**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Topic 51 - While Loops**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**What**

* A **while loop** continues to execute a block of code **as long as a specified condition is true**.
* It’s particularly useful when you don’t know in advance how many times you want to execute the loop.

**Why**

* **Dynamic Repetition**: While loops let you keep checking conditions until they’re met, making them great for **user input** or **unknown length processing**.
* **User Control**: They allow users to keep interacting with a program until they decide to quit.

**How**

1. **Basic Structure of a While Loop**  
   Define a condition that keeps the loop going:

python

Copy code

user\_input = ""

while user\_input != "q":

user\_input = input("Enter a city, or 'q' to quit: ")

print(f"You entered: {user\_input}")

1. **Using While Loops to Process User Input Until a Condition Is Met**  
   In this example, we use while to continuously accept city names from the user until they type "q" to quit:

python

Copy code

cleanest\_cities = ["Cheyenne", "Santa Fe", "Tucson", "Great Falls", "Honolulu"]

user\_input = ""

while user\_input != "q":

user\_input = input("Enter a city, or 'q' to quit: ")

if user\_input != "q":

for city in cleanest\_cities:

if user\_input == city:

print("It's one of the cleanest cities")

break # Exit for loop if a match is found

1. **Using Indentation to Define Nested Blocks in Loops and Conditionals**  
   Python relies on indentation to group statements together. Inside a while loop, any nested for loops, if statements, or other commands **must be indented** to clarify they are part of the loop:

python

Copy code

cleanest\_cities = ["Cheyenne", "Santa Fe", "Tucson", "Great Falls", "Honolulu"]

user\_input = ""

while user\_input != "q": # Starts while loop

user\_input = input("Enter a city, or 'q' to quit: ")

if user\_input != "q": # Checks if user wants to continue

for city in cleanest\_cities: # Starts for loop to check city

if user\_input == city:

print("It's one of the cleanest cities")

break # Ends for loop if match is found

**Things to Remember**

* **Initialize Variables**: Always assign an initial value to variables before starting a while loop. For instance, user\_input = "" sets the initial state for the loop condition.
* **Indentation**: Use indentation to clearly define blocks of code controlled by while, for, and if statements.
* **Break Statement**: Use break inside a loop to **exit the loop immediately** when a condition is met.