Week 6 materials

topics covered

- increment & decrement values
- loops
- the while loop

some recap

- to increment a value, we can use `+=`:

```
a = 2
a += 1

# a will be 3
# a += 1 is the same as a = a + 1 (we are updating value of variable a, on the right hand side we have 2 + 1)
```

- to decrement a value, we can use `-=`:

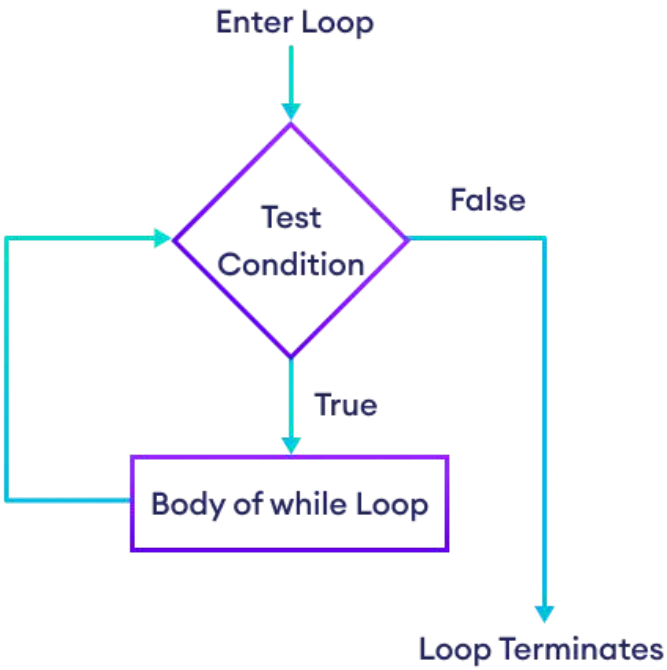
```
a = 3
a -= 1

# a will be 2
# a -= 1 is the same as a = a - 1 (we are updating value of variable a, on the right hand side we have 3 - 1)
```

- we can increment or decrement by any value: `a += 10` `b -= 5`

- loops are used to repeat a block of code: e.g. if we want to show a message 100 times, then we can use a loop
- Python `while` loop is used to run a block code until a certain condition is met

- syntax:

```
while condition:
    # body of while loop
```



- example 1 involving numbers:

```
# program to display numbers from 1 to 5

# initialize the variable
i = 1
n = 5

# while loop from i = 1 to 5
while i <= n:
    print(i)
    i = i + 1
```

output:

1
2
3
4
5

- example 2 involving lists:

```
a = ['foo', 'bar', 'baz']
while a:
    print(a.pop(-1))
```

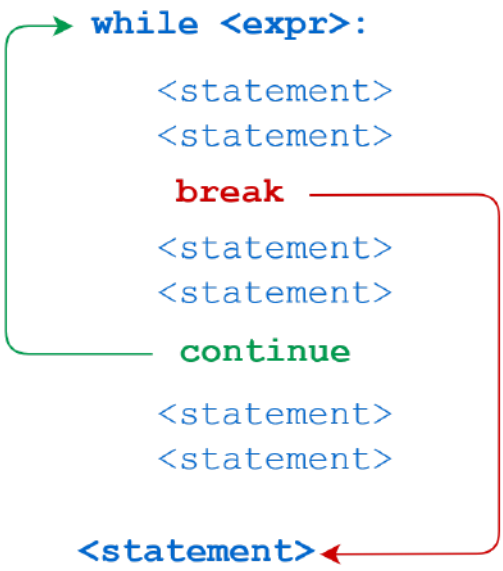
output:

baz

bar

foo

- Python `break` statement immediately terminates a loop entirely
- Python `continue` statement immediately terminates the current loop iteration. Execution jumps to the top of the loop



- a `while` loop that theoretically never ends is called an infinite loop

  `while True:` will be executing statements infinitely

here's some some examples with while loop:

```
# Prints all letters except whitespace ' ', 'e', and 's'
i = 0
a = 'hello ladies and gentlemen'

while i < len(a):
    if a[i] == ' ' or a[i] == 'e' or a[i] == 's' :
        i += 1
        continue

    print('Current Letter :', a[i])
    i += 1
```

```
# Python while loop with user input
a = int(input('Enter a number (-1 to quit): '))
```

```
while a != -1:
    a = int(input('Enter a number (-1 to quit): '))
```

```
# Initialize a counter
count = 0

# Loop infinitely
while True:
    # Increment the counter
    count += 1
    print(f"Count is {count}")

    # Check if the counter has reached a certain value
    if count == 10:
        # If so, exit the loop
        break

# This will be executed after the loop exits
print("The loop has ended.")
```

exercises

Save each python program which you wrote any code for in this format: p5_exerciseNum.py e.g. p5_20.py

1. Print ‘Hello World’ 5 times using While loop.

2. Imagine you have a list `my_list = [’apple’, ‘banana’, ‘lemon’]`. Iterate over the items in the list `my_list` using while loop and print each element.

3. Iterate from `i = 1` to `i = 10`, but break the loop if `i` equals 4. Use while loop, break statement, and print out `i` every time you iterate.

4. Iterate from `i = 1` to `i = 10`, but if `i` equals 4, skip the next statements in the loop and continue with the next iteration. Use while loop, break statement, and print out `i` every time you iterate.

5. Write a program that uses a while loop to print numbers from 1 to 10.

6. Write a program that prompts the user to enter a password. Keep asking for the password until the user enters the correct password, which is "password123". Print a success message when the correct password is entered.

