

Capítulo 8

Funciones

Los enunciados presentados en este capítulo corresponden a los que se plantean en el capítulo 11 del libro LÓGICA DE PROGRAMACIÓN del mismo autor.

Concepto

Una función es un fragmento de programa que a) cumple con un pequeño objetivo, b) forma parte integral de otro(s) programa(s), c) tiene un nombre que la identifica, d) puede recibir parámetros para alcanzar su objetivo, e) tiene un cuerpo de instrucciones que permite lograr el objetivo, f) puede retornar valores a la función que la invoca siempre y cuando se le envíen los parámetros apropiados

Utilidad

Las funciones permiten resolver los tres grandes problemas de la programación:

- a. La simplificación del objetivo
- b. La reutilización del código
- c. La realización fácil de pruebas de escritorio

Recuerde que una función se hace efectiva, cuando otra la invoca con los parámetros apropiados.

Ejercicios

1. **Construir una función que reciba como parámetro un entero y retorne su último dígito.**

Función Ult_Dig (Entero: Num)

Variables

 Entero: UltDig

Inicio

 UltDig = Num MOD 10

 Retorne(UltDig)

Fin

- 2. Construir una función que reciba como parámetro un entero y retorne sus dos últimos dígitos.**

Función 2Ult_Dig (Entero: Num)

Variables

 Entero: 2UltDig

Inicio

 UltDig = Num MOD 100

 Retorne(UltDig)

Fin

- 3. Construir una función que reciba como parámetro un entero y retorne la cantidad de dígitos.**

Función Cant_Dig(Entero: Num)

Variables

 Entero: CuentaDig

Inicio

 CuentaDig = 0

 Mientras Num > 0

 CuentaDig = CuentaDig + 1

 Num = Num / 10

 Fin_Mientras

 Retorne(CuentaDig)

Fin

- 4. Construir una función que reciba como parámetro un entero y retorne la cantidad de dígitos pares.**

Función Cant_DigPares(Entero: Num)

Variables

 Entero: CuentaDigPares

Inicio

 CuentaDigPares = 0

 Mientras Num > 0

 UltDig = Num MOD 10

```

    Si UltDig MOD 2 = 0 entonces
        CuentaDigPares = CuentaDigPares + 1
    Fin_Si
    Num = Num / 10
Fin_Mientras
Retorne(CuentaDigPares)
Fin

```

- 5. Construir una función que reciba como parámetro un entero y retorne la cantidad de dígitos primos.**

```

Función Cant_DigPrimos(Entero: Num)
Variables
    Entero: CuentaDigPrimos
Inicio
    CuentaDigPrimos = 0
    Mientras Num > 0
        UltDig = Num MOD 10
        Si      UltDig = 2 OR UltDig = 3 OR
        UltDig = 5 OR UltDig = 7 entonces
            CuentaDigPrimos = CuentaDigPares + 1
        Fin_Si
        Num = Num / 10
    Fin_Mientras
    Retorne(CuentaDigPrimos)
Fin

```

- 6. Construir una función que reciba como parámetro un entero y retorne el carácter al cual pertenece ese entero como código ASCII.**

```

Función De_Ent_A_Ascii(Entero: Num)
Variables
    Caracter: CarAscii
Inicio
    CarAscii = Num
    Retorne(CarAscii)
Fin

```

Nota: La implementación de esta función dependerá de las posibilidades que ofrezca el lenguaje de programación

- 7. Construir una función que reciba como parámetro un carácter y retorne el código ASCII asociado a él.**

Función De_Ascii_A_Ent(Caracter: CarAscii)

Variables

 Entero: Num

Inicio

 Num = CarAscii

 Retorne(Num)

Fin

Nota: La implementación de esta función dependerá de las posibilidades que ofrezca el lenguaje de programación

8. Construir una función que reciba como parámetro un entero y retorne 1 si dicho entero está entre los 30 primeros elementos de la serie de Fibonacci. Deberá retornar 0 si no es así.

