Solucionario - Práctica N° 5
Funciones y Procedimientos

Solucionario 1:

```
VARIABLES
   ENTERO : *v, i, mayor, menor
   REAL     : promedio, cantidad
ACCION Hallar_mayor_menor_promedio_numeros
   ESCRIBIR("Ingrese la cantidad de elementos del vector : ")   LEER(cantidad)

   v ← RESERVAR ENTERO[ cantidad ]

   PARA i DESDE 0 HASTA cantidad-1 HACER
      ESCRIBIR("Ingrese el numero : ")  LEER( v[ i ] )
   FIN_PARA

   mayor ← hallar_mayor(v,cantidad)
   menor ← hallar_menor(v,cantidad)
   promedio ← hallar_promedio(v,cantidad)

  ESCRIBIR("El numero mayor es : ", mayor)
  ESCRIBIR("El numero menor es : ", menor)
  ESCRIBIR("El promedio de  los numeros : ", promedio)

FIN_ACCION

ENTERO FUNCION hallar_mayor(ENTERO *vector, ENTERO n)
   ENTERO : i, mayor

   mayor ← -999

   PARA i DESDE 0 HASTA n-1 HACER
      SI( vector[ i ] > mayor )
        ENTONCES
           mayor ← vector[ i ]
      FIN_SI
   FIN_PARA

   RETORNAR mayor
FIN_FUNCION

ENTERO hallar_menor(ENTERO *vector, ENTERO n)
   ENTERO : i, menor

   menor ← 999

   PARA i DESDE 0 HASTA n-1 HACER
       SI( vector[ i ] < menor)
          ENTONCES
             menor ← vector[ i ]
       FIN_SI
   FIN_PARA
```

```
   RETORNAR menor
FIN_FUNCION

ENTERO FUNCION hallar_promedio(ENTERO *vector, ENTERO n)
   ENTERO : i, suma, promedio

   suma ← 0

   PARA i DESDE 0 HASTA n-1 HACER
       suma ← suma + vector[ i ]
   FIN_PARA

   promedio ← suma/n

   RETORNAR promedio
FIN_FUNCION
```

Solucionario 2:

```
VARIABLES
    ENTERO : *v, i, cantidad
ACCION Ordenar_vector_numero_diferentes
    ESCRIBIR("Ingrese la cantidad de elementos del vector : ")   LEER(cantidad)

    v ← RESERVAR ENTERO[ cantidad ]

    PARA i DESDE 0 HASTA cantidad-1 HACER
        ESCRIBIR("Ingrese el numero : ")  LEER(v[ i ])
    FIN_PARA

    actualizar_vector(v, cantidad)

    ESCRIBIR("Los elementos del vector son : ")

    PARA i DESDE 0 HASTA cantidad-1 HACER
      SI( v[ i ] < > -99 )
        ENTONCES
           ESCRIBIR( v[ i ] )
      FIN_SI
    FIN_PARA

FIN_ACCION

PROCEDIMIENTO actualizar_vector(ENTERO *v, ENTERO n)
   ENTERO : i, j, aux

   PARA i DESDE 0 HASTA n-1 HACER
      PARA j DESDE 0 HASTA n-2 HACER
          SI( v[ j ] > v[ j+1 ] )
            ENTONCES
              aux ← v[ j ]
              v[ j ] ← v[ j+1 ]
              v[ j+1 ] ← aux
          FIN_SI
      FIN_PARA
```

```
   FIN_PARA

    PARA i DESDE 0 HASTA n-1 HACER
       aux ← v[ i ]
       PARA j DESDE i+1 HASTA n-1 HACER
          SI( aux = v[ j ] )
            ENTONCES
              v[ j ] ← -99
          FIN_SI
       FIN_PARA
    FIN_PARA

FIN_PROCEDIMIENTO
```

Solucionario 3:

```
VARIABLES
   ENTERO : n, fibo
ACCION Hallar_numero_fibonacci
   ESCRIBIR("Ingrese la ubicacion del numero fibonacci : ")   LEER(n)

   fibo ← hallar_numero_fibonacci( n )

   ESCRIBIR("El numero fibonacci es : ", fibo)

FIN_ACCION

ENTERO FUNCION hallar_numero_fibonacci(ENTERO n)
   ENTERO : i
   ENTERO : fibo, fibo1, fibo2

   i ← 0
   fibo1 ← 0
   fibo2 ← 1

   SI( n = 1 )
     ENTONCES
        RETORNAR 1
     SINO
        MIENTRAS( i < n-1 )
          HACER
             fibo ← fibo1 + fibo2
             fibo1 ← fibo2
             fibo2 ← fibo
             i ← i + 1
        FIN_MIENTRAS
   FIN_SI

   RETORNAR fibo

FIN_FUNCION
```

