Computing GC content with for loops

BILD 62

Quick check in...

https://www.menti.com/blhpa9c4jpki

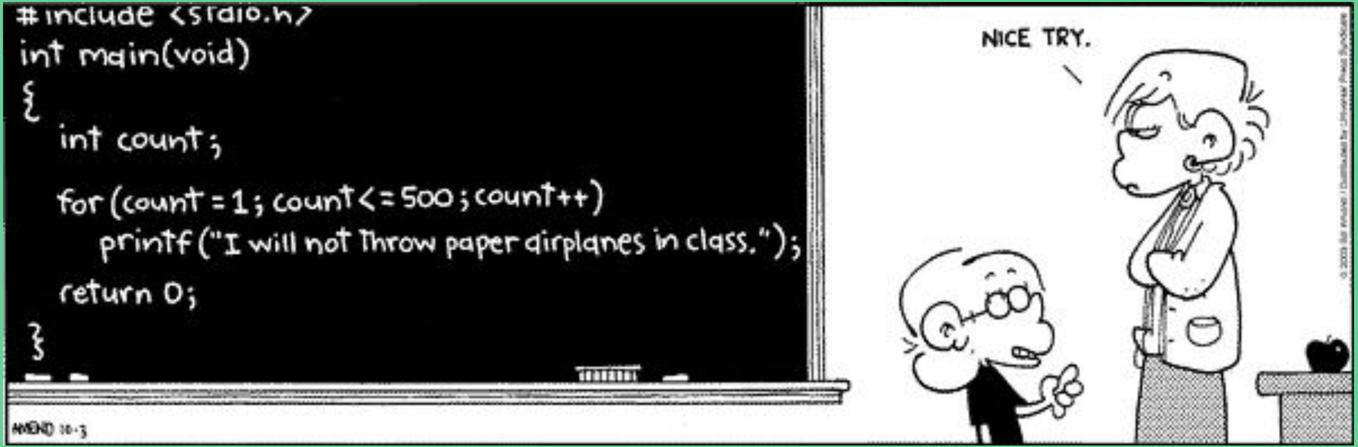
Recap of last time: computing GC content with conditionals

- Can we do this with `elif` statements?
- Can we alert the user if the function gets incorrect input (a string of incorrect length)?

A **loop** is a procedure to repeat a piece of code.

(another way of saying this is that it **iterates** through code)

- Loops enable you to re-run blocks of code for as many times as you need.
- Python has two main ways to run loops: **`for & while`**



```
#include <stdio.h>
int main(void)
{
   int count;

   for(count=1;count<=500;count++)
      printf("I will not Throw paper airplanes in class.");
   return 0;
}
```

Bill Amend, FoxTrot, October 3, 2003

Objectives for this week

- Write a **`for`** loop to iterate over elements in an object
- Use a **counter** in a loop
- Use a **`for`** loop to count GC content and CCAAT boxes in a DNA string
- Write **`while`** loops and implement **`continue`** and **`break`**
- Develop a **growth mindset** towards your programming growth

`for` loop syntax

iteration variable

iter*able* variable

```
for value in a_list:
    print(value)
```

colon

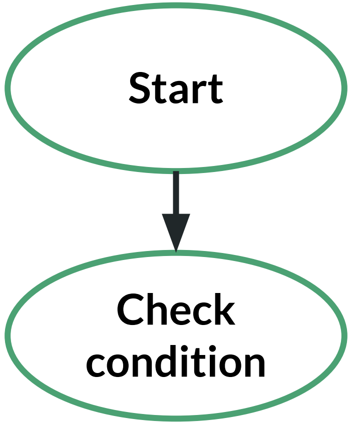
body of the loop

indented by 4 spaces (or tab)

A **for loop** is a procedure to repeat code for every element in a sequence.