Función EsFiboONo(Entero: Num)

Variables

 Entero: VFibo(30), Ind

Inicio

 VFibo(1) = 0

 VFibo(2) = 1

 Para Ind = 3 hasta 30 Paso 1

 VFibo(Ind) = VFibo(Ind – 1) + VFibo(Ind – 2)

 Fin_Para

 Para Ind = 1 hasta 30 Paso 1

 Si Num = VFibo(Ind) entonces

 Retorne(1)

 Sino

 Retorne(0)

 Fin_Si

 Fin_Para

Fin

9. Construir una función que reciba un entero y le calcule su factorial sabiendo que el factorial de un número es el resultado de multiplicar sucesivamente todos los enteros comprendidos entre 1 y el número dado. El factorial de 0 es 1. Aunque no están definidos los factoriales de números negativos, esta condición se evalúa por fuera de la función antes de su invocación.

Función Factorial(Entero: Num)

Variables

```

Entero: Factorial, Ind
Inicio
    Factorial = 1
    Para Ind = 1 hasta Num Paso 1
        Factorial = Factorial * Ind
    Fin_Para
    Retorne(Num)
Fin

```

10. Construir una función que reciba como parámetro un entero y retorne el primer dígito de este entero.

Función PrimerDig(Entero: Num)

Variables

No se requieren variables locales

```

Inicio
    Mientras Num >= 10
        Num = Num / 10
    Fin_Mientras
    Retorne(Num)
Fin

```

11. Construir una función que reciba como parámetro un entero y un dígito y retorne 1 si dicho entero es múltiplo de dicho dígito y 0 si no es así.

Función NumYDig(Entero: Num, Dig)

Variables

No se requieren variables locales

```

Inicio
    Si Num MOD Dig = 0 entonces
        Retorne(1)
    Sino
        Retorne(0)
    Fin_Si
Fin

```

12. Construir una función que reciba como parámetro un entero y un dígito y retorne 1 si dicho dígito está en dicho entero y 0 si no es así.

Función DigEnNum(Entero: Num, Dig)

Variables

Entero: UltDig

```

Inicio
    Mientras Num >= 0

```

```
Si Num MOD 10 = Dig entonces
    Retorne(1)
Sino
    Num = Num / 10
Fin_Si
Fin_Mientras
Retorne(0)
Fin
```

13. Construir una función que reciba como parámetro un entero y un dígito y retorne la cantidad de veces que se encuentra dicho dígito en dicho entero.

Función CuentaDigEnNum(Entero: Num, Dig)

Variables

```
    Entero: Contador
Inicio
    Contador = 0
    Mientras Num >= 0
        Si Num MOD 10 = Dig entonces
            Contador = Contador + 1
        Fin_Si
        Num = Num / 10
    Fin_Mientras
    Retorne(Contador)
Fin
```

14. Construir una función que reciba como parámetros dos números enteros y retorne el valor del mayor.

Función MayorDe2Nums(Entero: Num1, Num2)

Variables

```
    No se requieren variables locales
Inicio
    Si Num1 > Num2 entonces
        Retorne(Num1)
    Sino
        Retorne(Num2)
    Fin_Si
Fin
```

Nota: En caso de que los dos números sean iguales no habrá un mayor pero esta condición debe evaluarse antes de invocar la función.

15. Construir una función que reciba como parámetros dos números enteros y retorne 1 si el primer número es múltiplo del segundo y 0 si no.

Función Num1MultDeNum2(Entero: Num1, Num2)

Variables

No se requieren variables locales

Inicio

Si Num1 MOD Num2 = 0 entonces

Retorne(1)

Sino

Retorne(0)

Fin_Si

Fin

16. Construir una función que reciba como parámetro un entero y retorne 1 si corresponde al código ASCII de una letra minúscula (Los códigos ASCII de las letras minúsculas van desde 97 que es el código de la letra a hasta 122 que es el código de la letra z). Deberá retornar 0 si no es así.

Función Ascii_Minus(Entero: Num)

Variables

No se requieren variables locales

Inicio

Si Num >= 97 AND Num <= 122 entonces

Retorne(1)

Sino

Retorne(0)

Fin_Si

Fin

17. Construir una función que reciba como parámetro un entero y retorne 1 si corresponde al código ASCII de un dígito (Los códigos ASCII de las letras minúsculas van desde 48 que es el código del dígito 0 hasta 57 que es el código del dígito 9). Deberá retornar 0 si no es así.

