

Aclaración: los ejercicios marcados con * se recomiendan realizar en forma obligatoria durante la semana correspondiente a la realización de la práctica, acorde a lo estipulado en el cronograma. Además, se recomienda consultar la solución realizada con los ayudantes durante la práctica y de ser posible, escribir el programa en Lazarus Pascal y probar su ejecución. El resto de los ejercicios es necesario realizarlos como parte del estudio y preparación para el parcial.

Objetivos de la práctica:

Se espera que el alumno logre:

- Reconocer la importancia de la modularización como estrategia en la resolución de problemas.
- Distinguir entre los dos tipos de módulos en Pascal (funciones y procedimientos), a partir de poder identificar cuál es más conveniente en cada problema.
- Comprender y Aplicar el mecanismo de comunicación de pasaje de parámetros en la resolución de problemas.
- Ensayar la utilización de la estructura de control CASE en problemas en los que resulte conveniente su uso.
- Utilizar el tipo de datos conjunto en la resolución de problemas vinculados al manejo de caracteres.

PARTE A

1. * Escriba un módulo que reciba un número entero y devuelva los dos dígitos más chicos que contiene dicho número. Elija un nombre significativo para dicho módulo. Analice con sus compañeros y justifiquen cuándo es conveniente utilizar una función o un procedimiento.
2. Implemente los siguientes módulos:
 - a) uno que reciba un entero y devuelva dos enteros con la cantidad de dígitos pares e impares que contiene.
 - b) uno que cuente la cantidad de dígitos de un número, utilizando el módulo del inciso a).
 - c) uno que informe el porcentaje de números impares de un entero, utilizando el módulo del inciso a).
3. * Escriba un módulo Max4 que reciba cuatro enteros y retorne el mayor:
 - a) Implemente con una función.
 - b) Implemente la función Max, que recibe 2 enteros y retorna el mayor, y re-implemente la función de a) utilizando Max.
 - c) Compare y reflexione acerca de las implementaciones de a) y b). Piense en la modularización, la expresividad y la legibilidad.
4. Escriba un módulo que reciba un entero y retorne si es capicúa (mismo número de izquierda a derecha que de derecha a izquierda).
5. Implemente un módulo que reciba un número y retorne verdadero si todos sus dígitos son primos.
 - a) Modularice la solución implementando un módulo auxiliar que reciba un dígito (un número entre 0 y 9) y retorne verdadero si es un número primo, o falso en caso contrario.
 - b) Utilice este módulo auxiliar para verificar si todos los dígitos del número ingresado son primos.
6. * a) Implemente un módulo que permita imprimir los últimos dígitos de un número en orden inverso. El módulo debe recibir el número y la cantidad de dígitos a imprimir. Si recibe el número 1954 y la cantidad 3, debe imprimir 4 - 5 - 9.
* b) Escriba un programa que lea números enteros por teclado hasta que llegue el número 0. Utilice el módulo implementado en a) para imprimir los últimos 3 y 5 dígitos de cada número ingresado.
7. Escriba un programa que lea una secuencia de caracteres terminada en '*' y procese palabras analizando si su longitud es exactamente 7. El programa debe informar cuántas palabras de longitud 7 encontró. El procesamiento de cada palabra debe ser realizado en un módulo. Puede haber blancos al principio y al final de la secuencia.

8. * a) Escriba un procedimiento que lea la edad de una cantidad de personas y devuelva el promedio de estas. La cantidad de personas se recibe como parámetro.
- b) Escriba un programa que procese la edad de 25 personas utilizando el módulo desarrollado en a) e informe el resultado.
9. Dado el siguiente código, completar los módulos con los cálculos pedidos. Luego realizar un módulo que lea 10 números reales e informe el promedio entre todos los números leídos y el porcentaje de números mayores de 50 entre todos los leídos. Debe utilizar los módulos que completó previamente.

