UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POR MI RAZA HABLARÁ EL ESPÍRITU

ENES JURIQUILLA

Ingeniería Aeroespacial

204 - Fundamentos de Programación

Ing. Andrés David Flores Ferro

Actividad en clase 22 de 25 pts	
Protocolo de práctica	**56 de 75 pts**
Portada	08 de 10 pts
Introducción	10 de 15 pts
Desarrollo	15 de 25 pts
Conclusiones	15 de 15 pts
Referencias	08 de 10 pts
CALIFICACIÓN TOTAL	**Entrega 78 pts**

Práctica 4

Pseudocódigo

Equipo 1 Grupo: 2002 Semestre 2

	Nro. Cuenta
Paul Sebastien Gutiérrez de Loza	425037061
Emilio Lara Aguilar	425117677
Mariangel Ramos González	425062412

Fecha: 28 - febrero - 2025

1/1

Fecha de revisión: 01 / 03 / 2025

- Introducción con muy poco contenido, el proposito es rescatar información relevante de clase e investigar algo adiconal con detalle
- No se atendieron correcciones en desarrollo y no se rescata para que funciona cada cosa, se describe lo que tiene contenido solamente.
- Como se menciono en clase la idea del protocolo era que conectaran mas toda la técnica y bueno aquí ya solo se encuentra el DFD y solo mencionan las modificaciones que hacen de su tabla pero no la rescatan o profundizan mas en actualizar entradas salidas y modificaciones en el proceso.
- Agregar dato de tipo de archivo consultado en referencias [Web,PDF, Libro, etc]

Introducción:

El pseudocódigo es una herramienta fundamental en la programación, ya que permite representar algoritmos sin la necesidad de usar un lenguaje de programación. Su uso facilita el diseño de soluciones al estructurar las instrucciones de manera lógica, comprensible y a diferencia de los lenguajes de programación, emplea una sintaxis simplificada, que permite enfocarse en la lógica del problema en vez de enfocarse en la sintaxis o detalles estructurales de algún lenguaje de programación.

Durante el proceso de buscar la forma correcta de escribir pseudocodigos, nos dimos cuenta que estos pueden ser estructurados y no estructurados. Los estructurados son como los que vimos en los documentos de clase y prácticas. Estos siguen un orden lógico con estructuras de control como los IF, WHILE, etc.
El pseudocódigo no estructurado no utiliza las estructuras de control IF, WHILE etc, este utiliza unas llamadas GOTO, que hacen la función de los Ifs o Whiles, haciéndolo más confuso.

Al analizar las dos opciones nos dimos cuenta que el pseudocódigo no estructurado es más difícil de entender porque mezcla condicionales. En cambio el estructurado es muy útil y fácil de comprender, ya que la indentación, sintaxis y su estructura, es muy ordenada, permitiendo una mejor interpretación y modificación.

Desarrollo:

Para unir los programas:

```
Proceso Union_de_programas
    definir opciones Como Caracter
    escribir "Bienvenido a la biblioteca de programas del grupo 2002"

    Mientras opciones ≠ "salir" Hacer
    escribir ""
    escribir ""
    escribir ""
    escribir "Para escoger un programa escriba el nombre como aparece en la lista y precione enter."
    Escribir "Aviso: Lo que aparece abajo del nombre es una breve descripción de lo que hace el programa."
    escribir "A la hora de escoger lo que quiere que haga el programa, escriba solo lo que esta dentro del paréntesis a un lado de dicha opción."
    Escribir ""
    Escribir "Lista de Programas:"
    Escribir "1- Calculadora (calc)"
    escribir "Este programa es una calculadora que puede hacer, sumas, restas, división, multiplicación y potenciación."
    Escribir ""
    Escribir "2- Distancia entre puntos (dist)"
    escribir "Este programa calcula la distancia entre dos puntos."
    Escribir ""
    Escribir "3- Ordenar números (orden)"
    escribir "Este programa ordena 3 números en orden ascendente."
    Escribir ""
    Escribir "4-salir"
    leer opciones

    Segun opciones Hace
        caso "calc":
            calculadora()
        caso "dist":
            CalcularDistancia()
        caso "orden":
            OrdenarNumeros()
        caso "salir":
            Escribir "Gracias por usar el programa, Fin del programa"
        De Otro Modo:
            escribir "Error, Intente de nuevo"
    Fin Segun

    Fin Mientras



FinProceso
```

Se hace un “mientras” para que el usuario pueda elegir entre los tres programas disponibles o solamente salir. Si ingresa la opción de salir cierra el programa, si escoge cualquiera de los tres programas abre el respectivo subprograma. Se le indica al usuario que para escoger alguna opción se debe de poner lo que está dentro del paréntesis para que este funcione, en caso de que no lo ponga saldrá error y le pedirá que ponga una opción válida.

