	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: Marco Antonio Martínez Quintana

Asignatura: Fundamentos de Programación

Grupo: 3

No de Práctica(s): 5

Integrante(s): Ramírez Pérez Daniela Itzel

No. de Equipo de cómputo empleado: No aplica

No. de Lista o Brigada: 36

Semestre: 1er

Fecha de entrega: Viernes 6 de noviembre

Observaciones:

CALIFICACIÓN: _____

Objetivo

Elaborar pseudocódigos que representen soluciones algorítmicas empleando la sintaxis y semánticas adecuadas.

Introducción

Cuando cualquier problema se analiza se debe de crear su algoritmo y finalmente se llega a la codificación de dicho algoritmo.

Para representar la forma escrita del algoritmo se usa el pseudocódigo, que en si usa la sintaxis con las siguientes reglas:

Sintaxis de pseudocódigo

1. Alcance del programa: Los limites están indicado por INICIO y FIN.

2. **Palabras reservadas con mayúsculas**: Las palabras propias del pseudocódigo deben de ser escritas en mayúsculas.

3. Sangría o tabulación: El pseudocódigo debe tener diversas alineaciones para que el código sea más fácil de entender y depurar.

4. Lectura / escritura: Para indicar lectura de datos se utiliza la etiqueta LEER. Para indicar

escritura de datos se utiliza la etiqueta ESCRIBIR.

```
ESCRIBIR "Ingresar la altura del
polígono"
LEER altura
```

5. Declaración de variables: la declaración de variables la definen un identificador (nombre), seguido de dos puntos, seguido del tipo de dato, es decir:

<nombreVariable>:<tipoDeDato>

Los tipos de datos que se pueden utilizar son:

ENTERO	Valor entero positivo y/o negativo
REAL	Valor con punto flotante y signo
BOOLEANO	Valor de dos estados: verdadero o falso
CARACTER	Valor tipo carácter
CADENA	Cadena de caracteres

Formas para declarar datos	EJEMPLO
Mismo tipo de dato: Se usan *arreglos* para declarar de una variable la cantidad que se requieren.	<nombreVariable>[cantidad]:<tipoDeDato>
Tipo de dato compuesto: Se usa para contener uno o más tipos de datos simples diferentes.	<nombreRegistro>:REG <nombreVariable_1>:<tipoDeDato> ... <nombreVariable_N>:<tipoDeDato> FIN REG
Tipo registro: Se usa para indicar el nombre del registro y el nombre de la variable.	domicilio:REG calle: CADENA número: ENTERO ciudad: CADENA FIN REG
Variables constantes: Se usa la palabra reservada CONST para indicar que un identificador no cambia su valor durante todo el pseudocódigo	NUM_MAX := 1000: REAL, CONST

6. Operadores aritméticos: Se tiene la posibilidad de utilizar operadores aritméticos y lógicos:

suma	+
resta	-
multiplicación	*
división real	/
División entera	div
módulo	mod
exponenciación	^
asignación	:=

igualdad	=
y-lógica o AND	&
o-lógica u OR	|
negación o NOT	!
relaciones de orden	<, >, <=, >=
diferente	<>

La tabla de verdad de los operadores lógicos AND, OR y NOT se describe a continuación:

A	B	A & B	A | B	!A
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

7. Notación de camello. Para nombrar variables y nombres de funciones se debe hacer uso de la notación de camello donde los nombres de cada palabra empiezan con mayúscula y el resto se escribe con minúsculas.

Tipos de Notación de camello	Ejemplos en variables	Ejemplos en funciones
Lower camel case que en la cual la primera letra de la variable inicia con minúscula.	realAreaDelTriangulo: REAL	calcularArea()
Upper camel case en la cual todas las palabras inician con mayúscula.	EnteroRadioCirculo: REAL	obtenerPerimetro()

Estructuras de control de flujo

Las estructuras de control de flujo permiten la ejecución condicional y la repetición de un conjunto de instrucciones. Hay 3 tipos de estructuras de control de flujo:

Estructura de control secuencial	Son las sentencias o declaraciones que se realizan una a continuación de otra en el orden en el que están escritas.	INICIO x : REAL x := 5.8 x := x * 2 FIN

