



UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

DEPARTAMENTO
DE INFORMÁTICA

UVA 4

Ciclos

IWI 131 – Programación

Prof. Álvaro Salinas

¿Qué errores tiene el siguiente código?

Un cliente necesita comprar precios para saber qué producto es el más conveniente. Contamos con un programa que permite al usuario ingresar la cantidad de productos a comparar y luego los precios de cada uno. Finalmente, se le indica cuál es el producto con menor precio.

```
n = int(input("Cantidad de productos a comparar: "))
i = 1
menor = 1000
while i < n:
    p = int(input("Precio producto "+str(i)+" : "))
    if p < menor:
        p = menor
print("El producto", i, "es el mas barato. Precio: $" + str(menor))
```

¿Qué errores tiene el siguiente código?

- Variable menor no es lo suficientemente grande.

```
n = int(input("Cantidad de productos a comparar: "))
i = 1
menor = float("inf")
while i < n:
    p = int(input("Precio producto "+str(i)+" : "))
    if p < menor:
        p = menor
print("El producto", i, "es el mas barato. Precio: $" + str(menor))
```

¿Qué errores tiene el siguiente código?

- Variable menor no es lo suficientemente grande.
- Para n veces, si comenzamos con 1, debemos usar <=.

```
n = int(input("Cantidad de productos a comparar: "))
i = 1
menor = float("inf")
while i <= n:
    p = int(input("Precio producto "+str(i)+" : "))
    if p < menor:
        p = menor
print("El producto", i, "es el mas barato. Precio: $" + str(menor))
```

¿Qué errores tiene el siguiente código?

- Variable menor no es lo suficientemente grande.
- Ciclo infinito.
- Para n veces, si comenzamos con 1, debemos usar <=.

```
n = int(input("Cantidad de productos a comparar: "))
i = 1
menor = float("inf")
while i <= n:
    p = int(input("Precio producto "+str(i)+": "))
    if p < menor:
        menor = p
    i += 1
print("El producto", i, "es el mas barato. Precio: $" + str(menor))
```

¿Qué errores tiene el siguiente código?

- Variable menor no es lo suficientemente grande.
- Para n veces, si comenzamos con 1, debemos usar <=.
- Ciclo infinito.
- No reemplazamos el menor actual.

```
n = int(input("Cantidad de productos a comparar: "))
i = 1
menor = float("inf")
while i <= n:
    p = int(input("Precio producto "+str(i)+" : "))
    if p < menor:
        menor = p
    i += 1
print("El producto", i, "es el mas barato. Precio: $" + str(menor))
```

¿Qué errores tiene el siguiente código?

- Variable menor no es lo suficientemente grande.
- Para n veces, si comenzamos con 1, debemos usar <=.
- Tampoco guardamos el número del producto.
- Ciclo infinito.
- No reemplazamos el menor actual.

```
n = int(input("Cantidad de productos a comparar: "))
i = 1
menor = float("inf")
while i <= n:
    p = int(input("Precio producto "+str(i)+" : "))
    if p < menor:
        menor = p
        menori = i
    i += 1
print("El producto", menori, "es el mas barato. Precio: $" + str(menor))
```

Completemos el código

Deseamos saber si un número es primo o compuesto. Para esto, debemos completar un código que solicita un número al usuario y le indica si es un número primo o no. Recordar que un número primo es aquel que solo es divisible por 1 y sí mismo.

Conversemos un poco sobre esto.

Completemos el código

Deseamos saber si un número es primo o compuesto. Para esto, debemos completar un código que solicita un número al usuario y le indica si es un número primo o no. Recordar que un número primo es aquel que es divisible únicamente por 1 y sí mismo.

En otras palabras, un número n es primo si no existen divisores de n entre 2 y $n-1$.

```
n = int(input("Numero: "))
es_primo = 
d = 
while d <= :
    if :
        es_primo = 
        
if :
    print("Es primo")
else:
    print("Es compuesto")
```

Completemos el código

Deseamos saber si un número es primo o compuesto. Para esto, debemos completar un código que solicita un número al usuario y le indica si es un número primo o no. Recordar que un número primo es aquel que es divisible únicamente por 1 y sí mismo.

