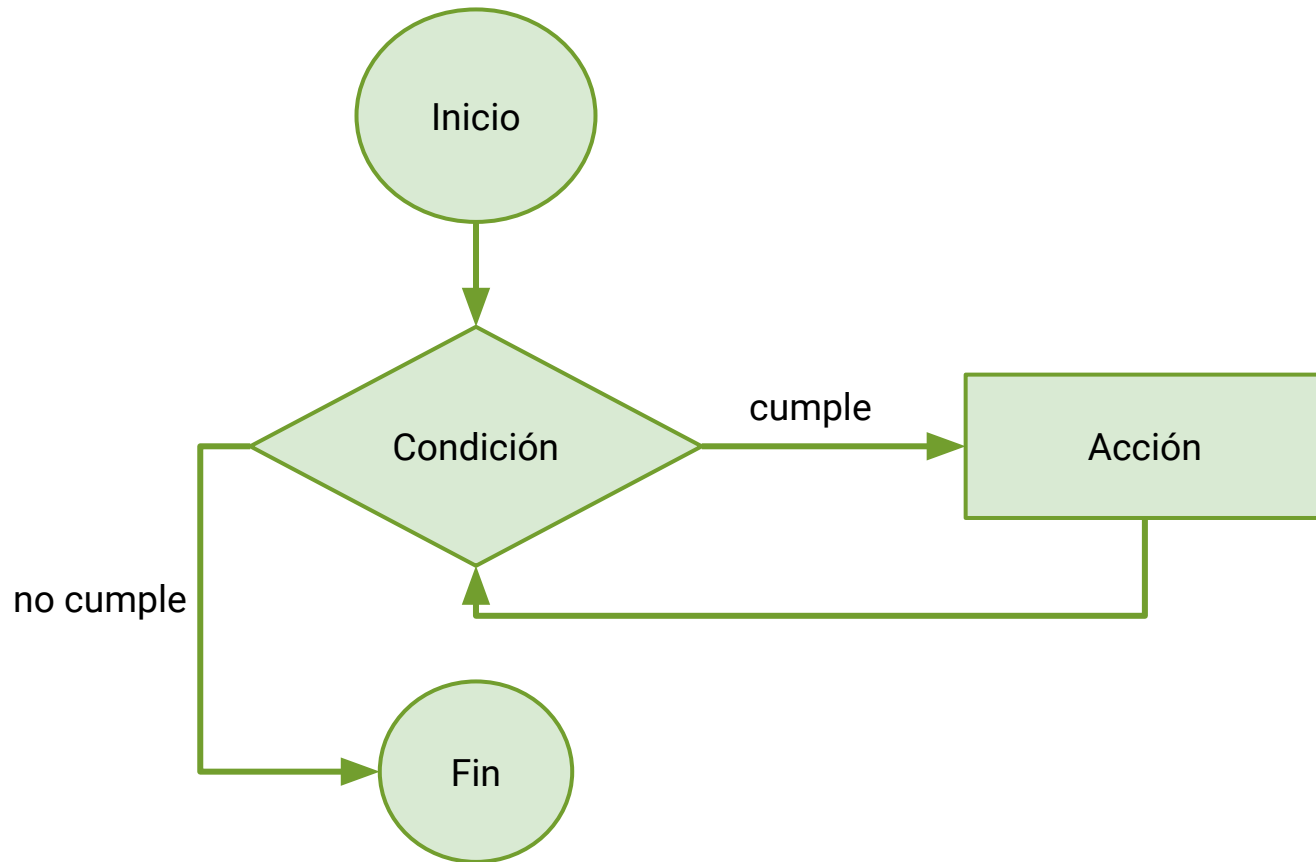


{desafío}
latam_

Ciclos y métodos _



| RECETA DE PANQUEQUES | ALGORITMO |
|---|---|
| <ul style="list-style-type: none"> - Agregar 1 taza de harina en un bowl - Agregar 1 taza de leche a la harina - Agregar 1 huevo a los ingredientes previos - Revolver y mezclar los 3 ingredientes - Precalentar el sartén - Agregar parte de la mezcla hasta cubrir el sartén y esparcir una capa delgada - Esperar 1 minuto - Dar vuelta la masa - Esperar otro minuto - Retirar el panqueque - Si queda masa, agregar parte de la mezcla hasta cubrir el sartén y esparcir una capa delgada - Esperar 1 minuto ... | <ol style="list-style-type: none"> 1. Agregar 1 taza de harina en un bowl. 2. Agregar 1 taza de leche a la harina. 3. Agregar 1 huevo a los ingredientes previos. 4. Revolver y mezclar los 3 ingredientes. 5. Precalentar el sartén. 6. Agregar parte de la mezcla hasta cubrir el sartén y esparcir una capa delgada. 7. Esperar 1 minuto. 8. Dar vuelta la masa. 9. Esperar otro minuto. 10. Retirar el panqueque. 11. Repetir pasos del 6 al 10 hasta terminar la mezcla. |



