**For**

1. Utilice un ciclo **for** para construir la función **multiplicaImpares**, que recibe como parámetro un número entero **n**. La función regresa la multiplicación de los primeros **n** números impares. **(20 puntos)**

**Ejemplos:**

**multiplicaImpares(3) =** 1 \* 3 \* 5 = **15**

**multiplicaImpares(4) =** 1 \* 3 \* 5 \* 7 = **105**

**multiplicaImpares(1) =** 1 = **1**

1. Utilice un ciclo **for** para construir la función **multiplicaPares**, que recibe como parámetro un número entero **n**. La función regresa la multiplicación de los primeros **n** números pares. **(20 puntos)**

**Ejemplos:**

**multiplicaPares(3) =** 2 \* 4 \* 6 = **48**

**multiplicaPares(4) =** 2 \* 4 \* 6 \* 8 = **384**

**multiplicaPares(1) =** 2 = **2**

decrementa

1. Utilice un ciclo **for** para construir la función **coseno**, que recibe como parámetros dos números enteros **x** y **n**. La función regresa la aproximación del coseno de **x**. Una aproximación del coseno de **x** se puede obtener usando la siguiente serie: **(20 puntos)**

*x2 x4 x6  x8 xn*

*coseno*(x, n)



2 4 6 8 *n*

**While**

1. Utilice un ciclo **while** para construir la función **aproximacionPI**, que recibe como parámetro el límite **N** de la serie para calcular la aproximación de PI. Existen diferentes algoritmos que nos permiten obtener una cercana aproximación al valor de la constante PI, uno de ellos es el Algoritmo Lord Brouncker: **(20 puntos)**



1. Utilice un ciclo **while** para construir la función **seno**, que recibe como parámetros dos números enteros **x** y **n**. La función regresa la aproximación del seno de **x**. Una aproximación del seno de **x** se puede obtener usando la siguiente serie: **(20 puntos)**



*seno(x , n)*

1 3 5 7 *n*

1. Utilice un ciclo **while** para construir la función **aproximacionPI**, que recibe como parámetro el límite **N** de la serie para calcular la aproximación de PI. Existen diferentes algoritmos que nos permiten obtener una cercana aproximación al valor de la constante PI, uno de ellos es el Algoritmo Lord Brouncker: **(20 puntos)**



1. Utilice un ciclo **while** y construya el procedimiento **TablaDeMultiplicar**, que recibe un entero **x** positivo como parámetro. El procedimiento deberá mostrar la tabla de multiplicar del número **x** hasta el número 10. **(20 puntos)**

**Ejemplos:**

**TablaDeMultiplicar(5)**

desplegará en pantalla:

Tabla del 5:

1 x 5 = 5

2 x 5 = 10

3 x 5 = 15

4 x 5 = 20

5 x 5 = 25

6 x 5 = 30

7 x 5 = 35

8 x 5 = 40

9 x 5 = 45

10 x 5 = 50

**TablaDeMultiplicar(12)**

desplegará en pantalla:

Tabla del 12:

1 x 12 = 12

2 x 12 = 24

3 x 12 = 36

4 x 12 = 48

5 x 12 = 60

6 x 12 = 72

7 x 12 = 84

8 x 12 = 96

9 x 12 = 108

10 x 12 = 120

**For y while**

1. Utilice los ciclos **do-while** y **for** para construir el procedimiento **aleatorios**, que despliega en pantalla 10 números aleatorios entre el rango de 0 a 300 hasta que el usuario teclee una N, indicando que desea terminar (Valida las letras minúsculas y mayúsculas). Nota: Las siguientes instrucciones te pueden ser útiles: srand(time(NULL)) y rand( )%. **(20 puntos)**

