XCCELERATE

LOOPS, CONDITIONALS AND ALGORITHMIC THINKING

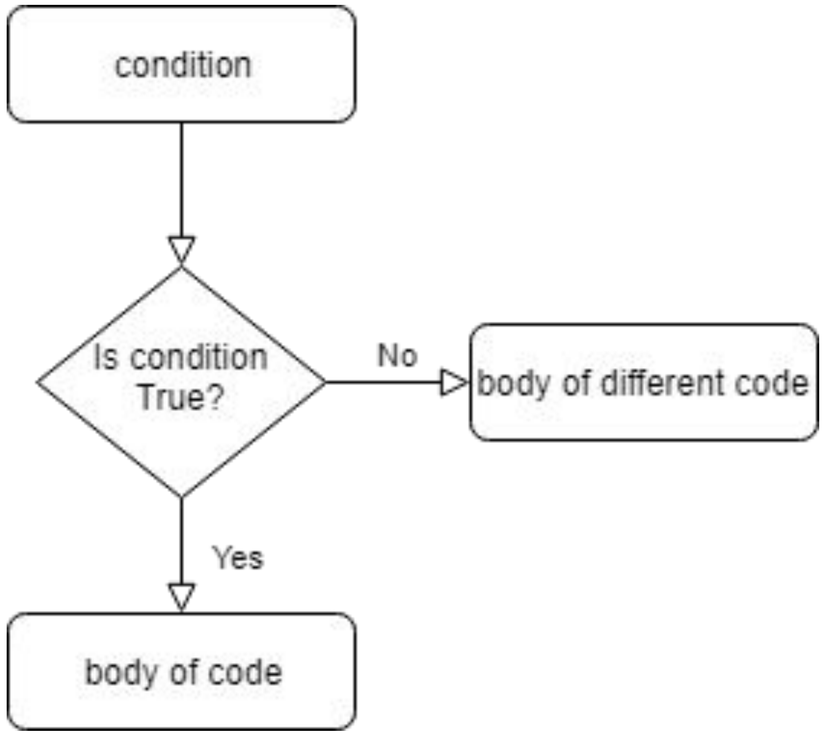
CONDITIONALS

CONDITIONALS

- Used for decision making and control flow of programs
- programs are sequentially run (top to bottom)
- **control flow**: directs the order of execution of statements in a program

