Chapter 3

How to code control statements/Loops

The syntax of the while statement

```
while boolean_expression:
    statements...
```

A while loop that continues as long as the user enters ‘y’ or ‘Y’

```
choice = "y"
while choice.lower() == "y":
    print("Hello!")
    choice = input("Say hello again? (y/n): ")
print("Bye!")  # runs when loop ends
```

The console after the loop runs

```
Hello!
Say hello again? (y/n): y
Hello!
Say hello again? (y/n): n
Bye!
```

A while loop that prints the numbers 0 through 4 to the console

```
counter = 0
while counter < 5:
    print(counter, end=" ")
    counter += 1
print("\nThe loop has ended.")
```

The console after the loop runs

```
0 1 2 3 4
The loop has ended.
```

Code that causes an infinite loop

```
while True:
    # any statements in this loop run forever
    # unless a break statement is executed as shown later
```

How to end an infinite loop

- Press Ctrl+C (Windows) or Command+C (macOS).

The syntax of a for loop with the range() function

```
for int_var in range_function:
    statements...
```

The range() function

```
range(stop)
range(start, stop[, step])
```

Examples of the range() function

```
range(5)                    # 0, 1, 2, 3, 4
range(1, 6)                 # 1, 2, 3, 4, 5
range(2, 10, 2)             # 2, 4, 6, 8
range(5, 0, -1)             # 5, 4, 3, 2, 1
```

A for loop that prints the numbers 0 through 4

```
for i in range(5):
    print(i, end=" ")
print("\nThe loop has ended.")
```

The console after the loop runs

```
0 1 2 3 4
The loop has ended.
```

A for loop that sums the numbers 1 through 4

```
sum_of_numbers = 0
for i in range(1,5):
    sum_of_numbers += i
print(sum_of_numbers)
```

The console after the loop runs

```
10
```

A break statement that exits an infinite while loop

```
print("Enter 'exit' when you're done.\n")
while True:
    data = input("Enter an integer to square: ")
    if data == "exit":
        break
    i = int(data)
    print(i, "squared is", i * i, "\n")
print("Okay, bye!")
```

The console

```
Enter 'exit' when you're done.

Enter an integer to square: 10
10 squared is 100

Enter an integer to square: 23
23 squared is 529

Enter an integer to square: exit
Okay, bye!
```

A continue statement that jumps to the beginning of a while loop

```
more = "y"
while more.lower() == "y":
    miles_driven = float(input("Enter miles driven:\t\t"))
    gallons_used = float(input("Enter gallons of gas used:\t"))

    # validate input
    if miles_driven <= 0 or gallons_used <= 0:
        print("Both entries must be greater than zero. ",
              "Try again.\n")
        continue

    mpg = round(miles_driven / gallons_used, 2)
    print("Miles Per Gallon:", mpg, "\n")

    more = input("Continue? (y/n): ")
    print()

print("Okay, bye!")
```

Loops that calculates the future value of a one-time investment

A for loop

```
investment = 10000
for i in range(20):
    yearly_interest = investment * .05
    investment = investment + yearly_interest
investment = round(investment, 2)
```

A while loop

```
year = 0
investment = 10000
while year < 20:
    yearly_interest = investment * .05
    investment = investment + yearly_interest
    year += 1
investment = round(investment, 2)
```

A for loop that calculates the future value of a monthly investment

```
monthly_investment = 100
monthly_interest_rate = .08 / 12
months = 120
future_value = 0
for month in range(months):
    future_value += monthly_investment
    monthly_interest_amount = future_value *
                              monthly_interest_rate
    future_value += monthly_interest_amount
future_value = round(future_value, 2)
```

Nested loops that get the total of 3 valid test scores

```
total_score = 0
for i in range(3):
    while True:
        score = int(input("Enter test score: "))
        if score >= 0 and score <= 100:
            total_score += score
            break
        else:
            print("Test score must be from 0 - 100.")
print("Total score:", total_score)
```