7. Write a program that calculates the sum of all even numbers between 0 and 50 using a while loop.

8. Write a program that prompts the user to enter a positive integer and prints its multiplication table from 1 to 10 using a while loop.

9. Write a program that asks the user for a series of numbers until the user enters a negative number. Then, calculate and print the sum of all the entered numbers, do not use lists in this program.

10. Write a program that allows the user to add items to a shopping list. The program should continue to ask for items until the user enters "done" to stop adding items. Finally, print the complete shopping list

11. Write a program that allows the user to add tasks to a to-do list. The program should prompt the user to add a new task or enter "done" to finish. Once done, print the final to-do list. Use list concept here, you need to append each task.

12. Write a program that asks the user to enter their grades for a course. The program should calculate the average grade and provide a corresponding letter grade based on the following scale: 90-100: A, 80-89: B, 70-79: C, below 70: F. To stop entering grades user should enter `-1`

13. Write a program to read a string and display it, replacing any occurrence of A in the string by o:

```
Enter a string with an A in it: ABBA
Replacing A with o gives: oBBo
```

14. Write a program to read a string and count the number of digits (0,1,2,..9) in it

```
Enter a string: 12abc34def56
There are 6 digits in 12abc34def56
```

Hint: if s[i] >='0' **and** s[i] <= '9' then s[i] contains a digit.

15. Write a program to read a string and count the number of uppercase letters (A..Z) and lowercase letters (a..z) in the string

```
Enter a string : ABC defg
3 uppercase letters and 4 lowercase letters in ABC defg
```

Hint: if s[i] >='A' **and** s[i] <= 'Z' then s[i] contains an uppercase letter.

16. Write a program that allows the user to create a checklist for party supplies. The program should prompt the user to enter an item for the checklist or enter "done" to finish adding items. Once the user is done, the program should display the final party supplies checklist.

17. Write a program that allows the user to track their daily expenses. The program should prompt the user to enter the amount spent for each expense and categorize them into `"food"`, `"transportation"`, or `"other"`. At the end, calculate and display the total amount spent in each category. To stop entering expenses user should enter `-1`

18. Write a program that asks the user to enter a series of numbers. The program should store each number in a list and stop accepting numbers when the user enters 0. At the end, calculate and display the sum of all the entered numbers, you can use `sum()`

19. Write a program that allows the user to enter their daily steps and displays the total steps and average steps. Store user's input in a list. Use `sum()` and `len()`

20. Write a program that allows the user to track daily temperatures. The program prompts the user to enter the temperature for each day and stores them in a list. It then calculates the highest temperature, lowest temperature, and average temperature. Display the results at the end. Stop receiving input from the user when they enter -999.

21. Write a program that allows a student to track their daily study time. The program prompts the user to enter their study time for each day and stores them in a list. Calculate the total study time and average study time per day. Display the results at the end. Stop receiving input from the user when they enter -1.

22. Write a program that allows the user to create a music playlist. The program prompts the user to enter song titles and stores them in a list. The user can continue adding songs until they choose to stop, and then the program displays the final playlist.

23. Using a `while` loop and an `if` statement append all the elements in a list to a new list unless the element is an empty string `""`.

24. Write a python program to read a number and print a right triangle using `*` E.g. :

    Input : 5

    ---------Output---------

    *

    * *

    * * *

    * * * *

    * * * * *

24. Write a program using a loop to

    read number of hours worked per week (maximum is 100)

    read rate per hour (max is 50)

compute gross pay (pay before tax)

compute tax at 10% of pay if pay > 100

tax is 0 if pay <100

display result as:

```
Gross Pay: 200
Rate: 10
Hours worked: 20
Tax: 20
Net Pay: 180

Press Y to continue for another employee
```

25. Write a program that allows the user to calculate their monthly expenses based on their budget and expenses for each category. The program should prompt the user to enter the budget amount and then ask for expenses in different categories. It should calculate the total expenses, determine if the user exceeded the budget, and display the results. The program should offer the option to calculate expenses for another month.

26. Write a program that allows the user to track their daily tasks. The program prompts the user to enter task names and their estimated durations. It then calculates the total time estimated for all tasks and determines if it exceeds a predefined limit. The program displays the result and offers the option to track tasks for another day.

27. Write a program that simulates a library catalog. Allow the user to add books by entering the title and author. Provide options to search for a book by title, display all books, or exit the catalog. Create different lists for different data.

28. Write a program that simulates a phonebook. Allow the user to add contacts by entering a name and phone number. Provide options to search for a contact by name, display all contacts, or exit the phonebook.

FINAL PROJECT:

Extend exercise number 19 from week 2:

Create a program that simulates a hospital registration system. Prompt the user to enter the following information:

- Name
- Surname
- Age
- Height (in cm)
- Health complaints

Store each piece of information in a separate variable with an appropriate name.

Finally, print the information in the following format:

Name: [Name]
Surname: [Surname]
Age: [Age]
Height: [Height]
Health complaints: [Health Complaints]

Replace [Name], [Surname], [Age], [Height], and [Health Complaints] with the actual values entered by the user.

Here is what you need to add:

1. you need to ask how many patients a user wants to enter data for
2. create separate lists for each piece of information e.g. name, surname etc
3. then ask all information for those patients
4. store data in appropriate lists with appropriate index
5. after user completes entering all information about the patients ask if they want to see information about particular patient, they should use surname to search for that patient, find the index of that patient and print all info about them
6. user might want to see information about several patients, so consider using while loop and each time ask user if they want to see the info about the patient
7. when user enters ‘no’ stop executing the program

Use user friendly outputs, and all of the concepts that you studied previous weeks. You might also check for erroneous input (optional).