Solucionario 4:

```
VARIABLES
   ENTERO : i, j, cantidad
   REAL     : suma, num, numero
ACCION Hallar_seno_angulo
   ESCRIBIR("Ingrese el angulo en grados sexagesimales   : ")  LEER(num)
   ESCRIBIR("Ingrese la cantidad de terminos de la serie : ")  LEER(cantidad)

  suma ← 0
  numero ← num*PI/180

  PARA i DESDE 0 HASTA cantidad-1 HACER
    SI( i MOD 2 = 0 )
      ENTONCES
        suma ← suma + potencia(numero,2*(i+1)-1)/factorial(2*(i+1)-1)
      SINO
        suma ← suma - potencia(numero,2*(i+1)-1)/factorial(2*(i+1)-1)
    FIN_SI
  FIN_PARA

  ESCRIBIR("El valor de Sen(", num, ") es : ", suma)

FIN_ACCION


REAL FUNCION potencia(REAL a, ENTERO b)
  ENTERO : i, pot

  pot ← 1

  PARA i DESDE 0 HASTA b-1 HACER
     pot ← pot*a
  FIN_PARA

  RETORNAR pot

FIN_FUNCION


ENTERO FUNCION factorial(ENTERO n)
  ENTERO : i, fact

  fact ← 1

  PARA i DESDE 1 HASTA n HACER
     fact ← fact*i
  FIN_PARA

  RETORNAR fact

FIN_FUNCION
```

Solucionario 5:

```
VARIABLES
   ENTERO  : numero
ACCION Hallar_factorial
   ESCRIBIR("Ingrese el numero : ")  LEER(numero)

   ESCRIBIR("El factorial de ", numero, " es : ", factorial(numero) )

FIN_ACCION

REAL FUNCION factorial(ENTERO n)
   REAL : facto

   SI( n > 0 )
     ENTONCES
        facto ← n*factorial( n-1 )
     SINO
        facto ← 1
   FIN_SI

   RETORNAR facto

FIN_FUNCION
```

Solucionario 6:

```
VARIABLES
   ENTERO : cantidad
ACCION Hallar_numero_fibonacci
   ESCRIBIR("Numero fibonacci : ")
   ESCRIBIR("Ingrese la ubicacion del numero : ")   LEER(cantidad)

   ESCRIBIR("El numero fibonacci es : ", fibonacci(cantidad) )

FIN_ACCION

REAL FUNCION fibonacci(ENTERO n)
   ENTERO : fibo

   SI( n = 1 o n = 2 )
     ENTONCES
       fibo ← 1
     SINO
       fibo ← fibonacci( n-2 ) + fibonacci( n-1 )
   FIN_SI

   RETORNAR fibo

FIN_FUNCION
```

Solucionario 7:

```
VARIABLES
   ENTERO : n
   REAL     : cantidad
```

```
ACCION Hallar_numero_cuadrados

   ESCRIBIR("Ingrese la dimension del tablero : ")   LEER(n)

   cantidad←hallar_cuadrados(n)

   ESCRIBIR<<"La cantidad de cuadrados que existe en el tablero es : "<<cantidad;

FIN_ACCION

REAL FUNCION hallar_cuadrados(ENTERO n)
   REAL : cuadr

   SI ( n > 0 )
     ENTONCES
        cuadr ← n*n + hallar_cuadrados( n – 1 )
     SINO
        cuadr ← 0
   FIN_SI

   RETORNAR cuadr

FIN_FUNCION
```

Solucionario 8:

```
VARIABLES
   ENTERO : num1, num2
ACCION Intercambiar_numeros
   ESCRIBIR("Intercambiar numeros:")
   ESCRIBIR("Ingrese el primer numero  : ")   LEER(num1)
   ESCRIBIR("Ingrese el segundo numero : ")  LEER(num2)

   intercambiar(num1,num2)

   ESCRIBIR("El primer numero es : ", num1)
   ESCRIBIR("El segundo numero es : ", num2)

FIN_ACCION

PROCEDIMIENTO intercambiar(ENTERO &num1, ENTERO &num2)
   ENTERO : aux

  aux ← num2
  num2 ← num1
  num1 ← aux

FIN_PROCEDIMIENTO
```

Solucionario 9:

```
VARIABLES
   ARREGLO CARACTER  : usuario[10], password[10]
ACCION Verificar_usuario
```

```
HACER
    CERRAR_GRAFICO( )
    INICIAR_GRAFICO( )
    fondo_principal( )
    ventana_acceso( )

    COPIAR_CADENA(usuario,accesar_usuario( ))
    COPIAR_CADENA(password,accesar_password( ))

    SI( COMPARAR_CADENA(usuario,"fenix") <> 0 o COMPARAR_CADENA (password,"fisi") <> 0)
      ENTONCES
        CERRAR_GRAFICO( )
        INICIAR_GRAFICO(&gdriver, &gmode, "")
        mensaje_error( )
    FIN_SI

 MIENTRAS(COMPARAR_CADENA(usuario,"fenix") <> 0 o COMPARAR_CADENA(password,"fisi") <> 0)

 CERRAR_GRAFICO( )
 INICIAR_GRAFICO( )

 ingresar_sistema( )

 CERRAR_GRAFICO( )

FIN_ACCION
```