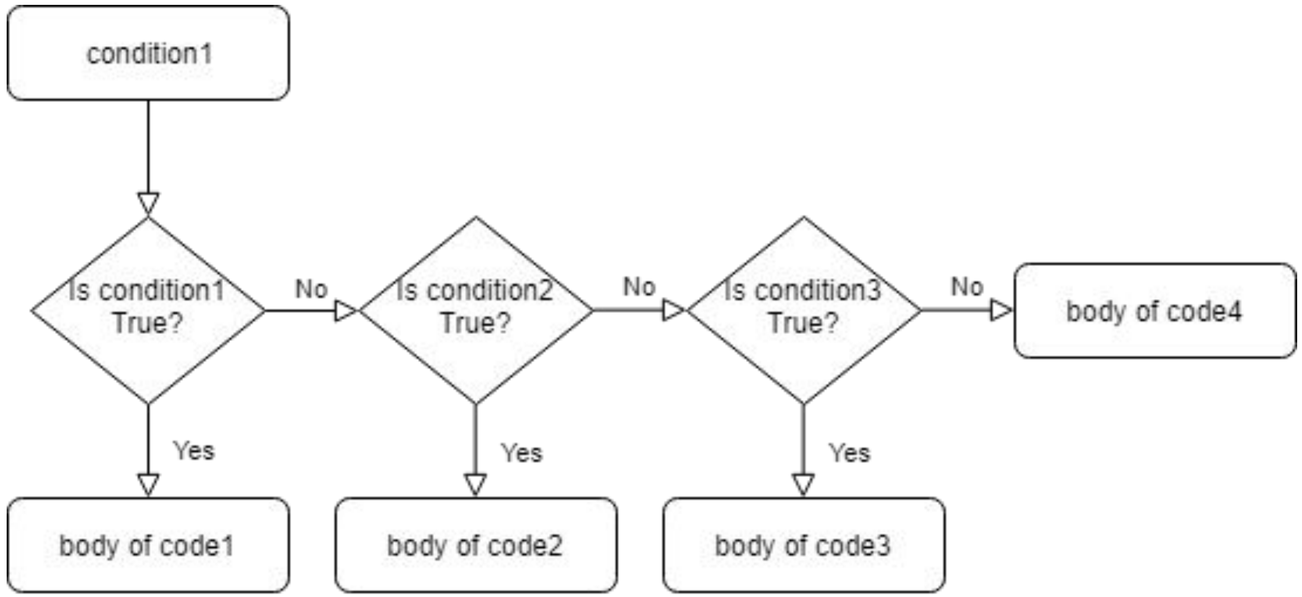
IF… ELSE STATEMENTS

- *if* and *else* must be aligned together
- body of code for each *if* and *else* must also be aligned together




```
if condition == True:
|   |
|   body of code # a Tab indentation or 4 whitespaces for execute code
|   |
else:
|   |
|   body of different code # a Tab indentation or 4 whitespaces for execute code
```

ELIF STATEMENTS




```
if condition1:
    body of code1
elif condition2:
    body of code2
elif condition3:
    body of code3
else:
    body of code4
```

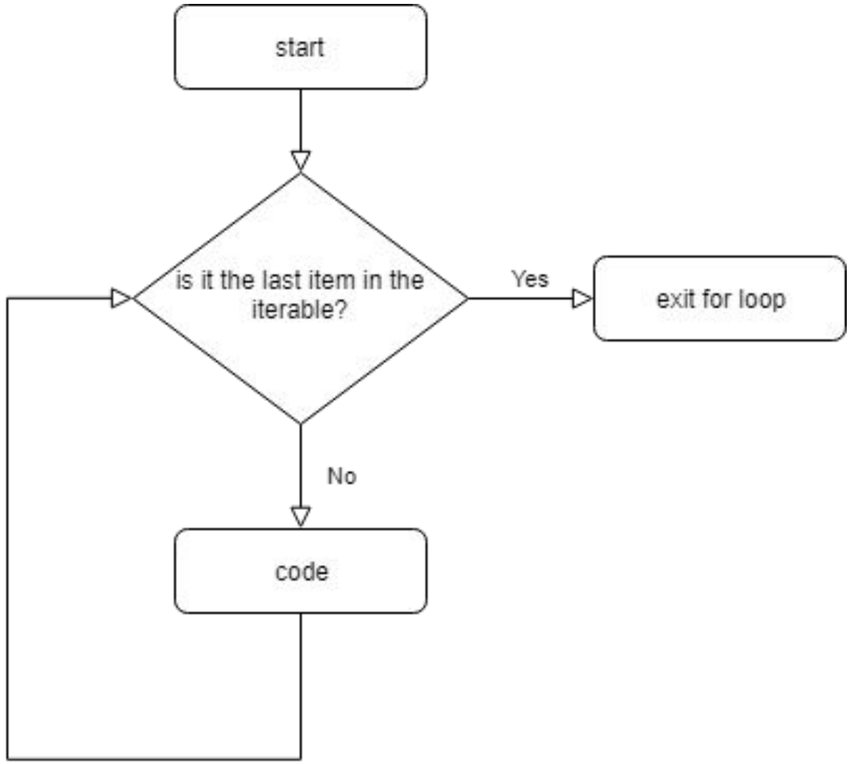
LOOPS

LOOPS

- used for repetition; run the same code over and over again
- **iterator:** variable that goes through each element in the iterable
- **iterable:** an object that is can be iterated over
  - list
  - tuple
  - dictionary
  - string
  - set