Función Ascii_Digito(Entero: Num)

Variables

No se requieren variables locales

Inicio

Si Num >= 48 AND Num <= 57 entonces

Retorne(1)

Sino

Retorne(0)

Fin_Si

Fin

- 18. Construir una función que reciba como parámetro un valor entero y retornar 1 si dicho valor es el factorial de alguno de los dígitos del número. Deberá retornar 0 si no es así.**

Función BuscaFact(Entero: Num)

Variables

 Entero: Factorial, Sw, Dig, Aux

Inicio

 Aux1 = Num

 Mientras Aux1 > 0

 Factorial = 1

 Dig = Aux1 MOD 10

 Para Aux2 = 1 hasta Dig Paso 1

 Factorial = Factorial * Aux2

 Fin_Para

 Si Factorial = Num entonces

 Retorne(1)

 Fin_Si

 Fin_Mientras

 Retorne(0)

Fin

- 19. Construir una función que reciba como parámetro un entero y retorne 1 si dicho valor es un número de mínimo 3 dígitos. Deberá retornar 0 si no es así.**

Función Num3Dig(Entero: Num)

Variables

 No se requieren variables locales

Inicio

 Si Num >= 100 AND Num <= 999 entonces

 Retorne(1)

 Sino

 Retorne(0)

 Fin_Si

Fin

- 20. Construir una función que reciba como parámetro un entero y retorne 1 si en dicho valor todos los dígitos son iguales. Deberá retornar 0 si no es así.**

Función DigitosIguales(Entero: Num)

Variables

```

Entero: Contador1, Contador2, Dig, Aux
Inicio
    Aux = Num
    Contador1 = 0
    Mientras Aux > 0
        Contador1 = Contador1 + 1
        Aux = Aux / 10
    Fin_Mientras

    Dig = Num MOD 10
    Contador2 = 0
    Mientras Num > 0
        Si Num MOD 10 = Dig entonces
            Contador2 = Contador2 + 1
        Fin_Si
        Num = Num / 10
    Fin_Mientras

    Si Contador1 = Contador2 entonces
        Retorne(1)
    Sino
        Retorne(0)
    Fin_Si
Fin

```

- 21. Construir una función que reciba como parámetro un entero y retorne 1 si en dicho valor el primer dígito es igual al último. Deberá retornar 0 si no es así.**

Función PrimDig_UltDig(Entero: Num)

Variables

 Entero: PrimDig, Ultdig

Inicio

 UltDig = Num MOD 10

 Mientras Num > 10
 Num = Num / 10
 Fin_Mientras

 PrimDig = Num
 Si UltDig = PrimDig entonces
 Retorne(1)
 Sino
 Retorne(0)
 Fin_Si

Fin

22. Construir una función que reciba como parámetro un entero y retorne 1 si dicho valor es múltiplo de 5. Deberá retornar 0 si no es así.

Función Multiplo_De_5(Entero: Num)

Variables

No se requieren variables locales

Inicio

Si Num MOD 5 = 0 entonces

Retorne(1)

Sino

Retorne(0)

Fin_Si

Fin

23. Construir una función que reciba como parámetro dos enteros y retorne 1 si la diferencia entre los dos valores es un número primo. Deberá retornar 0 si no es así.

Funcion Dif_Num_Primo(Entero: Num1, Num2)

Variables

Entero: Aux, Contador, Dif

Inicio

Dif = Num1 – Num2

Contador = 0

Para Aux = 1 hasta Dif Paso 1

Si Dif MOD Aux = 0 entonces

Contador = Contador + 1

Fin_Si

Fin_Para

Si Contador = 2 entonces

Retorne(1)

Sino

Retorne(0)

Fin_Si

Fin

24. Construir una función que reciba como parámetro dos enteros de dos dígitos cada uno y retorne 1 si son inversos. Ejemplo: 83 es inverso de 38. Deberá retornar 0 si no es así.