`for` loop syntax

```
a_list = [1,2,3]

for value in a_list:

    print(value)
```

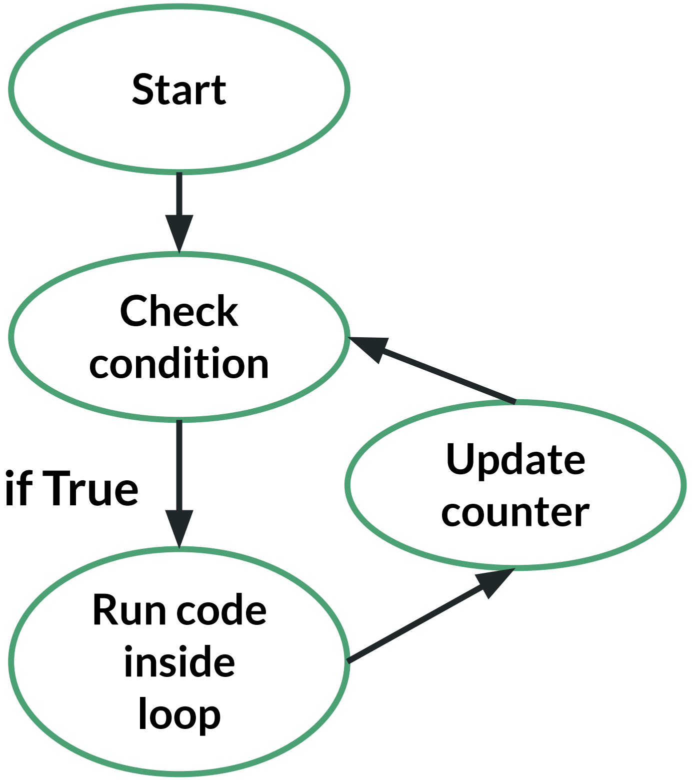


`for` loop syntax

```
a_list = [1,2,3]

for value in a_list:

    print(value)
```

```
1
```

output

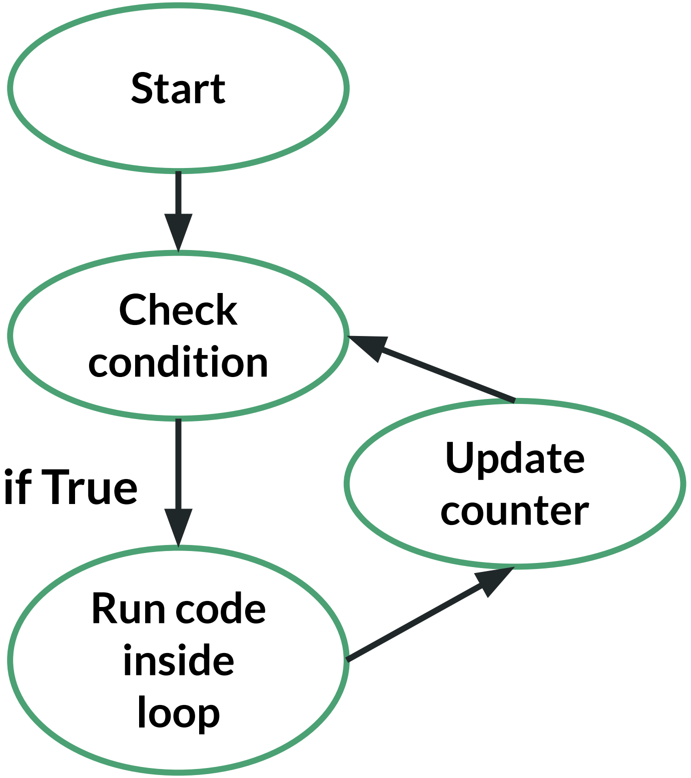


for loop syntax

```
a_list = [1,2,3]

for value in a_list:

    print(value)
```

```
1
2
```