En otras palabras, un número n es primo si no existen divisores de n entre 2 y $n-1$.

```
n = int(input("Numero: "))
es_primo = True
d = 
while d <= :
    if :
        es_primo = 
        
if :
    print("Es primo")
else:
    print("Es compuesto")
```

Completemos el código

Deseamos saber si un número es primo o compuesto. Para esto, debemos completar un código que solicita un número al usuario y le indica si es un número primo o no. Recordar que un número primo es aquel que es divisible únicamente por 1 y sí mismo.

En otras palabras, un número n es primo si no existen divisores de n entre 2 y $n-1$.

```
n = int(input("Numero: "))
es_primo = True
d = 2
while d <= :
    if :
        es_primo = 
    
if :
    print("Es primo")
else:
    print("Es compuesto")
```

Completemos el código

Deseamos saber si un número es primo o compuesto. Para esto, debemos completar un código que solicita un número al usuario y le indica si es un número primo o no. Recordar que un número primo es aquel que es divisible únicamente por 1 y sí mismo.

En otras palabras, un número n es primo si no existen divisores de n entre 2 y $n-1$.

```
n = int(input("Numero: "))
es_primo = True
d = 2
while d <= n**0.5:
    if :
        es_primo = 
        
if :
    print("Es primo")
else:
    print("Es compuesto")
```

Completemos el código

Deseamos saber si un número es primo o compuesto. Para esto, debemos completar un código que solicita un número al usuario y le indica si es un número primo o no. Recordar que un número primo es aquel que es divisible únicamente por 1 y sí mismo.



En otras palabras, un número n es primo si no existen divisores de n entre 2 y $n-1$.

```
n = int(input("Numero: "))
es_primo = True
d = 2
while d <= n**0.5:
    if n % d == 0:
        es_primo = 
        
    if :
        print("Es primo")
    else:
        print("Es compuesto")
```

Completemos el código

Deseamos saber si un número es primo o compuesto. Para esto, debemos completar un código que solicita un número al usuario y le indica si es un número primo o no. Recordar que un número primo es aquel que es divisible únicamente por 1 y sí mismo.

En otras palabras, un número n es primo si no existen divisores de n entre 2 y $n-1$.

```
n = int(input("Numero: "))
es_primo = True
d = 2
while d <= n**0.5:
    if n % d == 0:
        es_primo = False
        
if :
    print("Es primo")
else:
    print("Es compuesto")
```

Completemos el código

Deseamos saber si un número es primo o compuesto. Para esto, debemos completar un código que solicita un número al usuario y le indica si es un número primo o no. Recordar que un número primo es aquel que es divisible únicamente por 1 y sí mismo.

En otras palabras, un número n es primo si no existen divisores de n entre 2 y $n-1$.

```
n = int(input("Numero: "))
es_primo = True
d = 2
while d <= n**0.5:
    if n % d == 0:
        es_primo = False
        d += 1
if  :
    print("Es primo")
else:
    print("Es compuesto")
```

Completemos el código

Deseamos saber si un número es primo o compuesto. Para esto, debemos completar un código que solicita un número al usuario y le indica si es un número primo o no. Recordar que un número primo es aquel que es divisible únicamente por 1 y sí mismo.

En otras palabras, un número n es primo si no existen divisores de n entre 2 y $n-1$.

```
n = int(input("Numero: "))
es_primo = True
d = 2
while d <= n**0.5:
    if n % d == 0:
        es_primo = False
        d += 1
if es_primo:
    print("Es primo")
else:
    print("Es compuesto")
```


Ciclos anidados

Veamos el caso en donde el usuario ingresa un número n y se muestran n líneas, cada una con un número de asteriscos (*) incremental desde 1 hasta n . Es como si dibujáramos con asteriscos un triángulo rectángulo cuyos catetos miden n .