Función Nums_Inversos(Entero: Num1, Num2)

Variables

Entero: Dig11, Dig12, Dig21, Dig22

Inicio

Dig11 = Num1 / 10

Dig12 = Num1 MOD 10

Dig21 = Num2 / 10

Dig22 = Num2 MOD 10

Si Dig11 = Dig22 AND Dig12 = Dig 21 entonces

Retorne(1)

Sino

Retorne(0)

Fin_Si

Fin

25. Construir una función que reciba como parámetro un entero y un dígito menor o igual a 5 y retorne el dígito del número que se encuentre en la posición especificada por el dígito que llegó como parámetro.

Función Pos_Dígito(Entero: Num, Dig)

Variables

Entero: Contador, NDig

Inicio

Contador = 0

Mientras Num > 0 AND Contador <= Dig-1

Num = Num / 10

Contador = Contador + 1

Fin_Mientras

NDig = Num MOD 10

Return(NDig)

Fin

26. Construir una función que reciba como parámetro un vector de 10 posiciones enteras y retorne el mayor de los datos del vector.

Función May_Vector10(Entero: Vector(10))

Variables

Entero: Ind, May

Inicio

May = 0

Para Ind = 1 hasta 10 Paso 1

Si Vector(Ind) > May entonces

May = Vector(Ind)

```
    Fin_Si  
Fin_Para  
Return(May)  
Fin
```

27. Construir una función que reciba como parámetro un vector de 10 posiciones enteras y retorne la posición en la cual se encuentra el mayor de los datos del vector.

Función PosMay_Vector10(Entero: Vector(10))

Variables

 Entero: Ind, May, PosMay

Inicio

 May = 0

 Para Ind = 1 hasta 10 Paso 1

 Si Vector(Ind) > May entonces

 May = Vector(Ind)

 PosMay = Ind

 Fin_Si

 Fin_Para

 Return(PosMay)

Fin

28. Construir una función que reciba como parámetro un vector de 10 posiciones enteras y retorne la cantidad de números primos almacenados en el vector.

Función NumsPrimos_Vector(Entero: V(10))

Variables

 Entero: Ind, Aux, Contador, CantPrimosV

Inicio

 CantPrimosV = 0

 Para Ind = 1 hasta 10 Paso 1

 Contador = 0

 Para Aux = 1 hasta V(Ind) Paso 1

 Si V(Ind) MOD Aux = 0 entonces

 Contador = Contador + 1

 Fin_Si

 Fin_Para

 Si Contador = 2 entonces

 CantPrimosV = CantPrimosV + 1

 Fin_Si

Fin_Para

Retorne(CantPrimosV)

Fin

- 29. Construir una función que reciba como parámetro un vector de 10 posiciones enteras y retorne la cantidad de números que pertenecen a los 30 primeros elementos de la serie de Fibonacci.**

Función NumsFibo_Vector(Entero: V(10))

Variables

Entero: VFibo(30), Contador, Ind, Aux

Inicio

 VFibo(1) = 0

 VFibo(2) = 1

 Para Ind = 3 hasta 30 Paso 1

 VFibo(Ind) = VFibo(Ind – 1) + VFibo(Ind – 2)

 Fin_Para

 Contador = 0

 Para Ind 1 hasta 10 Paso 1

 Para Aux = 1 hasta 30 Paso 1

 Si V(Ind) = VFibo(Aux) entonces

 Contador = Contador + 1

 Fin_Si

 Fin_Para

 Fin_Para

 Retorne(Contador)

Fin

- 30. Construir una función que reciba como parámetro un vector de 10 posiciones enteras y retorne la posición del mayor número primo almacenado en el vector.**

Función MayPrimo_Vector(Entero: V(10))

Variables

Entero: Ind, Aux, Contador, CantPrimosV, MayPrimo

Inicio

 MayPrimo = 0

 Para Ind = 1 hasta 10 Paso 1

 Contador = 0

 Para Aux = 1 hasta V(Ind) Paso 1

 Si V(Ind) MOD Aux = 0 entonces

 Contador = Contador + 1

 Fin_Si

 Fin_Para

 Si Contador = 2 entonces

```
Si V(Ind) > MayPrimo entonces  
    MayPrimo = V(Ind)  
Fin_Si  
Fin_Para  
Retorne(MayPrimo)  
Fin
```