output

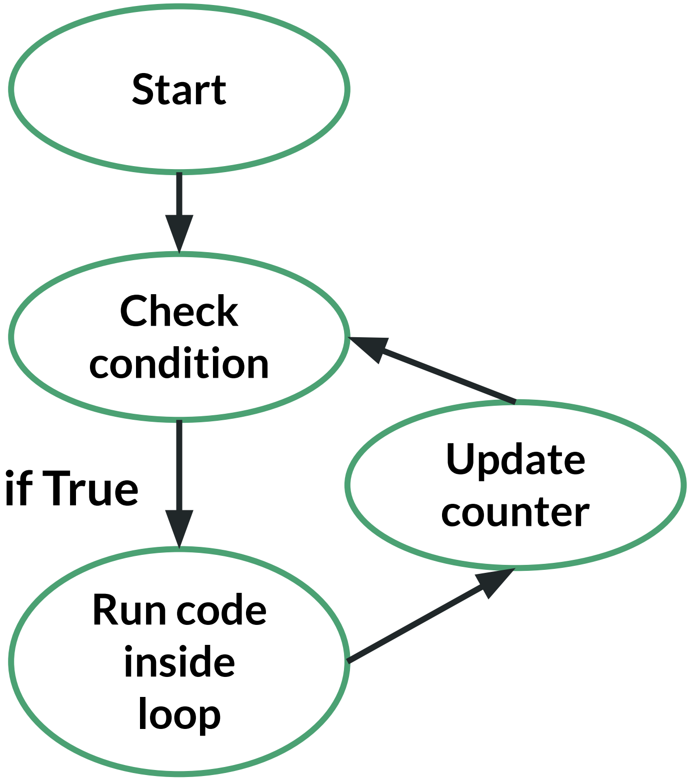


`for` loop syntax

```
a_list = [1,2,3]

for value in a_list:

    print(value)
```

```
1
2
3
```