Calcular distancia entre dos puntos

```
Algoritmo CalcularDistancia
    Definir x1, x2, y1, y2 Como real
    Definir sumx, sumy, elevx, elevy, sumt, restotal Como Real

    x1 ← 0
    y1 ← 0
    x2 ← 0
    y2 ← 0

    Escribir "Este programa calcula la distancia entre dos puntos cuando el usuario le da las coordenadas positivas"
```

```
Algoritmo CalcularDistancia
    Definir x1, x2, y1, y2 Como entero
    Definir sumx, sumy, elevx, elevy, sumt, restotal Como Entero

    x1 ← 0
    y1 ← 0
    x2 ← 0
    y2 ← 0

    Escribir "Este programa calcula la distancia entre dos puntos cuando el usuario le da las coordenadas positivas"
```

Aquí se definen las variables como números reales y se explica de qué trata el programa.

```
Repetir
    Escribir "Ingrese el valor de x1: "
    Leer x1
    Si x1 ≤ 0 Entonces
        Escribir "Error, el número debe de ser mayor que cero"
    FinSi
Hasta Que x1 > 0

Repetir
    Escribir "Ingrese el valor de y1: "
    Leer y1
    Si y1 ≤ 0 Entonces
        Escribir "Error, el número debe de ser mayor que cero"
    FinSi
Hasta Que y1 > 0
```

```
Repetir
    Escribir "Ingrese el valor de x2: "
    Leer x2
    Si x2 ≤ 0 Entonces
        Escribir "Error, el número debe de ser mayor que cero"
    FinSi
Hasta Que x2 > 0

Repetir
    Escribir "Ingrese el valor de y2: "
    Leer y2
    Si y2 ≤ 0 Entonces
        Escribir "Error, el número debe de ser mayor que cero"
    FinSi
Hasta Que y2 > 0
```

Aquí piden los datos de las coordenadas de ambos puntos y deben de cumplir con la restricción que sea mayor o igual a 0, si no lo cumple entonces saldrá un mensaje de error pidiendo que pongan el número mayor a cero

```
    sumx ← x2 - x1
    sumy ← y2 - y1
    elevx ← sumx↑2
    elevy ← sumy↑2
    sumt ← elevy + elevx
    restotal ← RAIZ(sumt)

    Escribir "La distancia entre los puntos: (", x1, ",", y1, ") y (", x2, ",", y2, ") es: ", restotal
    Escribir " Fin del programa"

FinAlgoritmo
```

Aquí se utiliza la fórmula de distancia entre dos puntos que una vez ya terminada nos arroja el mensaje: “La distancia entre los puntos: (x1,y1) y (x2,y2)” es: “resultado”, y así avisando al usuario que el programa ya ha acabado

Ordenar 3 números de menor a mayor

```
Algoritmo OrdenarNumeros
    Definir a, b, c Como Entero
    Definir p1, p2, p3 Como Entero

    a ← 0
    b ← 0
    c ← 0

    Escribir "Este programa ordena 3 números enteros positivos de menor a mayor"
```

Aquí se definen las variables cómo números enteros y se explica el propósito del programa.

```
Repetir
    Escribir "Ingrese el primer número: "
    Leer a
    Si a ≤ 0 Entonces
        Escribir "Error, por favor ingrese otro número"
    FinSi
Hasta Que a > 0

Repetir
    Escribir "Ingrese el segundo número: "
    Leer b
    Si b ≤ 0 Entonces
        Escribir "Error, por favor ingrese otro número"
    FinSi
Hasta Que b > 0

Repetir
    Escribir "Ingrese el tercer número: "
    Leer c
    Si c ≤ 0 Entonces
        Escribir "Error, por favor ingrese otro número"
    FinSi
Hasta Que c > 0
```

Aquí se piden las variables y si son menores o iguales a 0 entonces marcará error y pedirá que ingrese otro número.

