



**UNIVERSIDAD
DE ANTIOQUIA**

LÓGICA Y REPRESENTACIÓN I

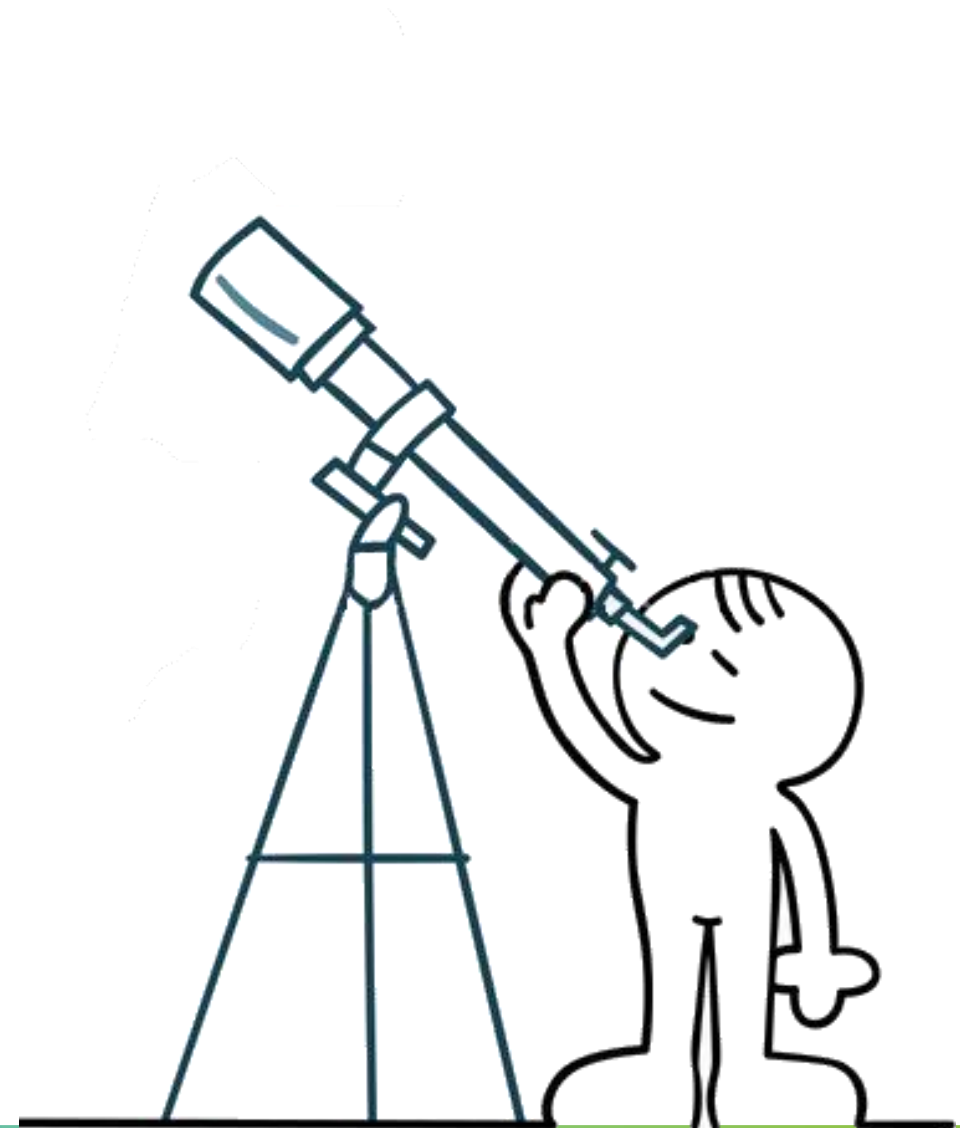
PROGRAMA DE INGENIERÍA DE SISTEMAS

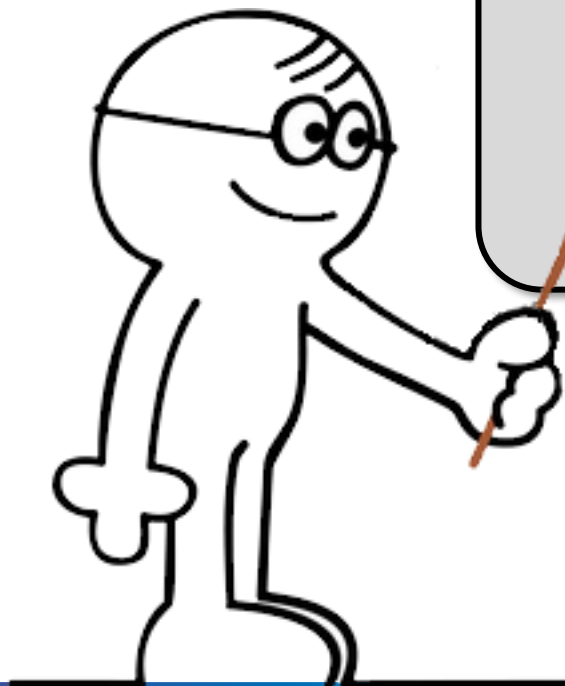
Material desarrollado por **Carlos Andrés Mera Banguero**

<https://github.com/carlosmera20/>

🦋 ESTRUCTURAS REPETITIVAS

- ✓ Introducción a las estructuras repetitivas
- ✓ Ciclo ***Para-Haga***
- ✓ Ciclo ***Mientras - Haga***





RECORDANDO: ESTRUCTURAS CONTROL

ESTRUCTURAS DE CONTROL

Las estructuras de control permiten cambiar el flujo de ejecución de un programa, haciendo que ciertos bloques de código se ejecuten si y solo si se dan unas condiciones particulares.



SELECTIVAS



Que permiten determinar si un bloque se ejecuta o no según si una **condición** es verdadera o falsa.