<pre> program CalculadoraBasica; //Módulo para sumar dos números function Sumar(a, b: real): real; begin //Completar código end; //Módulo para restar 2 números function Restar(a, b: real): real; begin // Completar código end; // Módulo para multiplicar 2 números function Multiplicar(a, b: real): real; begin // Completar código end; // Módulo para dividir 2 números function Dividir(a, b: real): real; begin // Completar código end; </pre>	<pre> //Programa principal var opcion: char; num1, num2, resultado: real; begin writeln('Calculadora: Operaciones Básicas'); writeln('Ingrese el primer número: '); readln(num1); writeln('Ingrese el segundo número: '); readln(num2); writeln('Seleccione la operación:'); writeln('a) Suma, b) Resta, c) Multiplicación, d) División'); readln(opcion); case opcion of 'a': resultado := Sumar(num1, num2); 'b': resultado := Restar(num1, num2); 'c': resultado := Multiplicar(num1, num2); 'd': resultado := Dividir(num1, num2); else resultado := -1; end; writeln('El resultado es: ', resultado); end. </pre>
--	--

10. * Escriba un programa que lea una secuencia de caracteres terminada en punto, y que a través de un procedimiento evalúe si cada una de sus palabras tiene la 'p' seguida de la 'a'. El programa debe informar cuántas palabras cumplen con esa condición.
11. * Dado el siguiente programa: informar que imprime en cada caso.

```

program ejercicio;

var alfa, beta, gama, epsilon: integer;

procedure calcular(alfa: integer; var gama: integer; var beta:integer;
                    var epsilon: integer)
begin
    alfa:= beta - 1 ;
    beta:= alfa + 8;
    gama:= beta + 15;
    epsilon:= beta - gama;
    write(alfa); write(beta); write(gama); write(epsilon);
end;

begin
    alfa:= 6; beta:= 13; gama:= -6; epsilon:= 2;
    calcular(epsilon, alfa, beta, gama);
    write(alfa); write(beta); write(gama); write(epsilon);
end.

```

12. Dada la siguiente función marque las invocaciones a dicha función que considere válidas:

```
function cuadrado(x:integer): integer;
begin
    cuadrado:= x*x;
end
```

- a) Write(cuadrado(5));
- b) c:= cuadrado(5); Write(cuadrado);
- c) If (cuadrado = 25) then
 Write('5*5=25');
- d) cuadrado(5);

- e) c:= cuadrado(5); Write (c);
- f) cuadrado(5, c); Write (c);
- g) If (cuadrado(5) = 25) then
 Write('5*5=25');

13. Complete y optimice el siguiente código para que compile correctamente y asegure claridad, sencillez y mantenibilidad.

Una empresa tiene clientes de los cuales se lee desde teclado su número de cliente, y se desea asignarles un vendedor personalizado. La empresa cuenta con 9 vendedores. Para dividir el trabajo se decidió que a partir de descomponer el número de cliente sumando sus dígitos se debe obtener el número de vendedor. Por ejemplo: el cliente con número/código 9283 --> 22 -> 4 (se le asigna el vendedor 4).

Sin embargo, es necesario hacer una última comprobación antes de asignar vendedor al cliente: el número de cliente debe ser divisible por el número de vendedor. Si no es divisible existe una función "generarRandom" el cual genera un número aleatorio de 1 a 100 que se suma al número de vendedor, una vez obtenido, se debe descomponer (como se hizo con el número de cliente anteriormente) y éste da como resultado el vendedor definitivo.

Completar el programa informando para cada número de cliente que se lee desde teclado cuál es su vendedor asignado y, además, en caso de haber sido reasignado, indicar cuál era el vendedor originalmente asignado. **Nota: se leen números de cliente hasta leer el -1**

```
program ElegirVendedor;

function generarRandom():integer;
begin
    randomize;
    generarRandom:=random(100)+1;
end;

function esDivisible(dividendo,
divisor:integer): boolean;
Begin
    esDivisible:=(dividendo mod divisor
= 0);
end;

procedure reasignar(vendedor: integer);
var
    numero,digito,suma:integer;
begin
    numero:=vendedor+generarRandom();
    while(numero>9) do begin
        suma:=0;
        while(numero > 0) do begin
            digito:=numero mod 10;
            {Completar}
            suma:=suma+digito;
        end;
        numero:=suma;
    end;
    vendedor:=numero;
end;
```

```
var numero,vendedor,código:integer;
    suma,digito:integer;
begin
    writeln('Ingrese nro de cliente');
    readln(codigo);
    while(codigo <> -1) do begin
        numero:=codigo;
        while(numero>9) do begin
            suma:=0;
            while(numero > 0) do begin
                digito:=numero mod 10;
                {Completar}
                suma:=suma+digito;
            end;
            numero:=suma;
        end;
        if(not esDivisible(numero, vendedor))
        then begin
            writeln('Al cliente ', codigo,
' le correspondía el vendedor ',
vendedor);
        end;
        writeln('Al cliente ',codigo,
' le corresponde el vendedor ',
vendedor);

        readln(codigo);
    end;
end.
```

