**Day 12 While Loop**

**What is a While Loop?**

A **while loop** is a fundamental control flow structure in Python that allows you to repeatedly execute a block of code as long as a specified condition remains True.

Here’s how it works:

1. **Check Condition:** The loop first checks the condition.
2. **Execute Code:** If the condition is True, the code block inside the loop is executed.
3. **Repeat:** After the block is executed, it returns to step 1 and checks the condition again.
4. **Exit:** This process continues until the condition becomes False, at which point the loop terminates and the program continues with the code that follows.

It is crucial that the code inside the loop modifies a variable that is part of the condition. If the condition *always* remains True, you will create an **infinite loop**.

**Example:**

Python

a = 1

while a < 5: # 1. Condition is checked

print(a) # 2. Code is executed

a += 1 # 3. The condition variable is modified

**Loops Day 12 Notes**

**For Loop Patterns**

**Example 1: Simple chr() Loop**

**Definition:** A basic for loop that iterates 4 times (for i in 1, 2, 3, 4). In each iteration, it prints the character for Unicode 65, which is 'A'. The end=" " keeps all the output on a single line.

**Example Code:**

Python

for i in range(1,5):

print(chr(65),end=" ")

**Output:**

Plaintext

A A A A

**Example 2: Nested Loop Printing a Variable**

**Definition:** This nested loop uses a = 5. The outer loop iterates 5 times (for i in 65, 66, 67, 68, 69). For each outer iteration, the inner loop iterates 4 times (for j in 1, 2, 3, 4) and prints the value of a (which is 5).

**Example Code:**

Python

a = 5

for i in range(65,70):

for j in range(1,a,1):

print(a,end =" ")

print()

**Output:**

Plaintext

5 5 5 5

5 5 5 5

5 5 5 5

5 5 5 5

5 5 5 5

**Example 3: Letter Square Pattern**

**Definition:** A nested loop that prints a 4x4 square. The outer loop (i) controls the rows. The inner loop (j) controls the columns, printing the character for 64 + j. This results in chr(65) ('A'), chr(66) ('B'), etc., being printed in each row.

**Example Code:**

Python

for i in range(1,5):

for j in range(1,5):

print(chr(64+j),end=" ")

print()

**Output:**

Plaintext

A B C D

A B C D

A B C D

A B C D

**Example 4: Right-Angle Letter Triangle**

**Definition:** This nested loop prints a right-angle triangle. The inner loop (j) depends on the outer loop (i), running from 1 up to i + 1. This prints an increasing number of letters in each row ('A', then 'A B', etc.).

**Example Code:**

Python

# iii)

for i in range(1,5):

for j in range(1,i+1,1):

print(chr(64+j ),end =" ")

print()

**Output:**

Plaintext

A

A B

A B C

A B C D

**Example 5: Mixed Pattern ('\*' and Letters)**

**Definition:** This pattern uses two inner loops.

* The first inner loop (j) prints \* symbols, increasing with each row (from 0 to 3).
* The second inner loop (k) prints the *same* letter (determined by chr(65 + i - 1)) repeatedly in that row.

**Example Code:**

Python

# iv)

for i in range (1,5):

for j in range(i):

print("\*",end= " ")

for k in range(i):

print(chr(65 + i - 1), end=" ")

print()

**Output:**

Plaintext

\* A

\* \* B B

\* \* \* C C C

\* \* \* \* D D D D

**While Loop Examples**

**Example 1: Basic while Loop (New Lines)**

**Definition:** This is a basic while loop that initializes a to 1. It prints the value of a and increments it (a += 1) as long as a is less than 5. Each print() statement ends with a new line by default.

**Example Code:**

Python

a = 1

while a < 5:

print(a)

a += 1

**Output:**

Plaintext

1

2

3

4

**Example 2: Basic while Loop (Single Line)**

**Definition:** This is the same logic as Example 1, but the end=" " parameter in the print() function causes all the output to appear on a single line, separated by spaces.

**Example Code:**

Python

a = 1

while a < 5:

print(a, end=" ")

a += 1

**Output:**

Plaintext

1 2 3 4

**Example 3: Nested while and for Loops**

**Definition:** This code nests a for loop inside a while loop.

* The outer while loop runs 4 times (for a = 1, 2, 3, 4).
* In each pass of the while loop, the inner for loop runs completely, printing "1 2 3 4".
* The print() after the for loop moves to a new line.

**Example Code:**

Python

a=1

while a<5 :

for i in range(1,5,1):

print(i,end=" ")

print()

a+=1

**Output:**

Plaintext

1 2 3 4

1 2 3 4

1 2 3 4

1 2 3 4

**Example 4: Sorting Even/Odd with while**

**Definition:** This code uses a while loop to iterate from 1 to 10. It uses an if-else statement with the modulo operator (%) to check if a is even or odd and then appends the number to the correct list.

**Example Code:**

Python

a = 1

even = []

odd = []

while a <= 10:

if a %2 == 0:

even.append(a)

else:

odd.append(a)

a += 1

print(even)

print(odd)

**Output:**

Plaintext

[2, 4, 6, 8, 10]

[1, 3, 5, 7, 9]

**Home work**

**Example 1: Mixed Pattern (Inverted Letters)**

**Definition:** This pattern uses range(4) (for i = 0, 1, 2, 3).

* The first inner loop (j) prints an increasing number of \* symbols (0, then 1, then 2, then 3).
* The second inner loop (k) prints letters in reverse, with the sequence getting shorter in each row (D C B A, then C B A, etc.).

**Example Code:**

Python

for i in range(4):

for j in range(i):

print('\*',end=" ")

for k in range(68-i,64,-1):

print(chr(k),end=" ")

print()

**Output:**

Plaintext

D C B A

\* C B A

\* \* B A

\* \* \* A

**Example 2: Mixed Pattern (Inverted '\*' and Letters)**

**Definition:** This pattern also uses range(4).

* The first inner loop (j) prints a decreasing number of \* symbols (3, then 2, then 1, then 0).
* The second inner loop (k) prints letters in reverse, with the sequence getting *longer* in each row (A, then B A, etc.).

**Example Code:**

Python

for i in range(4):

for j in range(3-i):

print('\*',end=" ")

for k in range(68-(3-i),64,-1):

print(chr(k),end=" ")

print()

**Output:**

Plaintext

\* \* \* A

\* \* B A

\* C B A

D C B A

**Example 3: Inverted Star Pyramid**

**Definition:** This code prints an inverted pyramid of stars, centered with spaces.

* The first inner loop (j) prints an increasing number of spaces for alignment.
* The second inner loop (k) prints a decreasing, odd number of stars (5, then 3, then 1).

**Example Code:**

Python

for i in range(3):

for j in range(65 + i, 65 + i + i + 1):

print(" ", end=' ')

for k in range(69 - 2 \* i, 64, -1):

print('\*', end=' ')

print()

**Output:**

Plaintext

 \* \* \* \* \*

   \* \* \*

     \*