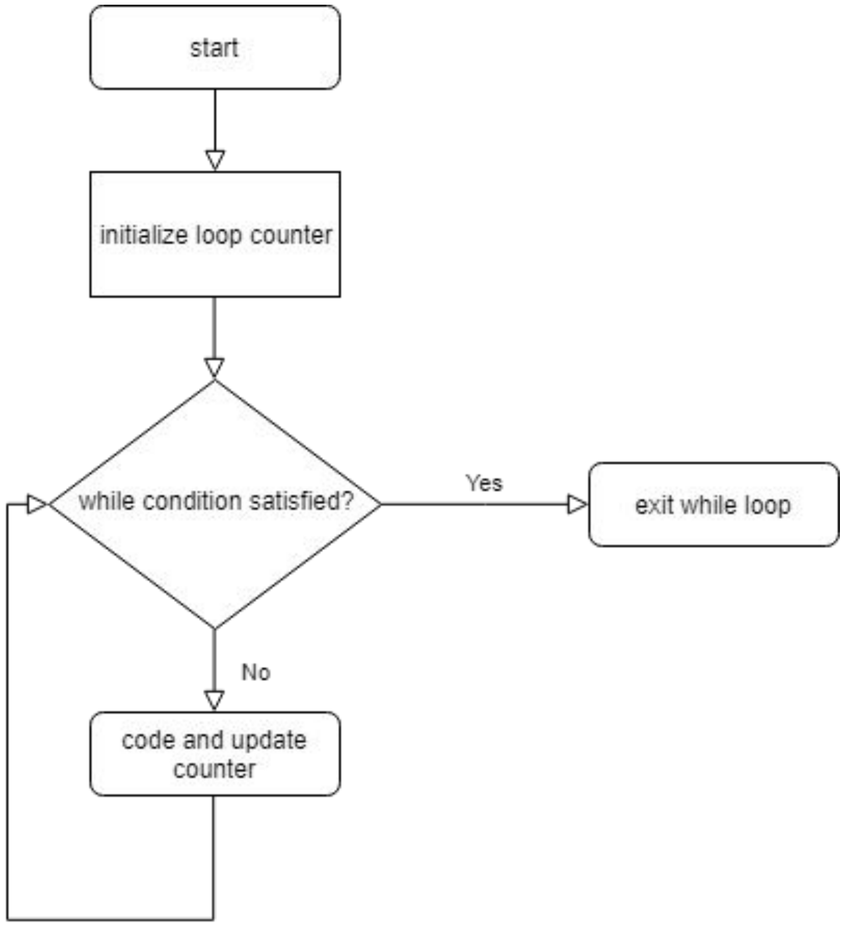
```
for iterator in iterable:
    code # code is Tab or 4 whitespaces indented
```

FOR LOOPS




```
for iterator in iterable:
    code # code is Tab or 4 whitespaces indented
```

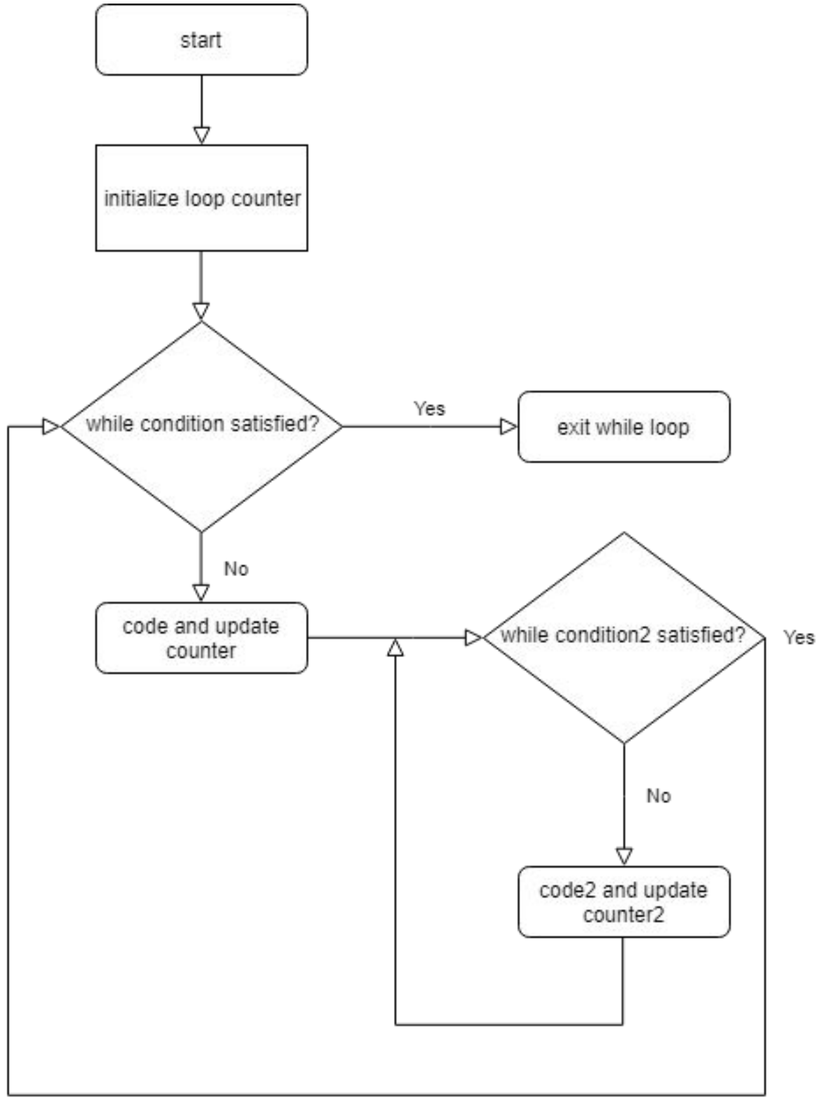
WHILE LOOPS




```
while condition:
    code # code is Tab or 4 whitespaces indented
```