The console

```
Enter test score: 110
Test score must be from 0 - 100.
Enter test score: -10
Test score must be from 0 - 100.
Enter test score: 100
Enter test score: 90
Enter test score: 0
Total score: 190
```

The operator used with assignment expressions

Operator	Name
`:=`	Walrus

A while statement that uses an infinite loop to process user data

```
print("Enter -1 to quit.")
print("=================")
while True:
    score = input("Enter a score: ") # assign
    if score == "-1":                # check
        break
    print(f"You entered {score}.")
print("Bye!")
```

How to rewrite the code using an assignment expression

```
print("Enter -1 to quit.")
print("==================")
# assign and check
while (score := input("Enter a score: ")) != "-1":
    print(f"You entered {score}.")
print("Bye!")
```

The console for both loops

```
Enter -1 to quit.
==================
Enter a score: 90
You entered 90.
Enter a score: 99
You entered 99.
Enter a score: -1
Bye!
```

Pseudocode for a Test Scores program

```
Display user message
WHILE TRUE
    get score
    IF score is from 0 to 100
            add score to score total
            add 1 to number of scores
    ELSE IF score is 999
            end loop
    ELSE
            print error message
Calculate average score
Display results
```

The user interface for the Test Scores program

```
The Test Scores program

Enter 999 to end input
======================
Enter test score: 85
Enter test score: 95
Enter test score: 155
Test score must be from 0 through 100. Try again.
Enter test score: 75
Enter test score: 999
======================
Total Score: 255
Average Score: 85

Bye!
```

The code for the Test Scores program (part 1)

```
#!/usr/bin/env python3

# display a welcome message
print("The Test Scores program")
print()
print("Enter 999 to end input")
print("======================")

# initialize variables
counter = 0
score_total = 0
test_score = 0
```

The code for the Test Scores program (part 2)

```
while True:
    test_score = int(input("Enter test score: "))
    if test_score >= 0 and test_score <= 100:
        score_total += test_score
        counter += 1
    elif test_score == 999:
        break
    else:
        print("Test score must be from 0 through 100. ",
              "Score discarded. Try again.")

# calculate average score
average_score = round(score_total / counter)

# format and display the result
print("======================")
print(f"Total Score: {score_total}"
      f"\nAverage Score: {average_score}")
print()
print("Bye")
```

Pseudocode for a Future Value program

```
Display user message
WHILE user wants to continue
    get monthly investment, yearly interest rate, and years
    convert yearly interest rate to monthly interest rate
    convert years to months
    set the future value to zero
    FOR each month
        add monthly investment amount to future value
        calculate interest for month
        add interest to future value
    display future value
    ask if user wants to continue
Display end message
```

The user interface for the Future Value Calculator

```
Welcome to the Future Value Calculator

Enter monthly investment:        100
Enter yearly interest rate:      12
Enter number of years:           10
Future value:                    23233.91

Continue (y/n)?:
```

The code for the Future Value Calculator (part 1)

```
#!/usr/bin/env python3

# display a welcome message
print("Welcome to the Future Value Calculator")
print()

choice = "y"
while choice.lower() == "y":

    # get input from the user
    monthly_investment = float(input(
        "Enter monthly investment:\t"))
    yearly_interest_rate = float(input(
        "Enter yearly interest rate:\t"))
    years = int(input(
        "Enter number of years:\t\t"))

    # convert yearly values to monthly values
    monthly_interest_rate = yearly_interest_rate / 12 / 100
    months = years * 12
```

The code for the Future Value Calculator (part 2)

```
    # calculate the future value
        future_value = 0
        for i in range(months):
            future_value += monthly_investment
            monthly_interest_amount = future_value *
                                      monthly_interest_rate
            future_value += monthly_interest_amount

        # display the result
        print(f"Future value:\t\t\t{round(future_value, 2)}")
        print()

        # see if the user wants to continue
        choice = input("Continue (y/n)?: ")
        print()

    print("Bye!")
```