REPETITIVAS



Que permiten que un bloque de código se ejecuta múltiples veces según si una **condición** es verdadera o falsa.

ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA

🦋 TIPOS DE ESTRUCTURAS DE CONTROL REPETITIVAS: existen dos tipos básicos de estructuras de repetición:

CICLO PARA-HAGA

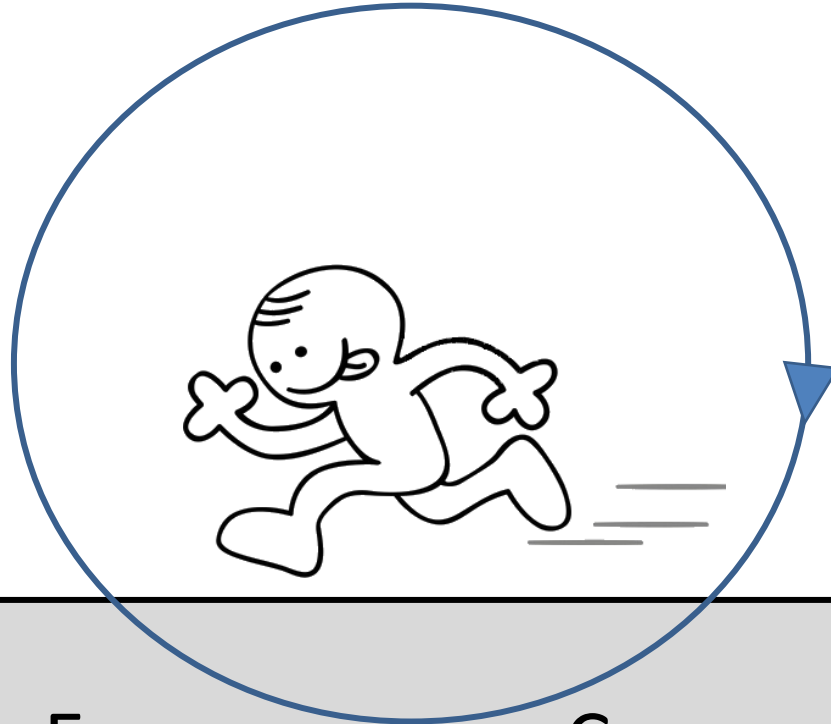
Ciclo **PARA** se usa cuando se conoce explícitamente el número de veces que se debe repetir un bloque de código. Este ciclo usa una variable contadora que es la que controla el ciclo.



CICLO MIENTRAS-HAGA

Ciclo **MIENTRAS**, se repite mientras que condición que se evalúa sea verdadera. Este ciclo usa o bien una variable contadora o centinela para controlar el ciclo.





