Trabajo Práctico 6
Funciones, Procedimientos y División de Problemas

Ejercicio 1: Considere definida la función Invertir.

```
function Invertir( num: integer ):integer;
{Objetivo: Invierte el orden de los dígitos de un número entero
 Entrada: el parámetro "Num" recibe el número que se desea invertir.
 Salida: Retornará un entero con los dígitos de "Num" en orden inverso
 Ejemplo: si "num" es 1234 retornará 4321}
var
   inv, ultimo: integer;
begin
   inv := 0; {variable auxiliar en la que se construye el "invertido"}
   while num > 0 do
   begin {voy sacando dígitos de Num del menos al más significativo}
       ultimo := num mod 10;
      num := num div 10;
      inv := inv * 10 + ultimo; {voy agregando al final del invertido}
   end;
   Invertir := inv;
end;
```

Realice un programa que, utilizando la función dada, determine si un número ingresado por el usuario es o no capicúa. Por ejemplo,
si Num = 12321, el programa deberá mostrar por pantalla "El número 12321 ES CAPICUA".
si Num = 2343, el programa deberá mostrar por pantalla "El número 2343 NO ES CAPICUA"

Ejercicio 2: Implemente las siguientes funciones

function *factorial*(Nro: integer): integer; { Calcula el factorial de un número Nro }

function *potencia*(Base, Exponente: integer): integer; { Calcula $Base^{Exponente}$ }

function *primo*(Nro: integer):boolean; { Determina si Nro es primo }.

function *techo*(Nro: real): integer; { Calcula $\lceil Nro \rceil$ }
{La función techo se aplica a un número real Nro y devuelve el mínimo número entero k mayor o igual a Nro ejemplos: techo(2.4) es 3 y techo(2.0) es 2}

Ejercicio 3: Escriba un programa en Pascal que lea un par de números n y e, y muestre por pantalla todos los números primos comprendidos entre 1 y n^e (*n elevado a la e*). <u>Nota</u>*: puede utilizar primitivas de ejercicios anteriores.*

Por ejemplo: para *n* = 2 y e=5 (donde n^e es 32), el programa deberá mostrar por pantalla:
`Los números primos entre 1 y 32 son: 2 3 5 7 11 13 17 19 23 29 31`

Ejercicio 4: Escriba un programa en Pascal dividiendo el problema en subproblemas, que elimine de un archivo de enteros todas las componentes que sean **primos <u>o</u>** sean **capicúas**.

Ejercicio 5: Suponga que cuenta con un archivo A de números enteros ya ingresados. Para cada uno de los siguientes incisos se desea generar otro archivo nuevo con los elementos de A que cumplan lo indicado:

a) Sean capicúas y tengan una cantidad impar de dígitos.

b) Sean primos o tengan todos los dígitos impares.

c) Tengan una cantidad par de dígitos, no sean capicúas y tengan al menos un dígito par.

Para cada uno de los incisos anteriores (en forma individual) se solicita que:

i) Divida el problema en subproblemas y haga un gráfico o esquema de su propuesta de diseño para la solución.

ii) Describa las funciones y procedimientos necesarios identificando los parámetros de entrada y salida, agregando una breve descripción del objetivo de la primitiva.

iii) Realice un programa en PASCAL que resuelva el problema. No es necesario implementar las primitivas, simplemente deberá declarar los encabezados de cada una.

Ejercicio 6: Explique la diferencia que hay entre parámetros por valor y por referencia en Pascal. Indique cuantos parámetros por valor y cuantos por referencia hay en cada uno de los siguientes procedimientos y funciones:

1. **PROCEDURE** *Eje1*(**var** letra1,letra2:char; N1,N2:integer; **var** Error:boolean);
2. **PROCEDURE** *Eje2*(**var** A:char; **var** b:integer; **var** c:boolean);
3. **FUNCTION** *F1*(a,b:integer; es: boolean):real;
4. **FUNCTION** *LeeLetra*: CHAR;
5. **FUNCTION** *LeeNumero*(l:char; **var** error:boolean):integer;

Ejercicio 7:

(a) Indique las diferentes opciones que existen para los parámetros efectivos cuando se corresponden con un parámetro formal por valor o por un parámetro formal por referencia.