31. Construir una función que reciba como parámetro un vector de 10 posiciones enteras y retorne el promedio entero del vector.

```
Función PromedioV(Entero: V(10))  
Variables  
    Entero: Ind, Promedio, Suma  
Inicio  
    Suma = 0  
    Para Ind = 1 hasta 10 Paso 1  
        Suma = Suma + V(Ind)  
    Fin_Para  
    Promedio = Suma / 10  
    Retorne(Promedio)  
Fin
```

32. Construir una función que reciba como parámetro un vector de 10 posiciones enteras y retorne el promedio real del vector.

```
Función PromedioV(Entero: V(10))  
Variables  
    Entero: Ind  
    Real: Promedio, Suma  
Inicio  
    Suma = 0  
    Para Ind = 1 hasta 10 Paso 1  
        Suma = Suma + V(Ind)  
    Fin_Para  
    Promedio = Suma / 10  
    Retorne(Promedio)  
Fin
```

33. Construir una función que reciba como parámetros un vector de 10 posiciones enteras y un valor entero de 1 dígito y retorne 1 si dicho dígito está en alguno de los datos del vector. Deberá retornar 0 si no es así.

```
Función Buscar_Dig_En_V(Entero: Vector(10), Dig)  
Variables
```

```

Entero: Ind
Inicio
    Para Ind = 1 hasta 10 Paso 1
        Aux = V(Ind)
        Mientras Aux > 0
            Si Aux MOD 10 = Dig entonces
                Retorne(1)
            Fin_Si
            Aux = Aux / 10
        Fin_Mientras
    Fin_Para
    Retorne(0)
Fin

```

- 34. Construir una función que reciba como parámetro un vector de 10 posiciones enteras y retorne la posición del número entero que tenga mayor cantidad de dígitos.**

Función CuentaDigitos(Entero: Num)

Variables

```

Entero: Contador
Inicio
    Contador = 0
    Mientras Num > 0
        Num = Num / 10
        Contador = Contador + 1
    Fin_Mientras
    Retorne(Contador)
Fin

```

Función CuentaDigitosV(Entero: V(10))

Variables

```

Entero: MayCantDig, NumMayCantDig, Ind,
Inicio
    MayCantDig = 0
    Para Ind = 1 hasta 10 Paso 1
        Si CuentaDigitos(V(Ind)) > MayCantDig entonces
            NumMayCantDig = V(Ind)
            MayCantDig = CuentaDigitos(V(Ind))
        Fin_Si
    Fin_Para
    Para Ind = 1 hasta 10 Paso 1
        Si V(Ind) = NumMayCantDig entonces

```

```
        Retorne(Ind)
    Fin_Si
Fin_Para
Fin
```

35. Construir una función que reciba como parámetro un vector de 10 posiciones enteras y retorne la posición en la que se encuentre el mayor número primo que termine en 3 almacenado en el vector.

Función PosMayNumPrimoV(Entero: V(10))

Variables

```
    Entero: Ind, MayPrimo, PosMayPrimo, Aux, Contador
Inicio
    MayPrimo = 0
    Para Ind = 1 hasta 10 Paso 1
        Contador = 0
        Aux = 1 hasta V(Ind) Paso 1
            Si V(Ind) MOD Aux = 0 entonces
                Contador = Contador + 1
            Fin_Si
        Fin_Para
        Si Contador = 2 entonces
            Si V(Ind) MOD 10 = 3 AND V(Ind) > MayPrimo
            Entonces
                Retorne(Ind)
            Fin_Si
        Fin_Si
    Fin_Para
    Retorne(0)
Fin
```

36. Construir una función que reciba como parámetro un entero y retorne ese elemento de la serie de Fibonacci dentro de sus 100 primeros elementos.

Función Fibonacci100_V(Entero: Num)

Variables

```
    Entero: Ind, VFibo(100)
Inicio
    VFibo(1) = 0
    VFibo(2) = 1
    Para Ind = 3 hasta 100 Paso 1
        VFibo(Ind) = VFibo(Ind - 1) + VFibo(Ind - 2)
    Fin_Para
```