ESTRUCTURAS DE CONTROL:
ESTRUCTURA REPETITIVA MIENTRAS-HAGA

✚ **CICLO MIENTRAS-HAGA:** se utiliza para repetir un bloque de código mientras que la condición que se evalúa sea verdadera. En este sentido, el ciclo Mientras se puede usar **cuando no se sabe cuántas veces se debe repetir el bloque de código**.

🕒 **A tener en cuenta:**

- ✓ Si la condición siempre es verdadera, el ciclo nunca terminará. En consecuencia, siempre se debe hacer algo para que la variable de control cambie dentro del ciclo.
- ✓ Es importante tener cuidado al usar ciclos mientras para evitar ciclos infinitos. Un ciclo infinito es un ciclo que se repite indefinidamente porque la condición nunca se vuelve falsa.
- ✓ A diferencia de un ciclo Para, la variable de control se debe inicializar manualmente antes del ciclo y se debe actualizar su valor dentro del ciclo.

ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA

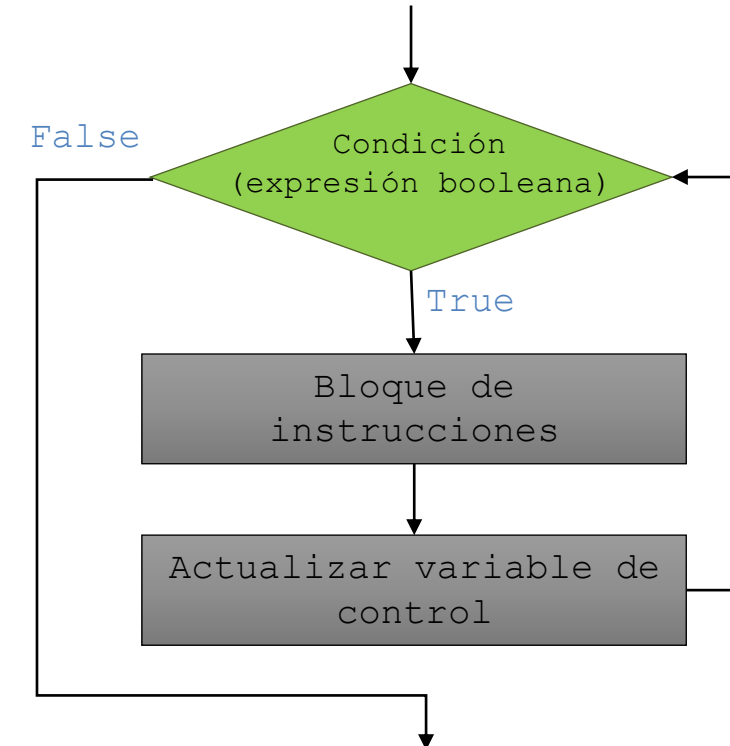
🦋 **CICLO MIENTRAS-HAGA:** esta es la estructura general del ciclo mientras:

Algoritmo

```
Mientras (condición) Haga  
    instrucción_1  
    instrucción_2  
    ...  
    Actualizar_var_de_control  
Fin_Mientras
```

Python

```
while (condición):  
    instrucción_1  
    instrucción_2  
    ...  
    Actualizar_var_de_control
```



ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA



EJERCICIO: escriba un algoritmo que muestre en la pantalla las palabras
“Hola Mundo” 10 veces.



👉 **DISEÑO DE LA SOLUCIÓN:** como el algoritmo es bastante sencillo lo escribimos en una sola clase :

PSEUDOCÓDIGO

```
Clase HolaMundo
    publico vacio saludar()
        Escribir("Hola Mundo!")
        Escribir("Hola Mundo!")
        Escribir("Hola Mundo!")
        Escribir("Hola Mundo!")
        Escribir("Hola Mundo!")
        Escribir("Hola Mundo!")
        Escribir("Hola Mundo!")
        Escribir("Hola Mundo!")
        Escribir("Hola Mundo!")
    Fin Metodo
Fin_Clase
```

Python

```
class HolaMundo:

    def saludar(self):
        print("Hola Mundo!")
        print("Hola Mundo!")
        print("Hola Mundo!")
        print("Hola Mundo!")
        print("Hola Mundo!")
        print("Hola Mundo!")
        print("Hola Mundo!")
        print("Hola Mundo!")
        print("Hola Mundo!")
        print("Hola Mundo!")
```

ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA



EJERCICIO: escriba un algoritmo que muestre en la pantalla 10 veces las palabras “Hola Mundo”, usando un ciclo mientras.