El “Si” pide que “a” sea mayor a “b” y a “c”, si es verdadero este tomará la posición “p1”, y procederá a comprar qué número es mayor o menor entre “b” y “c” y se les asignará una variable ya sea “p2” o “p3”. En caso de que “a” no sea mayor entonces verá si “b” es mayor que “a” o “c”, si es verdadero, “b” será p1 y de ahí se verá cuál es mayor y cuál es menor si “a” o “c”. Se repite la misma situación c. Una vez que ya todos los valores se les haya asignado una variable el sistema finalizará con el mensaje: “El orden de los números es: “ p3, p2, p1” “Fin del

programa".

```
Si a > b Y a > c Entonces
    p1 ← a
    Si c > b Entonces
        p2 ← c
        p3 ← b
    Sino
        p2 ← b
        p3 ← c
    FinSi
Sino
    Si b > a Y b > c Entonces
        p1 ← b
        Si c > a Entonces
            p2 ← c
            p3 ← a
        Sino
            p2 ← a
            p3 ← c
        FinSi
    Sino
        p1 ← c
        Si b > a Entonces
            p2 ← b
            p3 ← a
        Sino
            p2 ← a
            p3 ← b
        FinSi
    FinSi
FinSi

Escribir "El orden de los números es: ", p3, " , ", p2, " , ", p1, " Fin del programa"

FinAlgoritmo
```

Calculadora:

```
SubProceso calculadora
    Definir n1,n2,rr Como Real
    Definir num,i,j Como Entero
    Definir c Como Cadena
    definir vector1,vector2,vector3,vector4 como real
    dimension vector1[100],vector2[100],vector3[100],vector4[100]
    escribir ""
    escribir ""
    escribir ""
    Escribir "Bienvenido a la calculadora..."
    opsi← 0
    n1←0
    n2←0
    r←0
    Mientras c ≠ "SALIR" Hacer

        Escribir "Escriba lo que desea hacer en MAYUSCULAS."
        Escribir "SALIR"
        Escribir ""
        Escribir "SUMA"
        Escribir ""
        Escribir "RESTA"
        Escribir ""
        Escribir "MULTIPLICACION"
        Escribir ""
        Escribir "DIVISION"
        Escribir ""
        Escribir "EXPONENCIAL"
        Escribir ""
```

Aquí primero se definen todas las variables que vamos a necesitar, y el vector para que el usuario pueda definir cuantos números quiere operar.
También se hace el mientras para que el usuario pueda elegir entre sumar, restar, multiplicar, división o exponenciar.

```
caso "SUMA":
    Escribir "Ingrese cuantos números quiere sumar:"
    leer num
    Para i ← 1 Hasta num Con Paso 1 Hacer
        Escribir i, ":"
        leer vector1[i]
    Fin Para

    Para j←1 Hasta num Con Paso 1 Hacer
        r ← r + vector1[j]
    Fin Para
    Escribir "El resultado es: "
    Escribir r
    num←0
    i←1
    j←1
    r←0
```

Para sumar primero se ingresa la cantidad de números que se quieren operar, esto se usa también para que el usuario pueda ingresar esa cantidad de números, y luego solamente se suman los datos guardados en cada uno de los espacios del vector establecido.

```
caso "RESTA":
    Escribir "Ingrese cuantos números quiere restar:"
    leer num
    Para i←1 Hasta num Con Paso 1 Hacer
        Escribir i
        Escribir ":"
        leer vector2[i]
    Fin Para
    r ← vector2[1]
    rr ← r
    Para j←2 Hasta num Con Paso 1 Hacer
        rr ← rr - vector2[j]
    Fin Para
    Escribir "El resultado es: "
    Escribir rr
    num←0
    i←1
    j←1
    r←0
    rr←0
```