output

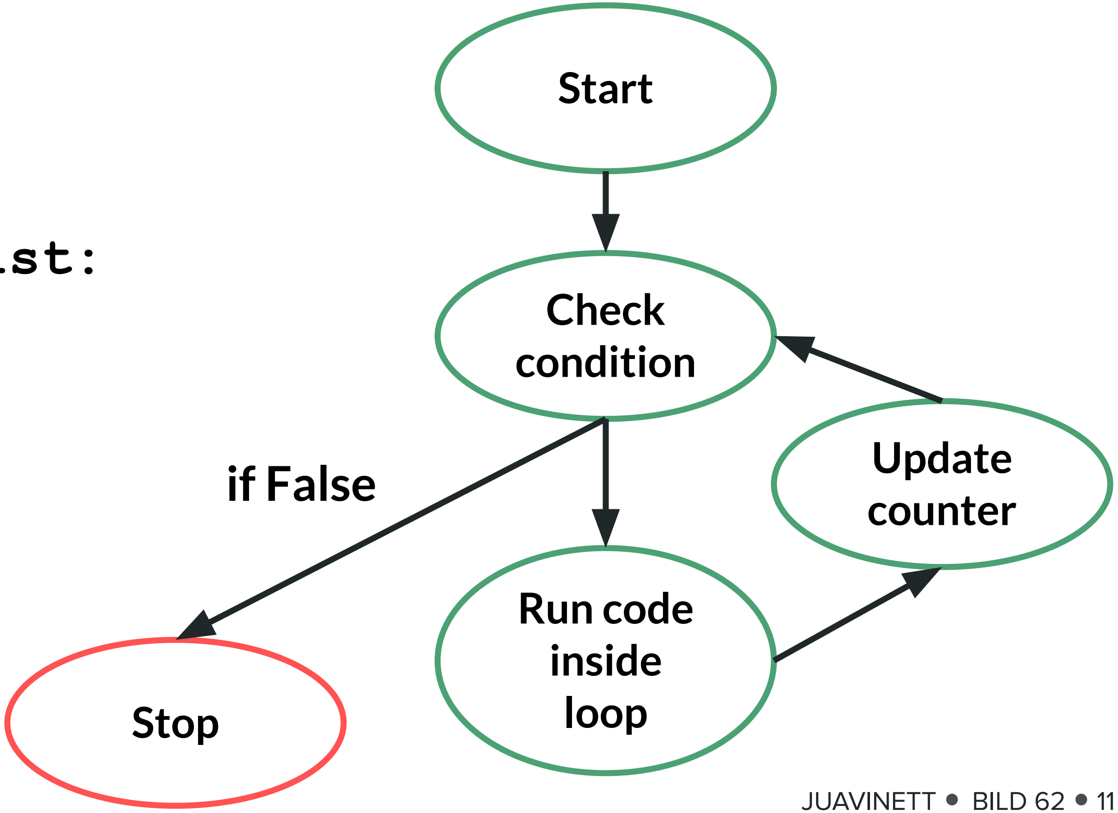


`for` loop syntax

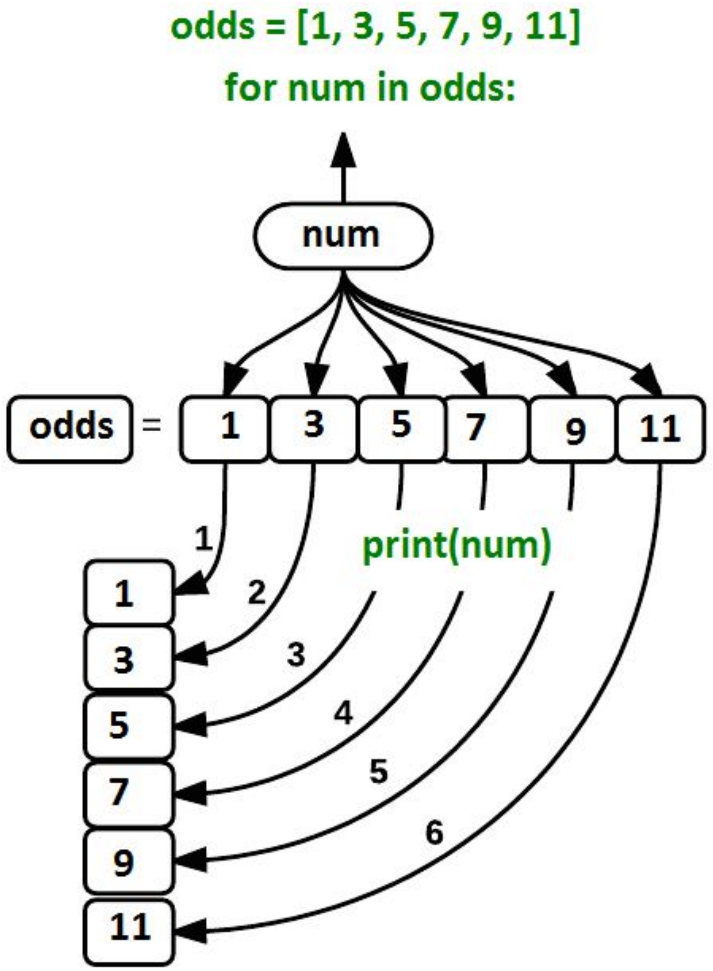
```
a_list = [1,2,3]

for value in a_list:

    print(value)
```

```
1
2
3
```

output



Another way to visualize working through a loop
(source)



efficiency benefit of `for` loops

Each of these would accomplish the same thing:

Option #1: 2+ lines of code

```
for value in a_list:
    print(value)
```

Option #2: as many lines of code as there are list entries

```
print(a_list[0])
print(a_list[1])
print(a_list[2])
...
```

Let’s create a ***living loop.***

We need to define an iterable variable

Second task: count the # of "CAT" boxes (CCAAT) in a string of DNA.

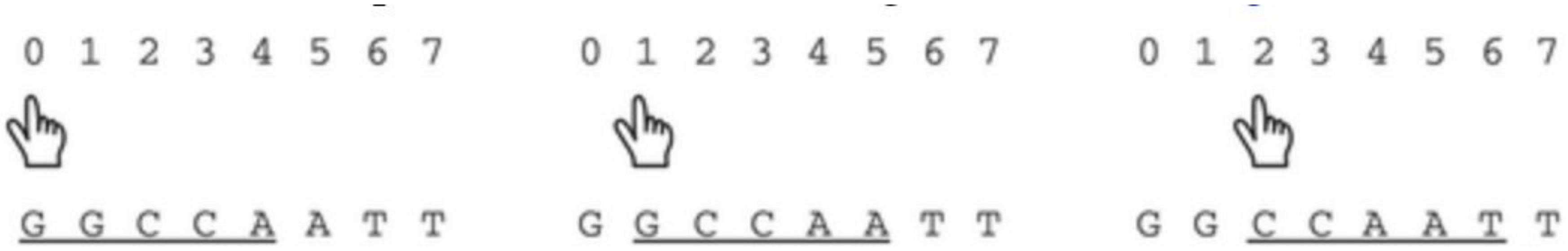
The "CAT" box generally appears near the spot where transcription begins!

```
>>> countCCAAT('GGCCAATTGCCAAT')
>>> 2
```

We can also loop over a list of indices!

Let's say we want to look for a "CAT" box, a common motif in DNA, with the sequence `CCAATT`