NESTED LOOPS

- a loop inside a loop



ALGORITHMIC THINKING

ALGORITHMS

- a process to be followed to solve a specific problem or a specific set of problems
- involves conditions to be adhered to
- all problems to be solved in programming should be using algorithms

EXAMPLE ALGORITHM: BUBBLE SORT

Problem

We are **given an array of numbers** and we are supposed to **sort the numbers in ascending order**.

1. Starting with the first element, compare the current element with the next element of the array.
2. If the current element is greater than the next element of the array, swap them.
3. If the current element is less than the next element, move to the next element.
4. Repeat steps 2 to 4 till there are no more swaps.

EXAMPLE ALGORITHM: BUBBLE SORT

Given the list **[1, 5, 3, 2, 4]**

1. Starting with the 1st element 1, we compare 1 and 5. Because, 1 < 5, there are no swaps.

   [1, 5, 3, 2, 4]

2. We now move to the next element 5. We compare 5 and 3. Because 5 > 3, we swap 3 and 5.

   [1, 3, 5, 2, 4]

3. 5 is still our current element. We compare 5 and 2. Because 5 > 2, we swap 2 and 5.

   [1, 3, 2, 5, 4]

4. 5 is still our current element. We compare 5 and 4. Because 5 > 4, we swap 4 and 5.

   [1, 3, 2, 4, 5]

5. There are no more numbers to compare to 5. Therefore, we move back to 1 since it's the 1st element. We compare 1 and 3. Because, 1 < 3, there are no swaps.

   [1, 3, 2, 4, 5]

EXAMPLE ALGORITHM: BUBBLE SORT

[1, 3, 2, 4, 5]

6. We now move to the 3 as our current element. We compare 3 and 2. Because 3> 2, we swap 3 and 2.

   [1, 2, 3, 4, 5]

7. We now compare 3 and 4. Because 3 < 4, there are no swaps.

   [1, 2, 3, 4, 5]

8. We now move to the next element 4. We compare 4 and 5. Because 4 < 5, there are no swaps.

   [1, 2, 3, 4, 5]

9. There are no more numbers to compare to 5. Therefore, we move back to 1 since it's the 1st element.
10. We now compare 1 and 2. Because 1 < 2, there are no swaps.
11. We now move to the next element 2. We compare 2 and 3. Because 2 < 3, there are no swaps.
12. We now move to the next element 3. We compare 3 and 4. Because 3 < 4, there are no swaps.
13. We now move to the next element 4. We compare 4 and 5. Because 4 < 5, there are no swaps.
14. Because we went through the whole array and had no swaps, this means the array has been sorted. Algorithm stops.

    [1, 2, 3, 4, 5]

XCCELERATE

LOOPS, CONDITIONALS AND ALGORITHMIC THINKING