Numero: 4

```
*  
**  
***  
****
```

Numero: 6

```
*  
**  
***  
****  
*****  
*****
```

Numero: 8


```
*  
**  
***  
****  
*****  
*****  
*****  
*****  
*****
```

Ciclos anidados

Dividamos el problema:

- Primero hagamos un ciclo que imprima i asteriscos en una sola linea.

```
i = int(input("i: "))
j = 1
linea = ""
while j <= i:
    linea += "*"
    j += 1
print(linea)
```



i: 5

Ciclos anidados

Dividamos el problema:

- Primero hagamos un ciclo que imprima i asteriscos en una sola linea.
- Ahora hagamos un código que imprima n líneas con 1 asterisco.

```
i = int(input("i: "))
j = 1
linea = ""
while j <= i:
    linea += "*"
    j += 1
print(linea)
```

i: 5

```
n = int(input("n: "))
i = 1
while i <= n:
    print("*")
    i += 1
```


n: 5
*
*
*
*
*

Ciclos anidados

Ahora combinémoslos. Para esto modificaremos un poco ambos códigos:

- En el primer código, el largo de la línea (i) ya no será solicitado al usuario, sino que será la variable del ciclo del segundo código.

```
n = int(input("n: "))  
i = 1  
while i <= n:  
    print("*")  
    i += 1
```



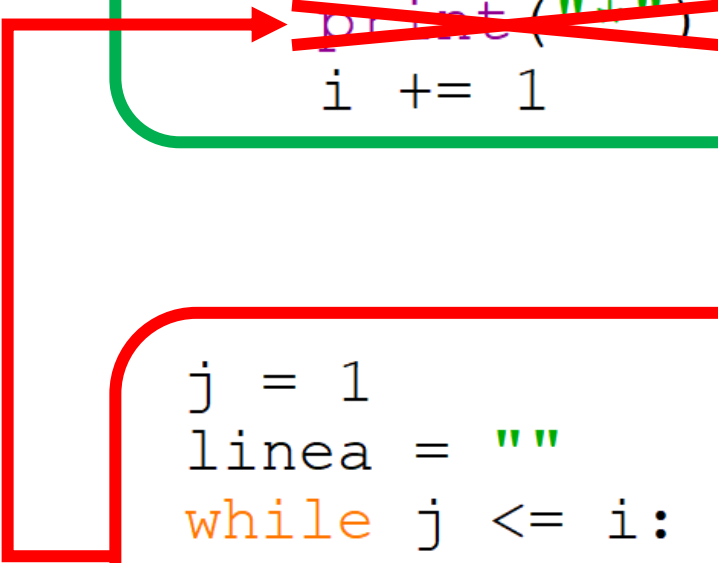
```
i = int(input("i: "))  
j = 1  
linea = ""  
while j <= i:  
    linea += "*"  
    j += 1  
print(linea)
```

Ciclos anidados

Ahora combinémoslos. Para esto modificaremos un poco ambos códigos:

- En el primer código, el largo de la línea (i) ya no será solicitado al usuario, sino que será la variable del ciclo del segundo código.
- En el segundo código, no imprimiremos una sola línea, sino la línea creada en el segundo código. Para esto, debemos reemplazar el print por toda la creación de la nueva línea.

```
n = int(input("n: "))  
i = 1  
while i <= n:  
    print("#")  
    i += 1
```



```
j = 1  
linea = ""  
while j <= i:  
    linea += "* "  
    j += 1  
print(linea)
```

Ciclos anidados

```
n = int(input("n: "))
i = 1
while i <= n:
    print("*")
    i += 1
```

```
i = int(input("i: "))
j = 1
linea = ""
while j <= i:
    linea += "*"
    j += 1
print(linea)
```

```
n = int(input("n: "))
i = 1
while i <= n:
    j = 1
    linea = ""
    while j <= i:
        linea += "*"
        j += 1
    print(linea)
    i += 1
```

Ciclos anidados

¿Fue todo esto necesario?

En términos prácticos, no.

No es realmente necesario utilizar un ciclo para crear cada línea, pues nosotros conocemos una operación que permite repetir un texto una determinada cantidad de veces. Bastaba con tomar el segundo código y multiplicar el asterisco a imprimir por el número i.

```
n = int(input("n: "))  
i = 1  
while i <= n:  
    print("*")  
    i += 1
```



```
n = int(input("n: "))  
i = 1  
while i <= n:  
    print("*" * i)  
    i += 1
```

Ciclos anidados

¿Fue todo esto necesario?

En términos de aprendizaje, sí.

No siempre vamos a contar una operación que simplifique uno de los ciclos involucrados, por lo que es importante comprender la estrategia de dividir el problema y luego mezclar las soluciones parciales. ¡Veamos otro caso!

Solicitemos al usuario un número n y mostrémosle todos los números primos entre 2 y n .

