Hour 1

Review of Functions:

- **Function signature** is the name of function and the parameters (local variables)
- **Doc string** of a function is the comment which tells us what the function is, what it does, parameters it takes and what it returns. It lies just below the signature inside the function. Triple quotations are used to denote start and end.
- **Body** of the function contains the code that the function runs.
- **Return** statement of the function is the final step after which the function returns back to from where it was called. Any code after the return statement does not run.

Scope:

- **Global variables** are outside any functions in the actual root of the program and can be accessed by any function in the file.
- Use less global variables as they can make the code crowded, messy and might lead to wrong usage of variables.

Boolean:

- Can be either True or False.
- Variable data type.
- Can understand 3 operators:
    - **not** is negation and flips True to False and vice versa.
    - **and** means both inputs have to be true to output true.
    - **or** means either one of the two or both have to be true to output true.
- Relational operators compare any 2 operators or groups of operators:
    - < and >
    - <= and >=
    - == equal
    - != not equal
    - Example:
        - dogWeight = 25 assigns integer 25 to the variable
        - dogWeight == 25 compares variable to integer and will output a True
        - dogWeight < 10 will output a False
        - dogWeight != 25 will also output a False as it is equal to 25

Answers for Class Exercise table 1:

1. True
2. False
3. Nothing
4. False
5. b = False assigns
6. False
7. Assigns

Conditionals:

- Written as if <condition>: <things to be done if true>.
- Use diamond shape for condition in flowchart
- In python tab spacing should be maintained after the if statement.
- If - else clause is used for 2 way condition and else is optional and just if can be used for 1 way condition.
- Can be sequentially stacked and nested.
- If- elif-else — elif is just else if in an abbreviated form and multiple elifs can be used in a chain.

Hour 2

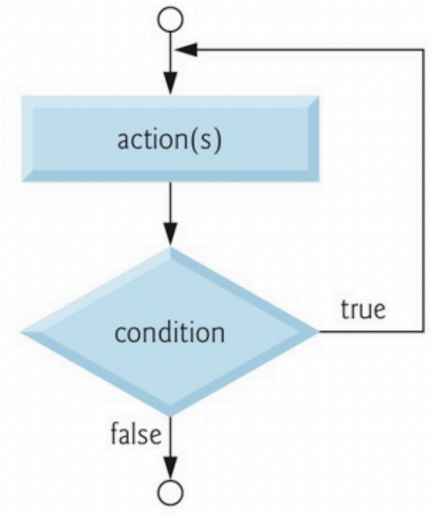
Breakout Exercise: A program to learn basic usage of if statement and else statement.

```
'''
CS5001 Class 3
Breakout Ex 1
Neel Patel
Fall 22

Program for preferance of the user about ice cream and broccoli.
'''


def main():

    # Ice cream
    ice_cream = input("Do you like ice cream? (yes / no)")
    if ice_cream.lower() == "yes":
        print("Yay! So do I!")

    # Broccoli
    broccoli = input("Do you like broccoli? (yes / no)")
    if broccoli.lower() == "yes":
        print("I like it with cheese")
    else:
        print("But it's healthy")

    # Both
    if broccoli == "yes" and ice_cream == "yes":
        print("Time for a healthy broccoli ice cream sundae. Yum!")

if __name__ == "__main__":
    main()
```

*Note: **.upper() and .lower()** changes the case of the string. We used that so that we don't have to write a longer code by doing something like if x == "q" or x == "Q".*

While Loop:

- When a code runs repeatedly, it is called **iteration**.
- It repeats action until specified.
- **Break** statement immediately exits the loop like an ejection seat on the bat mobile. It will stop exactly from the point the break statement was called.
- **Continue** statement says we are going to do another statement just not do anything for now, thus immediately.



```
product = 3
while True: # careful here...don't want an infinite loop!
    product = product * 3
    answer = input("Keep going ?")
    if answer != 'Y' and answer != 'y':
        break
print(product)
```

This is an example of an **infinite loop** and break statement used to cut off the infinite loop.

Hour 3 and 3.5

```
Program to flip coin and track the results.
'''
import random

def flip_coin():
    '''
    Returns heads or tails based on a random number.
    '''

    number = random.randint(0, 200)

    if number % 2 == 0:
        return "HEADS"
    return "TAILS"

def main():

    index = 0
    heads = 0
    tails = 0

    while index < 10: # index is our loop and sentinal value.
        if flip_coin() == "HEADS":
            heads += 1
        else:
            tails += 1 # shorthand notation for incrementation
        index = index + 1

    print(f"In {index} flips we got {heads} Heads and {tails} Tails.")

if __name__ == "__main__":
    main()
```

Above is a program for flipping a coin written with the use of conditional statements. Here it also counts the number of heads and tails we get and prints them out.

For the same program a **flag** can be used where you set the value of flag as true before entering the loop and set the value to false when you want to exit the loop. The while loop would look something like this:

```
flag = True # Assign a flag variable

while index < 10 and flag == True: # index is our loop and sentinal value.

    # Flag Condition:
    if index > 10:
        flag = False
        |
    if flip_coin() == "HEADS":
        heads += 1
    else:
        tails += 1 # shorthand notation for incrementation
    index = index + 1

print(f"In {index} flips we got {heads} Heads and {tails} Tails.")
```