🦋 **CICLO MIENTRAS - EJERCICIO** : escriba un algoritmo que muestre 10 veces en pantalla el mensaje “Hola Mundo”:

PSEUDOCÓDIGO

```
Clase HolaMundo10VecesWhile  
    publico vacio saludar()  
        Entero cont = 1  
  
        Mientras (cont <= 0) Haga  
            Escribir(“Hola Mundo!”)  
        Fin_Mientras  
    Fin_Metodo  
Fin_Clase
```



NOTA: El bucle **NUNCA SE EJECUTA** porque la condición es **false** desde el principio

🦋 **CICLO MIENTRAS - EJERCICIO** : escriba un algoritmo que muestre 10 veces en pantalla el mensaje “Hola Mundo”:

PSEUDOCÓDIGO

```
Clase HolaMundo10VecesWhile
publico vacio saludar()
    Entero cont = 1
    Mientras (cont != 10) Haga
        Escribir(“Hola Mundo!”)
        cont = cont + 2
    Fin_Mientras
Fin_Metodo
Fin_Clase
```



NOTA: El bucle **NUNCA TERMINA** porque la condición nunca llega a ser **false**

ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA

🦋 **CICLO MIENTRAS - EJERCICIO** : escriba un algoritmo que muestre 10 veces en pantalla el mensaje “Hola Mundo”:

PSEUDOCÓDIGO

```
Clase HolaMundo10VecesWhile
    publico vacio saludar()
        Entero cont = 1

        Mientras (cont <= 10) Haga
            Escribir("Hola Mundo!")
        Fin_Mientras
        cont += 1
    Fin_Metodo
Fin_Clase
```



NOTA: El bucle **NUNCA TERMINA** porque la variable del ciclo nunca se actualiza y por tanto la condición nunca llega a ser **false**

🦋 **CICLO MIENTRAS - EJERCICIO** : escriba un algoritmo que muestre 10 veces en pantalla el mensaje “Hola Mundo”:

PSEUDOCÓDIGO

```
Clase HolaMundo10VecesWhile
publico vacio saludar()
    Entero i
    i=1
    Mientras (i<=10) Haga
        Escribir(“Hola Mundo!”)
        i = i+1
    Fin_Mientras
Fin_Metodo
Fin_Clase
```



🦋 **CICLO MIENTRAS - EJERCICIO** : escriba un algoritmo que muestre 10 veces en pantalla el mensaje “Hola Mundo”:

PYTHON

```
class HolaMundo:
    def saludar(self):
        i=1
        while (i <= 10):
            print("Hola Mundo!")
            i+=1 #Incremente de uno en uno la variable de control
```



ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA



EJERCICIO: Desarrolle un algoritmo que muestre en la pantalla los números múltiplos de 3, los números múltiplos de 7 y los múltiplos de 3 y 7 que están entre 1 y un número ingresado por el usuario.



🦋 **EJERCICIO:** Desarrolle un algoritmo que muestre todos los números múltiplos de 3 y/o de 7 que están entre 1 y un número ingresado por el usuario:

PSEUDOCÓDIGO

```
Clase Multiplos3y7
publico vacio mostrar_multiplos()
Entero i, n

Escribir("Ingrese un número: ")
Leer(n)
i=1
Mientras (i <= n) haga
    Si (i MOD 3 == 0) and (i MOD 7 == 0) Entonces
        Escribir(i, " es múltiplo de 3 y 7")
    Sino
        Si (i MOD 3 == 0) Entonces
            Escribir(i, " es múltiplo de 3")

        //continua al otro lado ...
```

```
Sino
    Si (i MOD 7 == 0) Entonces
        Escribir(i, " es múltiplo de 7")
    Fin_si
Fin_si
Fin_si

i= i+1

Fin_Mientras
Fin_Metodo
Fin_Clase
```

🏆 **EJERCICIO:** Desarrolle un algoritmo que muestre todos los números múltiplos de 3 y/o de 7 que están entre 1 y un número ingresado por el usuario:

PYTHON

```
class Multiplos3y7:
    def mostrar_multiplos(self):

        i=1
        n = int(input("Ingrese un número:"))

        while (i <= n):
            if ((i % 3 == 0) and (i % 7 == 0)):
                print(f"{i} es múltiplo de 3 y 7")
            else:
                if (i % 3 == 0):
                    print(f"{i} es múltiplo de 3")
                else:
                    if (i % 7 == 0):
                        print(f"{i} es múltiplo de 7")
            i= i+1
```

ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA



EJERCICIO: Escriba un algoritmo que pida un número entre 0 y 100, incluidos. Si el número ingresado está por fuera del rango, el número de debe volver a pedir hasta que satisfaga la condición, cuando el número esté en el rango correcto se debe mostrar y terminar el programa



🏆 **EJERCICIO:** Escriba un algoritmo que pida un número entre 0 y 100. Si el número ingresado está por fuera del rango, el número de debe volver a pedir hasta que satisfaga la condición, cuando el número esté en el rango correcto se debe mostrar y terminar el programa :

PSEUDOCÓDIGO

Clase NumeroEntre1y100

publico vacio mostrar_numeros()

Entero n = -1

Mientras (n < 0 or n > 100) haga

 Escribir("Ingrese un número: ")

 Leer(n)

 Si (n < 0 or n > 100) Entonces

 Escribir("Ingresó un número por fuera del rango. Vuelva a intentarlo!")

 Fin_si

Fin_Mientras

Escribir("El número ingresado es ", n)

Fin_Metodo

Fin_Metodo

🏆 **EJERCICIO:** Escriba un algoritmo que pida un número entre 0 y 100. Si el número ingresado está por fuera del rango, el número de debe volver a pedir hasta que satisfaga la condición, cuando el número esté en el rango correcto se debe mostrar y terminar el programa:

PYTHON

```
class NumeroEntre1y100:
    def mostrar_numeros(self):
        n=-1
        while (n < 0 or n > 100):
            n = int(input("Ingrese un número:"))

            if (n < 0 or n > 100):
                print(f"Ingresó un número por fuera del rango. Vuelva a intentarlo!")

        print(f"El número ingresado es {n}")
```

ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA



EJERCICIO: Desarrolle un algoritmo que calcule el promedio de los números ingresados por el usuario. Como no se sabe cuántos números se ingresarán debe preguntar al usuario en cada iteración si desea continuar o no.



ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA

🏆 **EJERCICIO:** Desarrolle un algoritmo que calcule el promedio de los números ingresados por el usuario. Como no se sabe cuántos números se ingresarán debe preguntar al usuario en cada iteración si desea continuar o no.

PSEUDOCÓDIGO

Clase PromedioNumeros

```
publico vacio promediar_numeros()
```

```
Entero n, i = 0, suma = 0
```

```
Cadena continuar = "Si"
```

```
Mientras (continuar == "Si" or continuar == "S") haga
```

```
    Escribir("Ingrese un número: ")
```

```
    Leer(n)
```

```
    suma += n
```

```
    i++
```

```
    Escribir("Desea continuar? (Si/No): ")
```

```
    Leer(continuar)
```

```
Fin_Mientras
```

```
    Escribir("El promedio de los números ingresados es: "+(suma/i))
```

```
Fin_Metodo
```

```
Fin_Metodo
```


🦋 **EJERCICIO:** Desarrolle un algoritmo que calcule el promedio de los números ingresados por el usuario. Como no se sabe cuántos números se ingresarán debe preguntar al usuario en cada iteración si desea continuar o no.

PYTHON

```
class PromedioNumeros:
    def promediar_numeros(self):
        i = 0
        suma = 0
        continuar = "Si"

        while (continuar == "Si" or continuar == "S"):
            n = int(input("Ingrese un número: "))
            suma += n
            i += 1
            continuar = input("Desea continuar? (Si/No): ")

        print(f"El promedio de los números ingresados es: {suma/i}")
```

ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA



EJERCICIO: Escriba un algoritmo que permita leer números enteros positivos, contarlos y sumarlos. El algoritmo termina cuando se introduzca el cero o un numero negativo.



🏆 **EJERCICIO:** Escriba un algoritmo que permita leer números enteros positivos, contarlos y sumarlos. El algoritmo termina cuando se introduzca el cero o un numero negativo.

PSEUDOCÓDIGO

```
Clase SumaNumerosPositivos
publico vacio sumar_positivos()
    Entero: n=1, i=0, suma=0

    Mientras (n > 0) haga
        Escribir("Ingrese un número positivo o cero para terminar: ")
        Leer(n)
        Si (n > 0) Haga
            suma += n
            i++
        Fin_Si
    Fin_Mientras

    Escribir("El promedio de los números ingresados es: " + (suma/i))
Fin_Metodo
Fin_Clase
```

🏆 **EJERCICIO:** Escriba un algoritmo que permita leer números enteros positivos, contarlos y sumarlos. El algoritmo termina cuando se introduzca el cero o un numero negativo.