Numero: 10

2

3

5

7

Ciclos anidados

Nuevamente, dividamos el problema:

- Ya vimos en diapositivas anteriores un código que nos indica si un número es primo o no.

```
n = int(input("Numero: "))
es_primo = True
d = 2
while d <= n**0.5:
    if n % d == 0:
        es_primo = False
        d += 1
if es_primo:
    print("Es primo")
else:
    print("Es compuesto")
```

Ciclos anidados

Nuevamente, dividamos el problema:

- Ya vimos en diapositivas anteriores un código que nos indica si un número es primo o no.
- Por otro lado, nos interesa un código que imprima todos los números entre 2 y n. Claro que en la solución final deberemos imprimir solo aquellos que son primos, pero dicha condición será aportada por el primer código.


```
n = int(input("Numero: "))
es_primo = True
d = 2
while d <= n**0.5:
    if n % d == 0:
        es_primo = False
    d += 1
if es_primo:
    print("Es primo")
else:
    print("Es compuesto")
```

```
n = int(input("Numero: "))
i = 2
while i <= n:
    print(i)
    i += 1
```

Ciclos anidados

Ahora, adaptémoslos para poder combinarlos:

- Primero, ya no pediremos el número primo al usuario, ya que debemos revisar todos los números entre 2 y n. Utilicemos la variable i del otro ciclo.



```
i = int(input("Numero: "))  
es_primo = True  
d = 2  
while d <= i**0.5:  
    if i % d == 0:  
        es_primo = False  
        d += 1  
if es_primo:  
    print("Es primo")  
else:  
    print("Es compuesto")
```

```
n = int(input("Numero: "))  
i = 2  
while i <= n:  
    print(i)  
    i += 1
```

Ciclos anidados

Ahora, adaptémoslos para poder combinarlos:

- Primero, ya no pediremos el número primo al usuario, ya que debemos revisar todos los números entre 2 y n. Utilicemos la variable i del otro ciclo.
- Segundo, ya no nos interesa si el número es compuesto. Eliminemos el else.

```
es_primo = True
d = 2
while d <= i**0.5:
    if i % d == 0:
        es_primo = False
    d += 1
if es_primo:
    print("Es primo")
```

```
n = int(input("Numero: "))
i = 2
while i <= n:
    print(i)
    i += 1
```


Ciclos anidados

Ahora, adaptémoslos para poder combinarlos:

- Primero, ya no pediremos el número primo al usuario, ya que debemos revisar todos los números entre 2 y n. Utilicemos la variable i del otro ciclo.
- Segundo, ya no nos interesa si el número es compuesto. Eliminemos el else.
- Por ultimo, debemos dejar de imprimir si es primo y, en cambio, imprimir el número, lo cual reemplazaría el print del segundo ciclo.

```
es_primo = True
d = 2
while d <= i**0.5:
    if i % d == 0:
        es_primo = False
    d += 1
if es_primo:
    print(i)
```

```
n = int(input("Numero: "))
i = 2
while i <= n:
    print(i)
    i += 1
```



Ciclos anidados

```
n = int(input("Numero: "))
es_primo = True
d = 2
while d <= n**0.5:
    if n % d == 0:
        es_primo = False
    d += 1
if es_primo:
    print("Es primo")
else:
    print("Es compuesto")
```

```
n = int(input("Numero: "))
i = 2
while i <= n:
    print(i)
    i += 1
```

```
n = int(input("Numero: "))
i = 2
while i <= n:
    es_primo = True
    d = 2
    while d <= i**0.5:
        if i % d == 0:
            es_primo = False
            d += 1
    if es_primo:
        print(i)
    i += 1
```

Ejercicios

1. Se requiere desarrollar un programa para obtener el ganador de un concurso de talentos. El derecho a participación es libre, es decir, cualquier persona puede participar. Basta con pararse en una fila donde los participantes esperan su turno. En otras palabras, no se sabe de antemano cuántos participantes habrá. Cada participante recibirá una calificación entre 0 y 10. El ganador será quien obtenga la mayor puntuación.
Cree un programa que solicite puntajes para cada participante, preguntando luego de cada uno si es que hay más participantes en la fila (asuma que siempre habrá al menos 1 participante). Una vez que la fila haya terminado se debe indicar el ganador (en caso de empate, puede elegir a cualquiera).
Guíese por el ejemplo.

