| | |
|---|---|
| | **Carátula para entrega de prácticas** |
| Facultad de Ingeniería | Laboratorio de docencia |

# Laboratorios de computación salas A y B

| | |
|---|---|
| *Profesor:* | M. I. Marco Antonio Martínez Quintana |
| *Asignatura:* | Fundamentos de Programanción |
| *Grupo:* | 03 |
| *No de Práctica(s):* | 5 |
| *Integrante(s):* | González Ramírez Octavio Alberto |
| *No. de Equipo de cómputo empleado:* | No aplica |
| *No. de Lista o Brigada:* | 19 |
| *Semestre:* | Primero |
| *Fecha de entrega:* | Lunes 28 de septiembre de 2020 |
| *Observaciones:* | |

CALIFICACIÓN:_____

# Pseudocódigo

## Objetivo

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

## Introducción

Una vez que un problema dado ha sido analizado (se obtiene el conjunto de datos de entrada y el conjunto de datos de salida esperado) y se ha diseñado un algoritmo que lo resuelva de manera eficiente (procesamiento de datos), se debe proceder a la etapa de codificación del algoritmo.

Para que la solución de un problema (algoritmo) pueda ser codificada, se debe generar una representación del mismo. Una representación algorítmica elemental es el pseudocódigo.

Un pseudocódigo es la representación escrita de un algoritmo, es decir, muestra en forma de texto los pasos a seguir para solucionar un problema. El pseudocódigo posee una sintaxis propia para poder realizar la representación del algoritmo (solución de un problema).

## Sintaxis de pseudocódigo

El lenguaje pseudocódigo tiene diversas reglas semánticas y sintácticas. A continuación, se describen las más importantes:

1. Alcance del programa: Todo pseudocódigo está limitado por las etiquetas de INICIO y FIN. Dentro de estas etiquetas se deben escribir todas las instrucciones del programa.
2. Palabras reservadas con mayúsculas: Todas 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. La lectura de

datos se realiza, por defecto, desde el teclado, que es la entrada estándar del sistema. La escritura de datos se realiza, por defecto, en la pantalla, que es la salida estándar del sistema.

**Estructuras de control de flujo**

Las estructuras de control de flujo permiten la ejecución condicional y la repetición de un conjunto de instrucciones. Existen 3 estructuras de control: secuencial, condicional y repetitivas o iterativas.

*Estructura de control secuencial*

Las estructuras de control secuenciales son las sentencias o declaraciones que se realizan una a continuación de otra en el orden en el que están escritas.

**Estructuras de control condicionales (o selectivas)**

Las estructuras de control condicionales permiten evaluar una expresión lógica (condición que puede ser verdadera o falsa) y, dependiendo del resultado, se realiza uno u otro flujo de instrucciones. Estas estructuras son mutuamente excluyentes (o se ejecuta una acción o se ejecuta la otra)

La estructura de control de flujo más simple es la estructura condicional SI, su sintaxis es la siguiente:

```
SI condición ENTONCES
    [Acción]
FIN SI
```

Se evalúa la expresión lógica y si se cumple (si la condición es verdadera) se ejecutan las instrucciones del bloque [Acción]. Si no se cumple la condición, se continúa con el flujo normal del programa

**Estructuras de control iterativas o repetitivas**

Las estructuras de control de flujo **iterativas** o **repetitivas** (también llamadas cíclicas) 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 (WHILE en inglés) primero valida 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.

**Funciones**

Cuando la solución de un problema es muy compleja se suele ocupar el diseño descendente (divide y vencerás). Este diseño implica la división de un problema en varios 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 o no recibir algún(os) parámetro(s) (tipo(s) de dato(s)) como entrada; si la función recibe alguno se debe incluir entre los paréntesis. Todas las funciones pueden regresar un valor al final de su ejecución (el resultado).

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

**ACTIVIDAD**

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

```
Proceso sin_titulo
    Definir N Como Real;
    Leer N;
    Si N>80 Entonces
        Escribir “el semáforo está en rojo”;
    SiNo
        Si N<=80 y N>=51 Entonces
            Escribir “el semáforo está en naranja”;
        SiNo
            Si N<=50 y N>=1 Entonces
                Escribir “el semáforo está en amarillo”;
            SiNo
                Escribir “el semáforo está en verde”;
            FinSi
        FinSi
    FinSi
FinProceso
```

**ACTIVIDAD**

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

```
Proceso sin_titulo
    Definir Fact, Fac, N Como Real;
    Leer N;
    Fact←N;
    Repetir
        N←N-1;
        Fact←Fact*N;
    Hasta Que n=1
    Escribir “La factorización de N es igual a”, Fact;
FinProceso
```