**Materia**: Resolución de Problemas y Algoritmos.

Carreras: Ingeniería en Computación y Profesorado en Ciencias de la Computación.

**Práctico Nº 6:** PSeInt. Modularización

Tabla con los días para trabajar en este práctico:

| Semana | Fecha | | Teoría | Práctico |
|---|---|---|---|---|
| 10 | Lunes | 6/10/25 | **Subalgoritmos** | |
| | Martes | 7/10/25 | | |
| | Miércoles | 8/10/25 | | P6 Subalgoritmos |
| | Jueves | 9/10/25 | | |
| | Viernes | 10/10/25 | | |
| 11 | Lunes | 13/10/25 | | P6 Subalgoritmos |
| | Martes | 14/10/25 | | |
| | Miércoles | 15/10/25 | | P6 Subalgoritmos |
| | Jueves | 16/10/25 | | |
| | Viernes | 17/10/25 | | |

**Ejercicio 1: Versión 1 y modularización.**

Diseñar un algoritmo que permita trabajar con una secuencia de hasta 10 caracteres ingresados por el usuario. El usuario puede consultar la cantidad de veces que se repite un caracter en particular. Además el algoritmo debe mostrar la cadena de caracteres.

**Se pide:**

a) **Completar** la versión 1 del algoritmo principal que dé solución al problema planteado:

**Versión 1 del algoritmo principal (con modularización aplicada)**

T1: Declarar un arreglo **Cad** de dimensión 10 y de tipo caracter. Declarar la variable **i** de tipo entero para usar como índice del arreglo. Declarar la variable **limS** de tipo entero para la cantidad de caracteres que quiere ingresar el usuario. Declarar la variable **cant** de tipo entero y la variable **car** de tipo caracter.

T2: **Invocar** al procedimiento **IngresoCad** con los parámetros **Cad**, **limS** (en limS se **devuelve por referencia** la cantidad de caracteres almacenados)

T3: Invocar a la función **CarRep** con los parámetros **Cad** y **limS**. La función **retorna** la cantidad de repeticiones de un caracter consultado por el usuario en la variable **cant**.

T4: Mostrar la cantidad retornada por la tarea T3.

T5:_____

b) **Escribir** el algoritmo completo que permita resolver el enunciado definiendo los siguientes subalgoritmos:

- Definir el procedimiento **IngresoCad** que realice la carga correcta de los datos.
- Definir la función **CarRep** que cuente y devuelva la cantidad de repeticiones de un caracter ingresado por el usuario.
- Definir un procedimiento que muestre una cadena de caracteres.

**Ejercicio 2: Versión 1 y modularización.**

Se desea procesar una lista de hasta 15 números enteros ingresados por el usuario. El objetivo es identificar aquellos que sean múltiplos de 5 y de 2 al mismo tiempo. Cada vez que se detecte un número que cumpla esta condición, se debe mostrar el siguiente mensaje:

"El número x es múltiplo de 5 y de 2",
reemplazando x por el número correspondiente.

**Se pide:**

a) Analizar cada tarea e identificar qué hace cada subalgoritmo en la versión 1 dada.

b) Escribir el algoritmo completo en pseudocódigo, incluyendo la definición de los procedimientos **IngresoNumeros** y **MostrarMultiplos**.

c) Aplicar correctamente el concepto de modularización, reutilizando código y evitando repeticiones innecesarias.

## Versión 1 del algoritmo principal (con modularización aplicada)

**T1:** Declarar un arreglo **Num** de dimensión 15 y de tipo entero. Declarar la variable **i** de tipo entero para usar como índice del arreglo. Declarar la variable **limS** de tipo entero para representar la cantidad de números almacenados.

**T2:** Invocar al procedimiento **IngresoNumeros** con los parámetros Num y limS. (en limS se **devuelve por referencia** la cantidad de números almacenados)

**T3**: Invocar al procedimiento **FiltrarMultiplos** con los parámetros Num, limS. En este procedimiento se evalúan los múltiplos de 5 y de 2 y se muestra un mensaje por cada número que cumpla la condición.

**Ejercicio 3: Modularización y manejo de arreglos con datos de estudiantes**.

Teniendo en cuenta la solución que usted planteó en el práctico 5 para el ejercicio 4.
**Se pide:**

a) **Reescribir la versión 1** modificando la escritura para aplicar el criterio de modularización.

b) **Escribir el algoritmo completo** que resuelva el enunciado, definiendo los subalgoritmos (procedimientos o funciones) necesarios para resolver cada tarea planteada en la versión 1.