(b) Analizar cuáles de las invocaciones a procedimientos o funciones detalladas a continuación son correctas en base a lo indicado en el inciso (a) y a las siguientes declaraciones:

```
VAR     w: Char;
        x: Integer;
        y: Real;
        z: Boolean;

PROCEDURE Proc1 (a,b: Integer; var c: Char);
BEGIN ... END;

FUNCTION Funcion1 (x: char):Real;
BEGIN ... END;

FUNCTION Funcion2 (VAR a: Real; b: Boolean):Integer;
BEGIN ... END;
```

1. **Proc1(7, y, w);**
2. **Proc1(7, y, c);**
3. **Proc1(27, x, w, w);**
4. **Proc1(2.4, 5+8, w);**
5. **Proc1(7, 5, 'c');**
6. **x:= Funcion1(w);**
7. **y:= Funcion1(w);**
8. **y:= Funcion1('x');**
9. **Funcion1(w);**
10. **x := Funcion2(y, false);**
11. **y := Funcion2(y, true);**
12. **x := Funcion2(3+5, z);**
13. **x := Funcion2(3.5+y, z);**

Ejercicio 8: En una competencia por internet fueron seleccionados de todo el mundo un grupo de participantes que en la ronda semi-final tuvieron que demostrar sus habilidades en distintos juegos. En el archivo "`participantes.semi`" se encuentran los números de usuario de todos los participantes de la ronda semi-final (números naturales). En los archivos "`juego.uno`", "`juego.dos`" y "`juego.tres`", se encuentran los resultados de cada uno de los juegos. Cada uno de estos tres archivos tiene un par de elementos que corresponde al número de usuario de un participante, seguido del puntaje obtenido en ese juego (número natural), y así para todos los que participaron en ese juego. En cada uno de los siguientes incisos ponga especial atención en dividir el problema en partes y crear primitivas que faciliten la reutilización de código.

a) Se debe escribir un programa en Pascal que dado un número de usuario indique si puede participar en la final de la competencia. Un participante podrá participar en la final si se encuentra en los tres archivos de puntajes.

b) Se debe escribir un programa en Pascal que permita mostrar quien fue el mejor participante de cada juego. La mejor participación en un juego consiste de haber obtenido el puntaje más alto.

c) Se debe escribir un programa en Pascal que permita mostrar quien fue el mejor participante de cada juego y que puntajes obtuvo ese mismo participante en los otros dos juegos.

d) Se debe escribir un programa en Pascal que permita crear un archivo de enteros con los participantes que obtuvieron más de una cantidad "tope" de puntos en los tres juegos (tope ingresado por el usuario).

Ejercicio 9: Para cada uno de los incisos siguientes a, b y c, (en forma individual) se solicita que:

i) Divida el problema en subproblemas y haga un gráfico o esquema de su propuesta de diseño para la solución.

ii) Describa las funciones y procedimientos necesarios identificando los parámetros de entrada y salida, agregando una breve descripción del objetivo de la primitiva.

iii) Realice un programa en PASCAL que resuelva el problema. No es necesario implementar las primitivas, simplemente deberá declarar los encabezados de cada una.

Suponga que cuenta con tres archivos A, B y C, y todos tienen ingresados números reales. Para cada uno de los siguientes incisos se desea generar otro archivo nuevo con aquellos elementos de A, B y C, que respeten lo indicado:

a) Copiar todos los elementos que se encuentre en A y en B pero no en C.

b) Copiar aquellos elementos que aparezcan en A una cantidad par de veces o aparezcan en B una cantidad impar de veces, y que si aparece en C entonces no debería aparecer en la misma cantidad que apareció en A o en B.

c) Copiar todo elemento que ocurra en A antes que en B, y en B antes que en C. Si el elemento no aparece en algún archivo entonces no se copia.