Validar entrada de un
usuario utilizando **while**

```
1 # Primero se pide la entrada al usuario
2 # IMPORTANTE: La variable que se utilice para evaluar la condición DEBE
3 # estar definida antes de que se ejecute el ciclo
4 number = int(input("Ingresa un número entre 1 y 10"))
5
6 # Después se inicia el ciclo
7 # La condición de salida es que el número esté entre 1 y 10
8 while number < 1 or number > 10:
9     # Este código se ejecutará "mientras" se ingrese un número
10    # menor a 1 o mayor a 10
11    print("El número ingresado no está entre 1 y 10")
12    number = int(input("Ingresa un número entre 1 y 10"))
13
14 print("El número ingresado fue ", number)
```

Ejercicio: Menú de opciones

```
1 # Se inicializa la variable para evaluar la condición de while
2 opcion = ""
3
4 # La condición de salida será que opción sea "salir"
5 while opcion != "salir":
6
7     # Se agregan los print para dibujar las opciones del menú
8     print("Ingrese una opción")
9     print("1: Sumar 2 + 2")
10    print("2: Multiplicar 2 * 2")
11    print("salir: Salida")
12
13    # Se solicita la opción al usuario
14    opcion = input()
15
16    # Si escoge 1, se hará la suma 2 + 2
17    # opcion vale 1, por lo que volverá a entrar en el ciclo
18    if opcion == "1":
19        print(2 + 2)
20
21    # Si escoge 2, se hará la multiplicación 2 * 2
22    # opcion vale 2, por lo que volverá a entrar en el ciclo
23    elif opcion == "2":
24        print(2 * 2)
25
26    # Si escoge "salir", se muestra el mensaje indicando que se detendrá el ciclo
27    # opción ha tomado el valor de la condición de salida (no se volverá a mostrar el menú)
28    elif opcion == "salir":
29        print("Saliendo")]
```

Operadores de asignación

| OPERADOR | NOMBRE | EJEMPLO | RESULTADO |
|----------|-----------------------------|---------|--|
| = | Asignación | a = 2 | a toma el valor 2 |
| += | Incremento y asignación | a += 2 | a es incremento en 2 y asignado el valor resultante |
| -= | Decremento y asignación | a -= 2 | a es reducido en 2 y asignado el valor resultante |
| *= | Multiplicación y asignación | a *= 3 | a es multiplicado por 3 y asignado el valor resultante |
| /= | División y asignación | a /= 3 | a es dividido por 3 y asignado el valor resultante |

Sumatoria 1 a N

```
1  # Importar módulo sys
2  import sys
3
4  # Almacenar en una variable el valor ingresado como argumento
5  # Debe guardarse como int
6  limite = int(sys.argv[1])
7
8  # Crear e inicializar en 0 la variable "i" que funcionará como contador
9  i = 0
10
11 # Crear e inicializar en 0 la variable "suma" que irá acumulando la sumatoria
12 suma = 0
13
14 # Crear el ciclo while que iterará mientras el contador sea menor al límite
15 while i < limite:
16     # Aumentar en 1 el valor del contador en cada iteración
17     i += 1
18
19     # Acumular en la variable suma el valor de i en cada iteración
20     suma += i
21
22 # Imprimir el valor final de la sumatoria al salir del ciclo
23 print(suma)
```

Lista HTML

Paso 1: Código inicial

```
1  # Importar el módulo sys
2  import sys
3
4  # Almacenar en una variable el valor ingresado como argumento
5  # Debe guardarse como int
6  items = int(sys.argv[1])
7
8  # iniciar el conador en 0
9  i = 0
10
11 # iniciar el acumulador como string vacío
12 html = ""
13
14 # Iterar mientras el contador sea menor a items
15 while i < items:
16     # aumentar el contador en 1 en cada iteración
17     i += 1
18     # Concatenar los tag de apertura y cierre para cada elemento de la lista
19     html += "<li></li>"
20
21 print(html)
```

Paso 2: Agregar contenido a cada elemento de la lista

```
html += "<li>Elemento</li>"
```

Paso 3: Interpolar el valor del iterador en el contenido

```
html += "<li>Elemento {}</li>".format(i)
```

Paso 4: Agregar salto de línea

```
html += "<li>Elemento {}\\n</li>".format(i)
```


Paso 5: Agregar tags de apertura y cierre de la lista

```
11 # Agregar tag de apertura de la lista
12 html = "<ul>\n"
13
14 # Iterar mientras el contador sea menor a items
15 while i < items:
16     # aumentar el contador en 1 en cada iteración
17     i += 1
18     # Agregar salto de línea
19     html += "<li>Elemento {}\n</li>".format(i)
20
21 # Agregar tag de cierre de la lista
22 html += "</ul>"
23 print(html)
```

Paso 6: Agregar tabulación en los elementos de la lista

```
html += "\t<li>Elemento {}\n</li>".format(i)
```

Patrón “nick msn”