**Ejercicio 4: Modularización y manejo de arreglos con datos de hotel**.

Se necesita diseñar un algoritmo para trabajar con datos de habitaciones de un hotel: número de habitación, capacidad (2 o 4), y el costo de la habitación por día. El hotel cuenta con 50 habitaciones. El algoritmo debe:

- Ingresar y almacenar los datos de las habitaciones.
- Calcular e informar la cantidad de habitaciones que tienen capacidad 4.
- Calcular e informar el porcentaje de habitaciones con costo mayor a $55000 por día.
- informar los datos de una habitación consultada por el usuario. Por ejemplo, el usuario solicita los datos de la habitación 3, el algoritmo por pantalla muestra el número de habitación,la capacidad y el costo por día.

**Se pide:**

a) Graficar el o los arreglos para almacenar los datos de las 50 habitaciones del hotel, especificando para cada arreglo *su nombre, dimensión y tipo de datos.*

b) Escribir la versión 1 del algoritmo principal que dé solución al problema.

c) Escribir la versión final, definiendo los procedimientos y funciones necesarias para cada tarea.

**Ejercicio 5: Subalgoritmos, ejecución y Diagrama de Flujo**.

Teniendo en cuenta el código dado:

```
SubAlgoritmo ingreso ( nro por referencia)
    Escribir "Ingrese un nro"
    Leer Nro
FinSubalgoritmo
Subalgoritmo aux <- calculo (nro)
definir aux como entero
    si nro < 0 entonces
        aux ← abs(nro)
    sino
        aux ←    nro ↑ 2
    FinSi
FinSubAlgoritmo
Algoritmo p5_ej3
    Definir nro, val, k Como Entero
    dimension val[3]
    PARA k← 1 HASTA 3 CON PASO 1 HACER
        ingreso(nro)
        val[k] ← calculo(nro)
        Escribir "el nro ingresado fue: ", nro
        Escribir " El número calculado fue: ", val[k]
    FinPara
FinAlgoritmo
```

**Se pide:**

a) Identificar programa principal, procedimiento y función. Realice los diagramas de flujo para cada uno.

b) Ejecutar el programa para los valores -4, 5 y 10.

c) Explicar con sus palabras qué hace el programa.

**Ayuda:** para hacer la tabla de ejecución, recuerde hacer:

- Primero una columna para las acciones, luego
- el conjunto de columnas para las variables del algoritmo principal, luego
- las columnas para las variables del subalgoritmo ingreso, luego
- las columnas para las variables del subalgoritmo calculo y finalmente la columna para la Pantalla.

**Ejercicio 6: Modularización, Redes sociales y notas.**

Un grupo de 10 estudiantes universitarios registró durante una semana el promedio de horas diarias que usan redes sociales y su última calificación en una materia. Se desea analizar esta información para observar si existe una posible relación entre el uso intensivo de redes sociales y el bajo rendimiento académico.

Para ello se trabajará con dos arreglos:

- **Redes**: arreglo de 10 elementos que almacena las horas promedio por día que cada estudiante usa redes sociales.
- **Notas**: arreglo de 10 elementos que almacena la última nota obtenida (entre 0 y 10) de cada estudiante.

**Objetivo:**
Se desea generar un nuevo arreglo, llamado **EnRiesgo**, que contenga el número de posición (índice) de los estudiantes que: "*usan más de 4 horas diarias redes sociales y tienen una calificación menor a 6".*

**Se pide:**

a) Analizar el problema y graficar los arreglos a utilizar con un ejemplo en cada uno.
b) Redactar la Versión 1 del algoritmo principal, identificando claramente las tareas principales.
c) Escribir el algoritmo completo en PSeInt, utilizando un procedimiento que reciba como parámetros los arreglos **Redes** y **Notas**, y genere el nuevo arreglo **EnRiesgo**.

**Ejercicio 7: Modularización y Diagrama de Flujo.**

La función **EsMayorEdad** recibe la edad de una persona y determina si es mayor de edad. Se considera mayor de edad a toda persona que tenga 18 años o más.

**Se pide:**

a) Realizar el diagrama de flujo completo de la función **EsMayorEdad** dada. No es necesario dibujar el algoritmo principal, solo la lógica interna de esta función.

```
Subalgoritmo EsMayor <- Calcular (edad)
        definir EsMayor como logico
        Si edad >= 18 Entonces
            EsMayor <- Verdadero
        Sino
            EsMayor<- Falso
        FinSi
FinSubalgoritmo
```

**Ejercicio 8: Modularización y Diagrama de Flujo.**

El procedimiento **MostrarPares** recorre un arreglo de números enteros y muestra por pantalla todos los elementos que son números pares.

**Se pide:**

a) Realizar el diagrama de flujo completo del procedimiento **MostrarPares** dado. No es necesario dibujar el algoritmo principal, solo la lógica interna de este procedimiento.

```
SubAlgoritmo MostrarPares(numeros, ls)
    Definir i Como Entero
    Para i <- 0 hasta ls con paso 1  hacer
        Si numeros[i] MOD 2 = 0 Entonces
            Escribir numeros[i], " es par."
        FinSi
    FinPara
FinSubAlgoritmo
```