Solucionario 10:

```
VARIABLES GLOBALES
  CADENA CARÁCTER : A[5], B[5], C[5]
  ENTERO     : a, b, c

  a ← 0
  b ← 0
  c ← 0

VARIABLES
   ENTERO    : eF1, eF2
  CARÁCTER : caracter
ACCION Juego_Torre_Hanoi
   INICIAR_GRAFICO( )

   fondo_pantalla( )

   PARA ? DESDE ? HASTA ? HACER
      caracter=CAPTURAR_TECLA( )

      SEGUN_SEA(caracter)
         CASO  F1 : SI( eF1 = 0 )
                      ENTONCES
                        pintar_piramide( )
                        eF1 ← 1
                    FIN_SI
```

```
        CASO F2 :  SI( eF2 = 0 )
                     ENTONCES
                       iniciar_juego(5,1,2,3)
                       eF2 ← 1
        CASO ESC : SALIR( )
      FIN_SEGÚN

   FIN_PARA

   CERRAR_GRAFICO( )

FIN_ACCION


PROCEDIMIENTO mover_cuadro(ENTERO ndiscos, ENTERO origen, ENTERO destino, ENTERO auxiliar)
  SI( ndiscos = 1 )
    ENTONCES
       mover_disco(origen,destino)
    SINO
       mover_cuadro(ndiscos-1,origen,auxiliar,destino)
       mover_disco(origen,destino)
       mover_cuadro(ndiscos-1,auxiliar,destino,origen)
  FIN_SI

FIN_PROCEDIMIENTO


PROCEDIMIENTO iniciar_juego(ENTERO ndiscos, ENTERO origen, ENTERO auxiliar, ENTERO destino)

   mover_cuadro(ndiscos,origen,auxiliar,destino)

FIN_PROCEDIMIENTO


PROCEDIMIENTO mover_disco(ENTERO origen, ENTERO destino)
   RETARDADOR(800)

   SI( origen = 1 y destino = 2 )
     ENTONCES
       B[b] ← A[ a-1 ]
   FIN_SI

   SI( origen = 1 y destino = 3 )
     ENTONCES
       C[c] ← A[ a-1 ]
   FIN_SI

   SI( origen = 2 y destino = 1 )
     ENTONCES
       A[a] ← B[ b-1 ]
   FIN_SI

   SI( origen = 2 y destino = 3 )
     ENTONCES
       C[c] ← B[ b-1 ]
   FIN_SI
```

```
   SI( origen = 3 y destino = 1 )
     ENTONCES
        A[a] ← C[ c-1 ]
   FIN_SI

   SI( origen = 3 y destino = 2 )
     ENTONCES
        B[b] ← C[ c-1 ]
   FIN_SI

    SI( origen = 1 )
      ENTONCES
         borrar_caja(100+10*(A[a-1]-1),400-40*(a-1),200-10*(A[a-1]-1),400-40*a)
         a ← a - 1
    FIN_SI

    SI( origen = 2 )
      ENTONCES
         borrar_caja(300+10*(B[b-1]-1),400-40*(b-1),400-10*(B[b-1]-1),400-40*b)
         b ← b - 1
    FIN_SI

    SI( origen = 2 )
      ENTONCES
         borrar_caja(500+10*(C[c-1]-1),400-40*(c-1),600-10*(C[c-1]-1),400-40*c)
         c ← c - 1
    FIN_SI

    RETARDADOR(800)

    SI( destino = 1 )
      ENTONCES
         pintar_caja(100+10*(A[a]-1),400-40*a,200-10*(A[a]-1),400-40*(a+1))
         a ← a + 1
    FIN_SI

    SI( destino = 2 )
      ENTONCES
         pintar_caja(300+10*(B[b]-1),400-40*b,400-10*(B[b]-1),400-40*(b+1))
         b ← b + 1
    FIN_SI

    SI( destino = 3 )
      ENTONCES
         pintar_caja(500+10*(C[c]-1),400-40*c,600-10*(C[c]-1),400-40*(c+1))
         c ← c + 1
    FIN_SI

FIN_PROCEDIMIENTO
```

```
PROCEDIMIENTO pintar_caja(ENTERO x0, ENTERO y0, ENTERO x1, ENTERO y1)
   RELLENAR_RECTANGULO(x0,y0,x1,y1)
FIN_PROCEDIMIENTO

PROCEDIMIENTO borrar_caja(ENTERO x0, ENTERO y0, ENTERO x1, ENTERO y1)
   RELLENAR_RECTANGULO(x0,y0,x1,y1)
FIN_PROCEDIMIENTO
```