PARTE B

1. *a) Escriba un módulo que reciba 2 números enteros i y n , y calcule la potencia enésima de i (i^n).
 b) Escriba un programa que invoque el módulo de a) para que calcule el cuadrado de un número i (i^2), el cubo de un número i (i^3) y la potencia enésima de 2 (2^n).
2. *a) El factorial de un número n se expresa como $n!$ y se define como el producto de todos los números desde 1 hasta n . Por ejemplo, el factorial de 6 o $6!$ equivale a $6*5!$ que a $1*2*3*4*5*6$ que equivale a 720. Escriba una función que reciba un número n y retorne su factorial.
 b) Un número combinatorio (m,n) expresa todas las combinaciones de m elementos agrupados de n grupos. La expresión numérica de un número combinatorio es la siguiente:

$$\binom{m}{n} = \frac{m!}{(m-n)! * n!}$$

Utilizando la función factorial, escriba una función que calcule el número combinatorio (m, n) .

3. * Dado el siguiente programa, informar qué imprime en cada caso.

```
Program Uno;
var pri, cuar: integer;
procedure DatosDos( pri: integer; var cuar: integer);
begin
  pri := (pri + 8) * cuar;
  cuar:= pri + cuar;
  write(cuar);
end;
procedure DatosUno( var pri: integer; cuar: integer);
begin
  cuar:= cuar + ((pri * 2) + 3);
  if ( cuar < 6) then
    datosDos(cuar, pri)
  else Begin
    cuar:= 4;
    datosDos(cuar, pri);
  end;
  write(pri, cuar);
end;
begin
  pri:= 4; cuar:= 8;
  datosUno(cuar, pri);
  write(pri, cuar);
end.
```

4. *Escriba un programa que lea una secuencia de caracteres terminada en '#' e informe aquellas letras entre la "a" y la "z" que no fueron ingresadas. Implemente un módulo que contabilice las letras y otro que imprima el resultado.
5. *a) Implemente un módulo que lea una secuencia de caracteres que representan una palabra (termina con blanco o asterisco), y retorne la cantidad de consonantes y vocales de dicha palabra.
 b) Utilizando el módulo implementado en a) realice un programa que procese una secuencia de caracteres terminada en '*', e informe la cantidad de consonantes y vocales para cada una de sus palabras y la posición de las palabras (orden en el que fue ingresada) con mayor cantidad de consonantes y vocales.
6. * Se lee una secuencia de caracteres terminada en '.'. Determinar si la secuencia cumple con el patrón **A@B**. En caso de no cumplir, informar las partes que no verificaron el patrón.

A@B. donde:

@ es el carácter '@' que seguro existe.

A debe ser una secuencia de letras mayúsculas.

B debe ser una secuencia de caracteres que no aparecieron en **A**.

Ejemplo: la siguiente secuencia cumple el patrón `MTL@aePsz`.

7. Se lee una secuencia de caracteres terminada en `''`. Informar si la secuencia cumple con el patrón: `V&Q%W`. En caso de no ser así terminar de procesar e informar en qué subsecuencia se dejó de cumplir el patrón. Se sabe que:

& es el carácter `'&'` y **%** es el carácter `'%'` que seguro existen.

V es una secuencia de palabras, donde todas las palabras comienzan con la letra `'o'` y terminan con una la letra `'n'`.

Q es una secuencia de palabras, donde todas las palabras tienen todas las vocales.

W es una secuencia de palabras donde todas las palabras de longitud mayor que 5, tienen tres `'s'`.

Ejemplo. La siguiente secuencia cumple con el patrón:

`ocasion operacion ocupacion& euforia cautiverio ecuacion% sucesorias suspensorio casa*`