SDEV 1001

Programming Fundamentals

More Loops and Exceptions - 2

A LEADING POLYTECHNIC COMMITTED TO YOUR SUCCESS

Expectations - What I expect from you

- No Late Assignments

- No Cheating

- Be a good classmate

- Don’t waste your time

- Show up to class

Agenda

On the right is what we will cover today.

- While Loops in Python
- Basic While Loop Example
- Collecting User Input with a While Loop
- Using a Flag to Control the Loop
- While Loop and using `break`

While Loops in Python

- While loops repeat a block of code as long as a condition is true.
- Useful when you don’t know in advance how many times you need to repeat.

Analogy

Think of doing home work. You’re not going to stop until you finish all the questions, right? That’s like a while loop!

This is different from a for loop, which repeats a specific number of times. You’re not going to say "I will do homework for 1 hour three times" but rather "I will do homework until I finish all the questions".

Basic While Loop Example

A while loop continues as long as a condition is true.

- in the example below the loop will continue until the `counter` is greater than `3`.

Example:

```
counter = 1
while counter <= 3:
    print(f"Attempt {counter}")
    counter += 1
```

Heres what the output.:

```
Attempt 1
Attempt 2
Attempt 3
```

Collecting User Input with a While Loop

Something we're going to do in this course is collect user input until they decide to stop.

- In the example below, we will keep asking the user for numbers stored in the `value` variable until they type "done".

```
print("Enter numbers to add. Type 'done' to finish.")
total = 0
while True:
    value = input("Enter a number: ")
    if value == "done":
        break
    total += int(value)
print(f"Total sum: {total}")
```

Here's a sample output:

```
Enter numbers to add. Type 'done' to finish.
Enter a number: 5
Enter a number: 10
Enter a number: done
Total sum: 15
```

Using a Flag to Control the Loop

Here’s a different way to control a while loop using a boolean flag.

- You initialze a flag variable (here `keep_going`) to `True` and then set it to `False` when you want to stop the loop.

```
keep_going = True
while keep_going:
    answer = input("Keep going? (y/n): ")
    if answer == "n":
        keep_going = False
print("Loop ended.")
```

Here's what the output looks like:

```
Keep going? (y/n): y
Keep going? (y/n): y
Keep going? (y/n): n
Loop ended.
```

While Loop and using `break`

Here's a practical example where we calculate the average of test scores entered by the user.

- Note that instead of a flag like in the previous example, we use a `break` statement to exit the loop when the user types "q". This is also a common pattern in while loops.

```
count = 0
total = 0
while True:
    score = input("Enter a test score (or 'q' to quit): ")
    if score == "q":
        break
    total += int(score)
    count += 1
if count > 0:
    print("Average score:", total / count)
else:
    print("No scores entered.")
```

Summary

- Use while loops when you need to repeat until a condition is met.
- Use `break` to exit early, and flags to control the loop.
- Great for user input, repeated calculations, and more.

Example

Let’s go run a few examples together