Para restar se hace lo mismo de ingresar la cantidad de números a operar y esto también es para que el usuario pueda ingresar esa cantidad de números. Luego al primero se le restan todos los demás números.

```
r←1
Escribir "Ingrese cuantos números quiere multiplicar:"
leer num
Para i←1 Hasta num Con Paso 1 Hacer
    Escribir i
    Escribir ":"
    leer vector3[i]
Fin Para
Para j←1 Hasta num Con Paso 1 Hacer
    r ← r * vector3[j]
Fin Para
Escribir "El resultado es: "
Escribir r
num←0
i←1
j←1
r←0
```

Para multiplicar se hace lo mismo que en la suma pero en vez de sumar se resta.

```
caso "DIVISION":
    Escribir "Ingrese cuantos números quiere dividir:"
    leer num
    Para i←1 Hasta num Con Paso 1 Hacer
        Escribir i
        Escribir ":"
        leer vector4[i]
    Fin Para
    r ← vector4[1]
    rr ← r
    Para j←2 Hasta num Con Paso 1 Hacer
        rr ← rr / vector4[j]
    Fin Para
    Escribir "El resultado es: "
    Escribir rr
    num←0
    i←1
    j←1
    r←0
    rr←0
```

Para dividir se hace lo mismo que en la resta pero con la operación de dividir.

```
caso "EXPONENCIAL":
    Escribir "Solo se puede elevar una vez, un numero n a la m."
    Escribir "Ingrese el numero que quiere elevar: "
    leer n1
    Escribir "Ingrese el numero de la potencia: "
    leer n2
    r ← n1↑n2
    Escribir "El resultado es: "
    Escribir r
    n1← 0
    n2←0
    r←0
De Otro Modo:
    Escribir "Error, Reiniciando el programa"
```

En el caso de la exponenciación solo se puede hacer la operación con dos números, n^m. El primer número ingresado es n y el segundo número ingresado es m.

Conclusión:

Paul: El usar pseudocódigo no me pareció la mejor opción antes de programas, ya que suele ser muy confuso el estar revisando la indentación en cada pasa que haces, además, las estructuras como los Ifs o los Whiles, suele quedar a la interpretación de donde poner el ENTONCES y el DE LO CONTRARIO, si luego después de poner la condición o con identacion abajo del If. Como ya he programado en C y C++, se me hace mas fácil el usar paréntesis() y llaves{} porque permiten ubicar mejor el código y organizarlo mejor. Para alguien que nunca ha programado tal vez le sea tedioso el estar ubicando el código sin estas estructuras porque no se tiene un orden claro y podría confundir sus propios ifs y whiles.

Por otra parte, el no tener que estar pensando en la sintaxis del lenguaje es muy útil, porque puedes usar las palabras que conoces y conceptos que ya sabes para aplicarlos a cualquier lenguaje de programación, es decir puedes usar el mismo pseudocódigo para programar en C, Python o en Java. No soy muy fan de los pseudocódigos, pero entiendo el porqué son útiles y se deberían de usar.

Lara:Así como en los diagramas de flujo, el pseudocódigo es una forma muy útil para verificar que nuestro algoritmo funciona como un programa. Es más fácil encontrar soluciones a un problema usando el lenguaje natural, y no el lenguaje de programación. En mi opinión es una práctica muy útil.
Gel: Fue raro ya que ahora se tiene que ser específico a la hora de escribir todo y con palabras lo cuál es aún más raro, al menos para mi. También el tener que juntar los programas se puso muy específico cómo es que se tenía que hacer.

Referencias:

programacionpro.com & programacionpro.com. (2024, 16 septiembre). *¿Alguna vez te has preguntado qué es el pseudocódigo y cuáles son Leer más*. ProgramaciónPro. https://programacionpro.com/tipos-de-pseudocodigo-y-ejemplos/?utm_source=chatgpt.com#google_vignette

programacionpro.com & programacionpro.com. (2024a, septiembre 14). *¿Alguna vez te has preguntado qué es el pseudocódigo GOTO y cómo Leer más*. ProgramaciónPro. https://programacionpro.com/pseudocodigo-goto-todo-lo-que-necesitas-saber/

Maldonado, R. (2024, 5 septiembre). Instrucción goto: la causante del código espagueti. *KeepCoding Bootcamps*. https://keepcoding.io/blog/que-es-la-instruccion-goto-en-programacion/

3.5. Sentencia goto - AprendeAProgramar.com. (s. f.-b). https://www.aprendeaprogramar.com/mod/resource/view.php?id=610