueva película, para ello se debe leer y almacenar la información de las personas que desean participar de dicho casting. De cada persona, se lee: DNI, apellido y nombre, edad y el código de género de actuación que prefiere (1: drama, 2: romántico, 3: acción, 4: suspenso, 5: terror). La lectura finaliza cuando llega una persona con DNI 33.555.444, la cual debe procesarse. Una vez finalizada la lectura de todas las personas, se pide:

- *Informar la cantidad de personas cuyo DNI contiene más dígitos pares que impares.*
- *Informar los dos códigos de género más elegidos.*
- *Realizar un módulo que reciba un DNI, lo busque y lo elimine de la estructura. El DNI puede no existir. Invocar dicho módulo en el programa principal.*

```
program TP7_E1;
{$codepage UTF8}
uses crt;
const
  codigo_ini=1; codigo_fin=5; digito_ini=0; digito_fin=9; dni_salida=33555444;
type
  t_str20=string[20];
  t_codigos=0..codigo_fin;
  t_registro_persona=record
    dni: int32;
    apellido: t_str20;
    nombre: t_str20;
    edad: int16;
    codigo: t_codigos;
  end;
  t_vector_dni=array[digito_ini..digito_fin] of int8;
  t_vector_codigos=array[codigo_ini..codigo_fin] of int16;
  t_lista_personas=^t_nodo_personas;
  t_nodo_personas=record
    ele: t_registro_persona;
    sig: t_lista_personas;
  end;
procedure leer_persona(var registro_persona: t_registro_persona);
begin
  textcolor(green); write('Introducir DNI de la persona: ');
  textcolor(yellow); readln(registro_persona.dni);
  textcolor(green); write('Introducir apellido de la persona: ');
  textcolor(yellow); readln(registro_persona.apellido);
  textcolor(green); write('Introducir nombre de la persona: ');
  textcolor(yellow); readln(registro_persona.nombre);
  textcolor(green); write('Introducir edad de la persona: ');
  textcolor(yellow); readln(registro_persona.edad);
  textcolor(green); write('Introducir código de género de actuación de la persona (1: drama,
2: romántico, 3: acción, 4: suspenso, 5: terror): ');
  textcolor(yellow); readln(registro_persona.codigo);
end;
procedure agregar_adelante_lista(var lista_personas: t_lista_personas; registro_persona:
t_registro_persona);
var
  lista_nueva: t_lista_personas;
begin
```

```
  new(lista_nueva); lista_nueva^.ele:=registro_persona; lista_nueva^.sig:=nil;
  if (lista_personas=nil) then
    lista_personas:=lista_nueva
  else
  begin
    lista_nueva^.sig:=lista_personas;
    lista_personas:=lista_nueva;
  end;
end;
procedure descomponer_dni(var vector_dni: t_vector_dni; registro_persona:
t_registro_persona);
var
  digito: int8;
begin
  while (registro_persona.dni<>0) do
  begin
    digito:=registro_persona.dni mod 10;
    vector_dni[digito]:=vector_dni[digito]+1;
    registro_persona.dni:=registro_persona.dni div 10;
  end;
end;
function contar_pares_impares(vector_dni: t_vector_dni): boolean;
var
  i, pares, impares: int8;
begin
  pares:=0; impares:=0;
  for i:= digito_ini to digito_fin do
    if (vector_dni[i]<>0) then
      if (i mod 2=0) then
        pares:=pares+vector_dni[i]
      else
        impares:=impares+vector_dni[i];
  contar_pares_impares:=(pares>impares);
end;
procedure actualizar_maximos(vector_codigos: t_vector_codigos; var codigo_max1, codigo_max2:
t_codigos);
var
  i: int8;
  num_max1, num_max2: int16;
begin
  num_max1:=low(int16); num_max2:=low(int16);
  for i:= codigo_ini to codigo_fin do
  begin
    if (vector_codigos[i]>num_max1) then
    begin
      num_max2:=num_max1;
      codigo_max2:=codigo_max1;
      num_max1:=vector_codigos[i];
      codigo_max1:=i;
    end
    else
      if (vector_codigos[i]>num_max2) then
      begin
        num_max2:=vector_codigos[i];
        codigo_max2:=i;
      end;
  end;
end;
procedure leer_personas(var lista_personas: t_lista_personas; var personas_par: int16; var
codigo_max1, codigo_max2: t_codigos);
var
  registro_persona: t_registro_persona;
  vector_dni: t_vector_dni;
  vector_codigos: t_vector_codigos;
  i: int8;
begin
```

```
  for i:= codigo_ini to codigo_fin do
    vector_codigos[i]:=0;
  repeat
    leer_persona(registro_persona);
    agregar_adelante_lista(lista_personas,registro_persona);
    for i:= digito_ini to digito_fin do
      vector_dni[i]:=0;
    descomponer_dni(vector_dni,registro_persona);
    if (contar_pares_impares(vector_dni)=true) then
      personas_par:=personas_par+1;
    vector_codigos[registro_persona.codigo]:=vector_codigos[registro_persona.codigo]+1;
  until (registro_persona.dni=dni_salida);
  actualizar_maximos(vector_codigos,codigo_max1,codigo_max2);
end;
procedure eliminar_dni(var lista_personas: t_lista_personas; dni_eliminar: int32);
var
  actual, anterior: t_lista_personas;
begin
  actual:=lista_personas;
  while ((actual<>nil) and (actual^.ele.dni<>dni_eliminar)) do
  begin
    anterior:=actual;
    actual:=actual^.sig;
  end;
  if (actual<>nil) then
  begin
    if (actual=anterior) then
      lista_personas:=lista_personas^.sig
    else
      anterior^.sig:=actual^.sig;
    textcolor(green); write('El DNI '); textcolor(yellow); write(dni_eliminar);
textcolor(red); write(' SÍ'); textcolor(green); write(' fue encontrado y eliminado');
    dispose(actual);
  end
  else
  begin
    textcolor(green); write('El DNI '); textcolor(yellow); write(dni_eliminar);
textcolor(red); write(' NO'); textcolor(green); write(' fue encontrado y eliminado');
  end;
end;
var
  lista_personas: t_lista_personas;
  codigo_max1, codigo_max2: t_codigos;
  personas_par: int16;
  dni_eliminar: int32;
begin
  lista_personas:=nil;
  codigo_max1:=low(t_codigos); codigo_max2:=low(t_codigos); personas_par:=0;
  leer_personas(lista_personas,personas_par,codigo_max1,codigo_max2);
  textcolor(green); write('La cantidad de personas cuyo DNI contiene más dígitos pares que
impares es '); textcolor(red); writeln(personas_par);
  textcolor(green); write('Los dos códigos de género más elegidos son '); textcolor(red);
write(codigo_max1); textcolor(green); write(' y '); textcolor(red); writeln(codigo_max2);
  textcolor(green); write('Introducir DNI que se desea eliminar: ');
  textcolor(yellow); readln(dni_eliminar);
  eliminar_dni(lista_personas,dni_eliminar);
  dispose(lista_personas);
end.
```

Ejercicio 2.