Ejercicio 10: **Fecha Válida**

Realice una función para determinar si una fecha es válida. La fecha es representada por tres números enteros **dia, mes** y **anio**. Por ejemplo si dia= 21, mes= 10 y anio=2008, la fecha es válida. Si dia=29, mes=2 y anio = 2010 la fecha no es válida. Utilice para determinar la validez de la fecha la sentencia CASE.

Ejercicio 11: **Codificador**

Escriba un programa en Pascal que procese una secuencia de caracteres ingresada por teclado y terminada en punto, codificándola de la siguiente manera: cada vocal se reemplaza por el carácter que se indica en la tabla que sigue, el resto de los caracteres (incluyendo a las vocales acentuadas) se mantienen sin cambios.

a	e	i	o	U
@	#	$	%	*

Realice una función que reciba una vocal y retorne la codificación correspondiente. Utilice la sentencia CASE para la transformación.

Por ejemplo, si el usuario ingresa: `Ayer, lunes, salimos a las once y 10.`
La salida del programa debería ser: `@y#r, l*n#s, s@l$m%s @ l@s %nc# y 10.`

Ejercicio 12: **Operaciones**

Realice un programa en PASCAL que lea un número real A, un operador aritmético OP (que puede ser +, -, *, /) y otro número real B y calcule el valor de la expresión A OP B. Implemente una función que evalúe la expresión.
Un ejemplo de la interacción del programa con el usuario es:

```
--------------------------------------------------------------------------
Ingrese A: 2.0 <Enter>
Ingrese OP: + <Enter>
Ingrese B: 4.5 <Enter>

El resultado es: 6.5
--------------------------------------------------------------------------
```

Otro ejemplo es:

```
--------------------------------------------------------------------------
Ingrese A: 2.0 <Enter>
Ingrese OP: / <Enter>
Ingrese B: 0 <Enter>

No se puede calcular (división por cero).
--------------------------------------------------------------------------
```

Y uno más:

```
--------------------------------------------------------------------------
Ingrese A: 2.0 <Enter>
Ingrese OP: - <Enter>
Ingrese B: 4.5 <Enter>

El resultado es: -2.5
--------------------------------------------------------------------------
```

Ejercicio 13: *En este ejercicio preste especial atención a la correcta* ***división del problema****.*

Considere que una empresa de transportes de latas de bebidas tiene dos camiones y en cada uno de ellos puede llevar hasta 100 cajas de latas. Cada caja tiene un código individual único representado por un número entero el cual se utiliza para su seguimiento durante el transporte. La empresa tiene dos archivos de enteros "camión1" y "camión2", cada archivo con los códigos de las cajas que ya han sido cargadas en cada camión y que aún no ha salido de viaje. Además, para el nuevo lote de cajas que no fue cargado aún, la empresa tiene otro archivo de enteros llamado "lote_a_cargar" con los códigos de todas las cajas de ese lote.

Se debe escribir un programa en Pascal que leyendo la información de los archivos indicados antes, permita al operador del sistema realizar estas tareas: (1) mostrar cuántas cajas pueden aún cargarse en cada camión, (2) mostrar si dado el espacio disponible entre los dos camiones, alcanza para cargar todas las cajas de "lote_a_cargar", o de lo contrario cuantas cajas quedarían sin cargar, y (3) dado un código, mostrar en donde está esa caja (camión 1, camión 2, lote, o no está en transporte)

<u>Por ejemplo</u>, si se tienen estos datos,

Camión1	201 202 203 204 205 206
Camión 2	301 302 303 304 305
Lote_a_cargar	91 92 93 94 95 96 97

se mostrará por pantalla:

```
Ingrese opción (1) disponible (2) hay espacio (3) donde caja  > 1
Pueden cargarse. Camión 1: 94 cajas, y camión 2: 95 cajas
Ingrese opción (1) disponible (2) hay espacio (3) donde caja  > 2
El lote puede cargarse completamente
Ingrese opción (1) disponible (2) hay espacio (3) donde caja  > 3
Ingrese código de caja a buscar: 94
La caja buscada está en el lote a cargar
```