Posición

Símbolo

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|----|----|----|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| ★ | ⋮ | ⌒ | ⋮ | ★ | ⋮ | ⌒ | ⋮ | ★ | ⋮ | ⌒ | ⋮ | ★ |



| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|----|----|----|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| ★ | ⋮ | ⌒ | ⋮ | ★ | ⋮ | ⌒ | ⋮ | ★ | ⋮ | ⌒ | ⋮ | ★ |



| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|----|----|----|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| ★ | ⋮ | ⌒ | ⋮ | ★ | ⋮ | ⌒ | ⋮ | ★ | ⋮ | ⌒ | ⋮ | ★ |

Casos

- Impar: $\begin{smallmatrix} \blacksquare & \blacksquare \\ \blacksquare & \blacksquare \end{smallmatrix}$

- Divisible por 4: ★

- El resto: $\overline{\hspace{1cm}}$

Código final

```
1  #Importamos sys
2  import sys
3
4  # Almacenamos el argumento como int
5  limit = int(sys.argv[1])
6
7  #Patrón de referencia
8  #pattern = "*... /-\\ ...*... /-\\ ...*... /-\\ "
9
10 # almacenamos cada elemento del patrón en una variable
11 a = "*"
12 b = "... "
13 c = "/-\\ "
14
15 # inicializamos el acumulador
16 contain = ""
17
18 # Recorremos el rango hasta el límite más 1
19 for i in range(limit + 1):
20     # Primer caso: Concatenar puntos si el elemento es impar
21     if i % 2 != 0:
22         contain += b
23
24     # Segundo caso: Concatenar estrella si el elemento es divisible por 4
25     elif i % 4 == 0:
26         contain += a
27
28     # El resto serán las líneas
29     else:
30         contain += c
31
32 # Se imprime el acumulador
33 print(contain)
```

Desafío

Lluvia de agua nieve

```

1  # Importamos sys
2  import sys
3
4  # Importamos random
5  import random
6
7  # Almacenamos el ancho como int
8  width = int(sys.argv[1])
9
10 # Validamos que no sea menor a 10
11 if width < 10:
12     width = 10
13
14 # Creamos el contenedor
15 output = ""
16
17 # Recorremos de 1 a 9 en el loop externo
18 for i in range(1, 10):
19     # Creamos el primer número al azar, utilizando el iterador y el ancho del usuario
20     rand_number = random.randint(i, width)
21
22     # Usamos el número creado para agregar espacios en blanco antes de la nieve
23     output += " " * rand_number + "*" + "\n"
24
25     # Recorremos el segundo loop con rango de 1 a i
26     for j in range(1, i):
27         # Creamos el segundo número al azar, utilizando el segundo iterador y el ancho
28         rand_number_2 = random.randint(j, width)
29
30         # Concatenamos la lluvia siguiendo la misma lógica del loop externo
31         output += " " * rand_number_2 + "/" + "\n"
32
33 # Mostramos el lienzo creado
34 print(output)
35

```


Desafío Nube y Lluvia

(basado en el desafío agua y nieve)

Paso 1: Eliminar indentación, segundo for y cambiar range

```
18 for i in range(1, 10):
19     # Creamos el primer número al azar, utilizando el iterador y el ancho del usuario
20     rand_number = random.randint(i, width)
21
22     # Usamos el número creado para agregar espacios en blanco antes de la nieve
23     output += " " * rand_number + "*" + "\n"
24
25 # Recorremos el segundo loop con rango de 1 a 10
26 for j in range(1, 10):
27     # Creamos el segundo número al azar, utilizando el segundo iterador y el ancho
28     rand_number_2 = random.randint(j, width)
29
30     # Concatenamos la lluvia siguiendo la misma lógica del loop externo
31     output += "." * rand_number_2 + "/" + "\n"
32
```

Paso 2: Transformar la nieve en nube

```
18 for i in range(1, 10):
19     # Creamos el primer número al azar, utilizando el iterador y el ancho del usuario
20     rand_number = random.randint(i, width)
21
22     # Cambiar nieve por nube
23     output += "@" * rand_number + "\n"
24
```

Paso 3: Iniciar contenedor con @

```
14 # Inicializamos contenedor con el ancho de la nube
15 output = "@" * width + "\n|"
16
```

Paso 4: Cambiar lógica para crear primer random

```
18 for i in range(1, 10):
19     # Creamos inicio del random como el 80% de width
20     start_random = int(0.8 * width)
21
22     # Creamos el primer número al azar, utilizando el start_random y el ancho del usuario
23     rand_number = random.randint(start_random, width)
24
25     # Cambiar nieve por nube
26     output += "@" * rand_number + "\n"
27
```

Llamar vs Definir

- Cuando se **crea** una función, ésta se está definiendo.
- Cuando se **usa** una función, ésta se está llamando.

Parámetros

- Variables locales.
- Se utilizan en la lógica de la función.
- Se definen junto con la función.

Tipos de variables según su alcance

- Globales.
- Locales.
- De instancia.
- De clase.

{desafío}
latam_

*Academia de
talentos digitales*

www.desafiolatam.com