**Ejemplo:**

300 100 18 240 60 80 104 76 9 296

Desea continuar (S/N): S

50 120 21 297 4 82 120 240 96 194

Desea continuar (S/N): N

1. Utilice los ciclos **for** y **do-while** para construir el procedimiento **imprime5pares**, que despliega en pantalla 5 números consecutivos pares. Cada 5 números el procedimiento deberá preguntar al usuario si desea continuar, de ser así, continuará con la impresión de la secuencia de números hasta que el usuario tecle una N, indicando que desea terminar. NOTA: Valida las letras minúsculas y mayúsculas. **(20 puntos)**

**Ejemplo:**

2 4 6 8 10

Desea continuar (S/N): S

12 14 16 18 20

Desea continuar (S/N): N

1. Utilice los ciclos **do-while** y **for**, y construya el procedimiento **aleatorios**, que despliega en pantalla 5 números aleatorios entre el rango de 0 a 150 hasta que el usuario teclee una N, indicando que desea terminar (Valida las letras minúsculas y mayúsculas). Nota: Las siguientes instrucciones te pueden ser útiles: srand(time(NULL)) y rand( )%. **(20 puntos)**

**Ejemplo:**

15 130 93 3 27

Desea continuar (S/N): S

57 103 40 77 131

Desea continuar (S/N): S

150 91 16 103 39

Desea continuar (S/N): N

1. Utilice los ciclos **for** y **do-while** para construir el procedimiento **imprime6impares**, que despliega en pantalla 6 números consecutivos impares. Cada 6 números el procedimiento deberá preguntar al usuario si desea continuar, de ser así, continuará con la impresión de la secuencia de números hasta que el usuario tecle una N, indicando que desea terminar. **NOTA:** Valida las letras minúsculas y mayúsculas. **(20 puntos)**

**Ejemplo:**

1 3 5 7 9 11   
Desea continuar (S/N):   S   
13 15 17 19 21 23  
Desea continuar (S/N):   N

**Arreglos**

1. La función **cuentaMultiplosX** que recibe un vector de 10 valores enteros y un entero **x**. La función deberá regresar cuántos elementos del vector son múltiplos de **x**. **NOTA:** Suponer que el arreglo ya tiene valores **(20 puntos)**

**Ejemplo:**

Si el vector tuviera los siguientes valores: 1 3 **25 10** 4 **15** 7 6 8 **5** y **x** tuviera el valor de **5**.

La función regresaría: **4**

1. La función **sumaMultiplosX** que recibe un vector de 10 valores enteros y un entero **x**. La función deberá regresar el resultado de sumar todos los elementos múltiplos de **x** contenidos en el arreglo. **NOTA:** Suponer que el arreglo ya tiene valores **(20 puntos)**

**Ejemplo:**

Si el vector tuviera los siguientes valores: 1 3 **25 10** 4 **15** 7 6 8 **5** y **x** tuviera el valor de **5**.

La función regresaría: **55**

1. La función **promedioMultiplosX** que recibe un vector de 10 valores enteros y un entero **x**. La función deberá regresar el resultado de calcular el promedio de todos los elementos múltiplos de **x** contenidos en el arreglo. **NOTA:** Suponer que el arreglo ya tiene valores **(20 puntos)**

**Ejemplo:**

Si el vector tuviera los siguientes valores: 1 3 **25 10** 4 **15** 7 6 8 **5** y **x** tuviera el valor de **5**.

La función regresaría: **13.75**

**Matrices**

* + La función **intercambia\_columnas (matriz, renglon1, renglon2)**que recibe una matriz de números enteros y dos números: renglon1 y renglon2. La función intercambiará los valores del **renglon1** por los valores del **renglon2** de la matriz.

**Ejemplo:**Suponiendo que la matriz es de 4 renglones x 4 columnas y tiene asignados los siguientes valores:

2  5  6  4  
7  8  5  6  
3  4  5  1  
9  7  1  5

Al llamar a la funcion **intercambia\_renglones (matriz, 1, 2)** la matriz se modificaría de la siguiente forma:

2  5  6  4  
**3  4  5  1  
7  8  5  6**  
9  7  1  5

**Suma diagonal inversa.**