Ejercicio 14: Implemente un procedimiento que dado un dígito d $\in$ [1..9] muestre por pantalla el siguiente renglón:

1 2 3 .. d .. 3 2 1

Por ejemplo, si **d = 6** el procedimiento deberá imprimir **1 2 3 4 5 6 5 4 3 2 1**
El encabezamiento del procedimiento sería:

PROCEDURE ImprimeRenglón(digito:integer);

Escriba un programa en Pascal utilizando dicho procedimiento, para que solicite un dígito **d** al usuario, y muestre por pantalla una figura como la siguiente

1
1 2 1
1 2 3 2 1
1 2 3 4 3 2 1
...
1 2 3 .....d..... 3 2 1

Ejercicio 15: Se dice que *M* es el número maximal para *N*, si *M* es **el mayor número que puede formarse usando los dígitos de *N***. Ejemplos: Si *N*=125345, el número maximal *M* es 554321; si *N*=2756, *M* es 7652.

Escriba un programa en Pascal que lea dos números naturales *a* y *b* y muestre por pantalla todos los números *Num* comprendidos entre *a* y *b* que verifiquen que coinciden con su maximal:

Por ejemplo: para *a* = 320 y *b* = 332, el programa deberá mostrar por pantalla:
Los números entre 320 y 332 que coinciden con su maximal son: 320, 321, 322, 330,331, 332

Ejercicio 16: Conteste las siguientes preguntas dando un ejemplo en el caso que la situación planteada sea posible, o fundamentando su respuesta con conceptos teóricos. Dentro de un programa en Pascal:

- ¿Pueden dos procedimientos tener el mismo nombre?
- ¿Pueden haber dos funciones con el mismo identificador?
- ¿Puede un identificador de constante ser igual a un identificador de variable?
- Indique cuando un procedimiento P puede llamar a una función F que está declarada dentro de otro procedimiento Q, y cuando no.
- ¿Puede una variable local tener como nombre V si está declarada dentro de un procedimiento cuyo nombre también es V?
- ¿Puede una variable local tener como nombre V si está declarada dentro de una función cuyo nombre es V? ¿Hay alguna diferencia con respecto a que V sea un procedimiento?

Ejercicio 17: Reescriba el programa generado en el **ejercicio 1** utilizando el siguiente procedimiento.

```
procedure Invertir( num: integer; var inv:integer );
{Objetivo: Invierte el orden de los dígitos de un número entero
 Entrada: el parámetro "Num" recibe el número que se desea invertir.
 Salida: Retornará un entero con los dígitos de "Num" en orden inverso
 Ejemplo: si "num" es 1234 retornará 4321}
begin
   inv := 0;
   while num > 0 do
   begin
      inv := inv * 10 + (num mod 10);
      num := num div 10;
   end;
end;
```

Ejercicio 18: Escriba un programa que permita ingresar la fecha del día actual, y luego solicite al usuario las fechas de ida y vuelta de un viaje. El ingreso de cada fecha deberá realizarlo permitiendo el ingreso de una secuencia de caracteres con el formato dd/mm/aa. Controle que todas las fechas sean válidas y que caso que no lo sean continúe solicitando la fecha hasta que ingrese una fecha válida, a tal efecto utilice la función implementada en el ejercicio 10. Luego controle que la fecha de viaje sea posterior o igual a la del día actual, y que la de regreso sea posterior o igual a la de ida del viaje. Finalmente si pasa todos los controles solicite el número de tarjeta de crédito (que consiste de 4 números de exactamente 4 cifras) y la fecha de vencimiento de la tarjeta (la cual debe estar vigente), sobre dichos valores realice los controles necesarios.