PYTHON

```
class SumaNumerosPositivos:
    def sumar_positivos():
        n=1
        i=0
        suma=0

        while(n > 0):
            n = int(input("Ingrese un número positivo o cero para terminar: "))
            if (n > 0):
                suma += n
                i += 1

        print(f"El promedio de los números ingresados es: {suma/i}")
```



EJERCICIO: El jefe del programa de Ingeniería de Sistemas le ha pedido que desarrolle una aplicación web para hacer una encuesta entre los estudiantes de Lógica. La encuesta debe pedir el nombre, sexo y nota final de que el estudiante obtuvo en la asignatura. Con base en estos datos, la aplicación debe mostrar el porcentaje de estudiantes que ganaron y perdieron el curso, respectivamente. Además, debe indicar cuál es la nota promedio entre los hombres que perdieron la asignatura y la nota promedio entre las mujeres que la ganaron.

Tenga presente que no se sabe cuántos estudiantes harán la encuesta por lo que tras terminar una encuesta se debe preguntar si se va a diligenciar otra encuesta o a ver los resultados de la misma.



ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA

🏆 **ANÁLISIS DEL PROBLEMA:** Identifique el cliente, usuario, los requisitos funcionales, las entradas, salidas y el proceso

CLIENTE Y USUARIO

-
-



ENTIDADES DEL MUNDO

-
-



REQUERIMIENTOS FUNCIONALES

-
-



ENTRADAS, SALIDAS Y EL PROCESO

-
-



ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA

🦋 **ANÁLISIS DEL PROBLEMA:** Identifique el cliente, usuario, los requisitos funcionales, las entradas, salidas y el proceso

CLIENTE	El jefe de programa de Ingeniería de Sistemas
USUARIOS	Estudiantes de Lógica
REQUERIMIENTOS FUNCIONALES	<p>El sistema debe permitir:</p> <ul style="list-style-type: none">▪ Registrar ingresar la información de una encuesta▪ Mostrar el porcentaje de estudiantes que ganaron y perdieron el curso▪ Mostrar la nota promedio de los hombres que perdieron la asignatura▪ Mostrar la nota promedio de las mujeres que ganaron la asignatura
ENTIDADES DEL MUNDO	<ul style="list-style-type: none">▪ Estudiante, del que se debe almacenar el nombre que es una cadena de caracteres, el sexo que es un número (1. Mujer o 2. Hombre), y la nota final de lógica, que es un número real entre 0 y 5.
ENTRADAS	<ul style="list-style-type: none">▪ El nombre, el sexo y nota de lógica del estudiante que está ingresando la encuesta
SALIDAS	<ul style="list-style-type: none">▪ El porcentaje de estudiantes que ganaron y perdieron la asignatura▪ La nota promedio de los hombres que perdieron la asignatura▪ La nota promedio de las mujeres que ganaron la asignatura
PROCESO	<ul style="list-style-type: none">▪ El porcentaje de los estudiantes que ganaron se calcula como: $\text{porGanaron} = (\text{numEstudiantesGanaron} / \text{numEstudiantes}) * 100$▪ El porcentaje de los que perdieron la asignatura se calcula como: $\text{porPerdieron} = (\text{numEstudiantesPerdieron} / \text{numEstudiantes}) * 100$▪ La nota promedio de los hombres que perdieron la asignatura se calcula sumando las notas menores a 3 para estudiantes hombres y contando dichas notas. El promedio es $\text{sumaNotasHombresPerdieron} / \text{numHombresPerdieron}$▪ La nota promedio de las mujeres que ganaron la asignatura se calcula sumando las notas mayores a 3 para estudiantes mujeres y contando dichas notas. El promedio es $\text{sumaNotasMujeresGanaron} / \text{numMuejeresGanron}$



EJERCICIO: Los miembros del consejo directivo le han pedido que desarrolle un algoritmo que permita a la Vicerrectora de Docencia hacer una encuesta entre los estudiantes de la UdeA. La encuesta debe determinar el porcentaje de estudiantes encuestados que pertenecen a las Facultades de Ingeniería, Artes, Economía y Ciencias Exactas. Además, debe mostrar qué porcentaje de estudiantes son menores de edad, el número de estudiantes mujeres con más de 20 años que son de la Facultad de Economía y, por último, debe mostrar el semestre que en promedio están cursando los encuestados que pertenecen a la Facultad de Ingeniería.