	DESCRIPCIÓN	EJEMPLO
Estructuras de control condicionales	Permiten evaluar una expresión lógica que puede ser verdadera o falsa y, dependiendo del resultado, solo se pude realizar una acción u otra.	
Estructura condicional SI	Se evalúa la expresión lógica y si se cumple se ejecutan las instrucciones del bloque [Acción]. Si no se cumple la condición, se continúa con el flujo normal del programa. SI condición ENTONCES [Acción] FIN SI	INICIO a,b: ENTERO a := 3 b := 2 SI a > b ENTONCES ESCRIBIR "a es mayor" FIN SI FIN
La estructura condicional completa es SI-DE LO CONTRARIO	Se evalúa la expresión lógica y si la condición es verdadera se ejecutan las instrucciones del bloque SI [Acciones SI]. Si no se cumple la condición se ejecutan las instrucciones del bloque DE LO CONTRARIO [Acciones DE LO CONTRARIO]. Al final el pseudocódigo sigue su flujo normal. SI cond_booleana ENTONCES [Acciones SI] FIN SI DE LO CONTRARIO [Acciones DE LO CONTRARIO] FIN DE LO CONTRARIO	INICIO a,b:ENTERO a := 3 b := 5 SI a > b ENTONCES ESCRIBIR "a es mayor" FIN SI DE LO CONTRARIO ESCRIBIR "b es mayor" FIN DE LO CONTRARIO FIN
La estructura condicional SELECCIONAR-CASO	Evalúa la expresión lógica de la variable que está entre paréntesis y comprueba si es igual al valor que está definido en cada caso. Si la variable no tiene el valor de ningún caso se va a la instrucción por defecto (DEFECTO). SELECCIONAR (variable) EN CASO valor1 -> [Acción] CASO valor2 -> [Acción] CASO valor3 -> [Acción] DEFECTO -> [Acción] FIN SELECCIONAR	INICIO a :ENTERO a := 1 SELECCIONAR (a) EN CASO 1 -> ESCRIBIR "Iniciar sesión." CASO 2 -> ESCRIBIR "Registrarse." CASO 3 -> ESCRIBIR "Salir." DEFECTO -> ESCRIBIR "Opción inválida." FIN SELECCIONAR FIN

	DESCRIPCIÓN	EJEMPLO
Estructuras de control iterativas o repetitivas	Las estructuras de control de flujo iterativas o repetitivas permiten ejecutar una serie de instrucciones mientras se cumpla la expresión lógica. Existen dos tipos de expresiones cíclicas *MIENTRAS y HACER- MIENTRAS.*	
La estructura MIENTRAS	Evalúa la expresión lógica de la condición y si ésta es verdadera procede a ejecutar el bloque de instrucciones de la estructura, de lo contrario rompe el ciclo y continúa el flujo normal del pseudocódigo. MIENTRAS condición ENTONCES [Acción] FIN MIENTRAS El final de la estructura lo determina la etiqueta FIN MIENTRAS.	INICIO valorInicial,valorFinal:ENTERO valorInicial=0 valorFinal=3 MIENTRAS valorInicial < valorFinal ESCRIBIR valorInicial valorInicial := valorInicial + 1 FIN MIENTRAS FIN
La estructura HACER- MIENTRAS	Primero siempre se ejecuta las instrucciones descritas en la estructura y al final valida la expresión lógica. HACER [Acción] MIENTRAS condición Si la condición se cumple vuelve a ejecutar las instrucciones de la estructura, de lo contrario rompe el ciclo y sigue el flujo.	INICIO valorInicial,valorFinal:ENTERO valorInicial=0 valorFinal=3 MIENTRAS valorInicial < valorFinal ESCRIBIR valorInicial valorInicial := valorInicial + 1 FIN

Funciones

Cuando la solución de un problema es muy compleja se el diseño descendente. Este diseño divide al problema en subprocesos más sencillos que juntos forman la solución completa.
A estos subprocesos se les llaman métodos o funciones.

Una función está constituida por un identificador de función (nombre), de cero a n parámetros de entrada y un valor de retorno:

```
INICIO
    FUNC identificador (var:TipoDato,..., var:TipoDato) RET: TipoDato
        [Acciones]
    FIN FUNC
FIN
```

El identificador es el nombre con el que llama a la función. Las funciones pueden recibir algún parámetro de tipo de dato como entrada que se debe incluir entre los paréntesis.

Todas las estructuras de control de flujo (secuencial, condicional y repetitivas o iterativas) deben ir dentro de alguna función.

```
INICIO
FUNC principal (vacío) RET: vacío
    a, b, c: ENTERO
    a := 5
    b := 24
    c := sumar(a, b)
    ESCRIBIR c
  FIN FUNC
FIN

INICIO
```

Actividades

1. Realizar un diagrama de flujo y pseudocódigo que determine el color del semáforo COVID en base a una muestra de 100 individuos:
   - Si hay más de 80 individuos con COVID el color del semáforo es rojo
   - Si hay de 51 a 80 individuos con COVID el color del semáforo es naranja
   - Si hay de 1 a 50 individuos con COVID el color del semáforo es amarillo
   - Si no hay individuos con COVID el color del semáforo es verde

```
INICIO
   numero: ENTERO
SELECCIONAR (numero) EN
CASO 1 -> numero >= 80 && numero <= 100
          ESCRIBIR "Semáforo Rojo"

CASO 2 - >  numero >= 51 && numero <= 80
          ESCRIBIR “Semáforo Naranja"

CASO 3 ->  numero >= 1 && numero <= 50
          ESCRIBIR “Semáforo Amarillo"

DEFECTO ->
          ESCRIBE "Semáforo Verde"
FIN SELECCIONAR
FIN
```

2. Realizar el pseudocódigo que calcule dado un número el cálculo de su factorial:

Ejemplo:

1! = 1

2! = 2

3! = 6

4! = 24

```
INICIO

n : REAL
n! = n * (n-1)!

FIN
```

Conclusión

Gracias a la practica realizada logramos visualizar aún más como nuestros algoritmos están pasando de ser oraciones a un lenguaje que la computadora ya es capaz de leer y realizar. También ya entendemos como empezar a redactar nuestro código al considerar que cada valor es importante cuando nuestro software quiere leer la información dada.