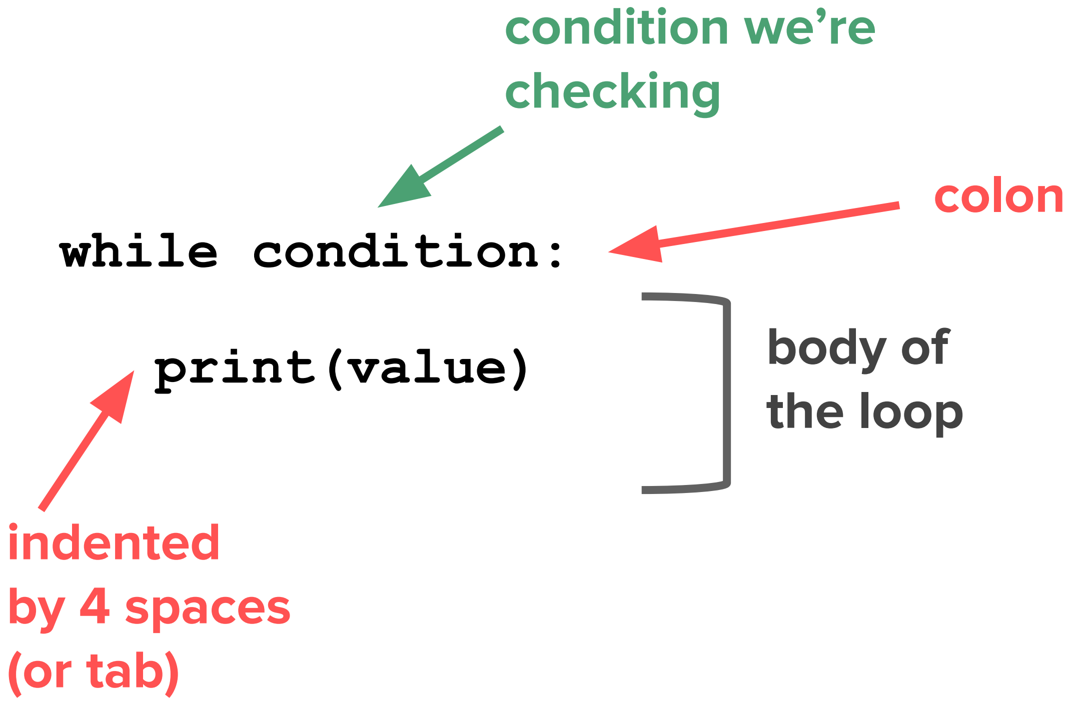
Since we want to look at a **slice** of DNA, rather than looping through individual items in the string, we need the indices.



Quick check in...

https://www.menti.com/blhpa9c4jpki

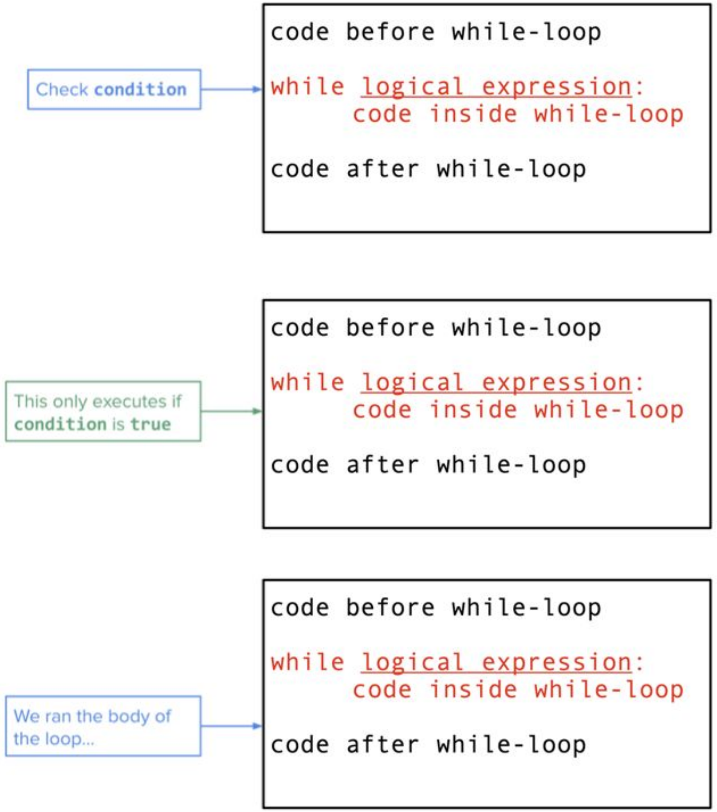
`while` loop syntax



```
while condition:
    print(value)
```

While this condition is true, the loop will run!