ESTRUCTURAS DE CONTROL REPETITIVAS



UNIVERSIDAD
DE ANTIOQUIA

🏆 **ANÁLISIS DEL PROBLEMA:** Identifique el cliente, usuario, los requisitos funcionales, las entradas, salidas y el proceso

CLIENTE Y USUARIO

-
-



ENTIDADES DEL MUNDO

-
-



REQUERIMIENTOS FUNCIONALES

-
-




ENTRADAS, SALIDAS Y EL PROCESO

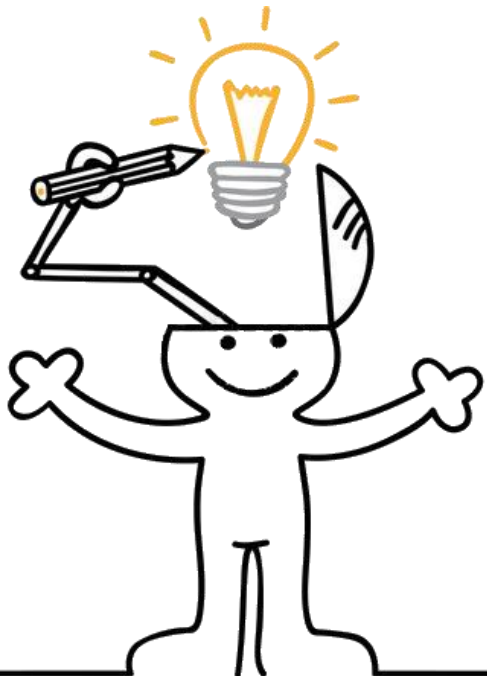
-
-



ESTRUCTURAS DE CONTROL REPETITIVAS

 **ANÁLISIS DEL PROBLEMA:** Identifique el cliente, usuario, los requisitos funcionales, las entradas, salidas y el proceso

CLIENTE	
USUARIOS	
REQUERIMIENTOS FUNCIONALES	
ENTIDADES DEL MUNDO	
ENTRADAS	
SALIDAS	
PROCESO	



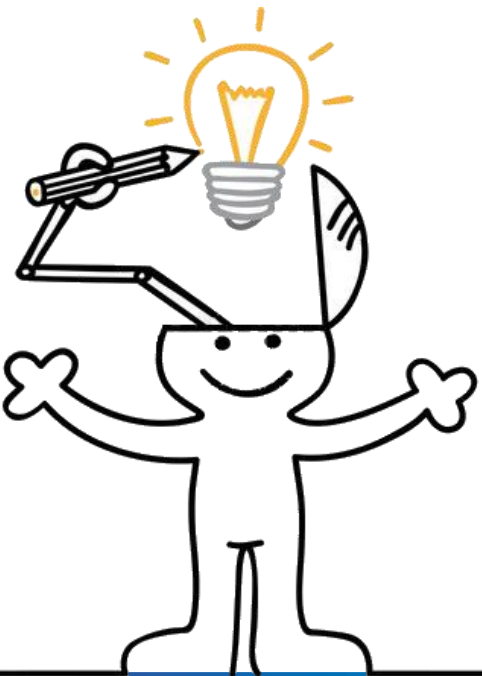
EJERCICIO: Haga un algoritmo que muestre los 100 primeros números primos



EJERCICIO: La conjetura de Collatz, conocida también como conjetura $3n+1$ o conjetura de Ulam, fue enunciada por el matemático Lothar Collatz en 1937. Escriba un algoritmo que muestre en pantalla la lista de números de la conjetura de Collatz, tal que dado un número positivo n :

- Si n es par, éste se divide por 2
- Si n es impar, éste se multiplica por 3 y se suma 1 al resultado.

Los pasos anteriores se repiten hasta que se obtenga como resultado el número 1.





**UNIVERSIDAD
DE ANTIOQUIA**

LÓGICA Y REPRESENTACIÓN I

PROGRAMA DE INGENIERÍA DE SISTEMAS

Material desarrollado por **Carlos Andrés Mera Banguero**

<https://github.com/carlosmera20/>