Ejercicios

1. Ejemplo:

Calificación participante 1: 6
¿Quedan participantes? (SI/NO): SI
Calificación participante 2: 2
¿Quedan participantes? (SI/NO): SI
Calificación participante 3: 4
¿Quedan participantes? (SI/NO): SI
Calificación participante 4: 7
¿Quedan participantes? (SI/NO): SI
Calificación participante 5: 5
¿Quedan participantes? (SI/NO): NO
El ganador es el participante 4 con 7 puntos

Ejercicios

2. Se desea crear un programa para automatizar la evaluación que los miembros del jurado realizan sobre la actuación de un participante. El número de jurados varia en cada edición del concurso, por lo que se requiere preguntar cuántos habrá al comenzar. El programa debe permitir a cada jurado ingresar el puntaje otorgado al participante evaluado (puntajes entre 0 y 10). Una vez que todos los jurados hayan realizado esta acción, el programa debe mostrar el promedio (aproximado a un decimal) de las calificaciones obtenidas por el participante. Guíese por el ejemplo.

Nota: Recuerde que este programa debe funcionar para cualquier número de jurados, pero para un único participante.

Ejercicios

2. Ejemplo:

Numero de jurados: 6

Jurado 1: 6

Jurado 2: 8

Jurado 3: 5

Jurado 4: 10

Jurado 5: 9

Jurado 6: 6

Promedio participante: 7.3

Ejercicios

3. ¡Mezclemos los enunciados anteriores!

Un concurso cuenta con una **fila de participantes** (no sabemos cuántos serán). Habrá un grupo de **jurados** que calificarán a cada participante. Ganará quien obtenga el **mayor promedio** de calificaciones. Aplican las reglas anteriores:

- Se debe preguntar la cantidad de jurados al comenzar.
 - Para cada participante, cada miembro del jurado debe ingresar una puntuación entre 0 y 10.
 - La calificación final de un participante, corresponde al promedio aproximado a un decimal de los puntajes otorgados por los miembros del jurado.
 - Luego de cada participante, se debe preguntar si quedan personas en la fila.
 - Asuma que siempre habrá al menos 1 participante.
 - Al terminar la fila, se indica al ganador (cualquiera en caso de empate).
- Guíese por el ejemplo.

Ejercicios

3. Ejemplo:

Número de jurados: 3

Participante 1:

Jurado 1: 8

Jurado 2: 5

Jurado 3: 7

Calificación final: 6.7

¿Quedan participantes? (SI/NO): SI

Participante 2:

Jurado 1: 5

Jurado 2: 7

Jurado 3: 5

Calificación final: 5.7

¿Quedan participantes? (SI/NO): SI

Participante 3:

Jurado 1: 8

Jurado 2: 7

Jurado 3: 4

Calificación final: 6.3

¿Quedan participantes? (SI/NO): NO

El ganador es el participante 1 con 6.7 puntos

Ruteo

Realice el ruteo del siguiente programa
(el número ingresado será 1234):

```
n = int(input("Numero: "))
m = 0
while n > 0:
    d = n % 10
    n = n // 10
    m = d + m * 10
print("Resultado:", m)
```

Pantalla

--

[illegible]

Desafío

Verifiquemos si un número es múltiplo de 9.

Parece una tarea fácil si utilizamos instrucciones como:

```
if n % 9 == 0:           if n / 9 == n // 9:
```

Pero en esta ocasión vamos a complicar un poco las cosas. Basaremos nuestro código en la siguiente propiedad: al sumar los dígitos de un múltiplo de nueve, el resultado es también un múltiplo de nueve. Ahora, ¿cómo sabemos si la suma de los dígitos es un múltiplo de nueve? Fácil, sumando sus dígitos. El procedimiento se repite sucesivamente hasta que se obtenga un 9.

Desafío

Ejemplos:

N: 873

Paso intermedio: 18

Final: 9

Es múltiplo de 9

Diagram illustrating the steps to check if 873 is a multiple of 9:

- Initial number: 873
- Sum of digits: $8 + 7 + 3 = 18$
- Sum of digits of the result: $1 + 8 = 9$
- Conclusion: Es múltiplo de 9 (It is a multiple of 9)

[illegible][illegible]