It will repeat until the condition is no longer True.



```
code before while-loop

while logical expression:
      code inside while-loop

code after while-loop
```

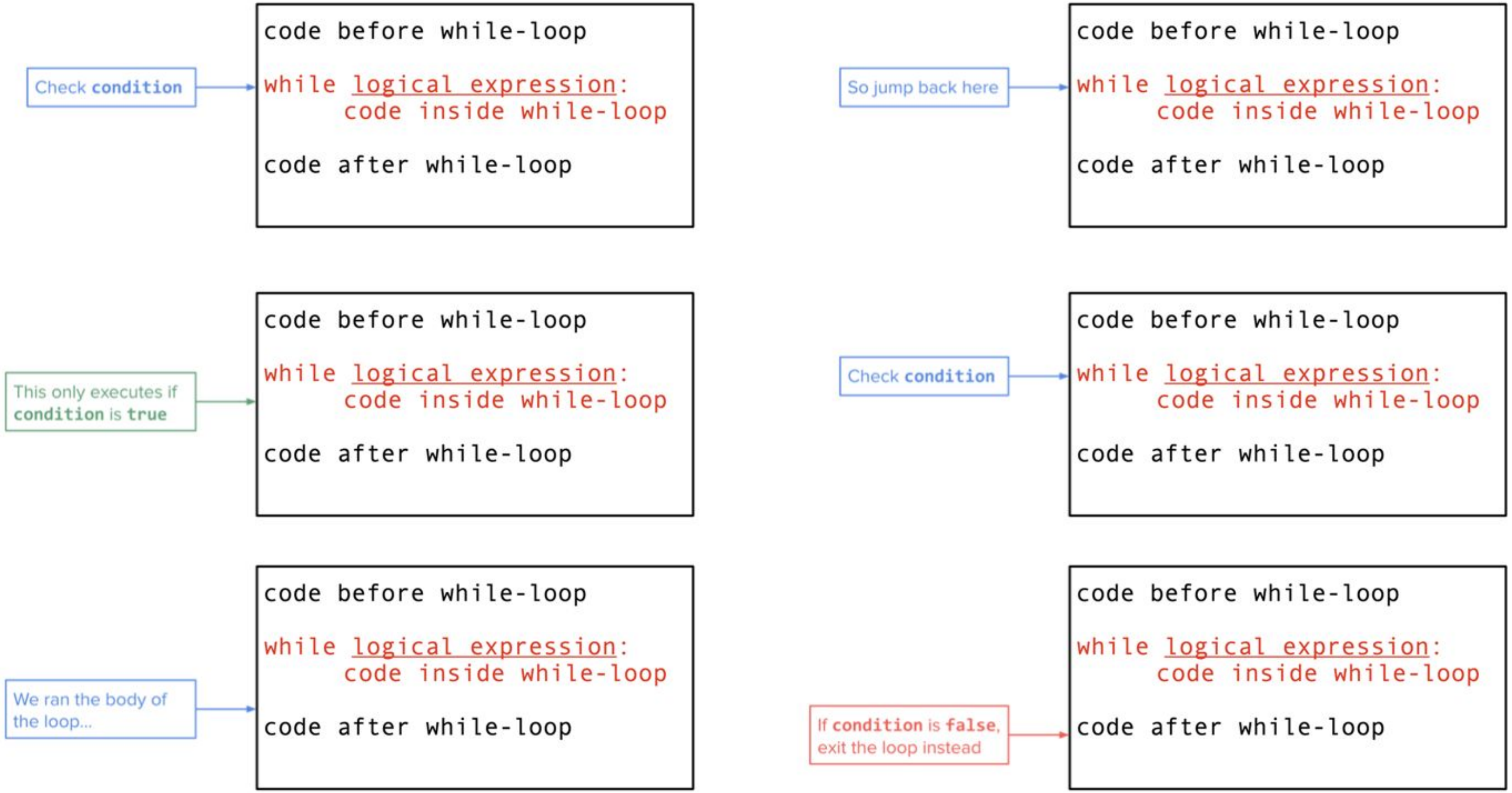
This only executes if condition is true

```
code before while-loop

while logical expression:
      code inside while-loop

code after while-loop
```

We ran the body of the loop...

```
code before while-loop

while logical expression:
      code inside while-loop

code after while-loop
```

Order of execution in a while loop (from Stepik)


Check condition
This only executes if condition is true
We ran the body of the loop...
So jump back here
Check condition
If condition is false, exit the loop instead

```
code before while-loop

while logical expression:
         code inside while-loop

code after while-loop
```

Order of execution in a while loop (from Stepik)

Mindsets about intelligence (and programming)

Fixed Mindset

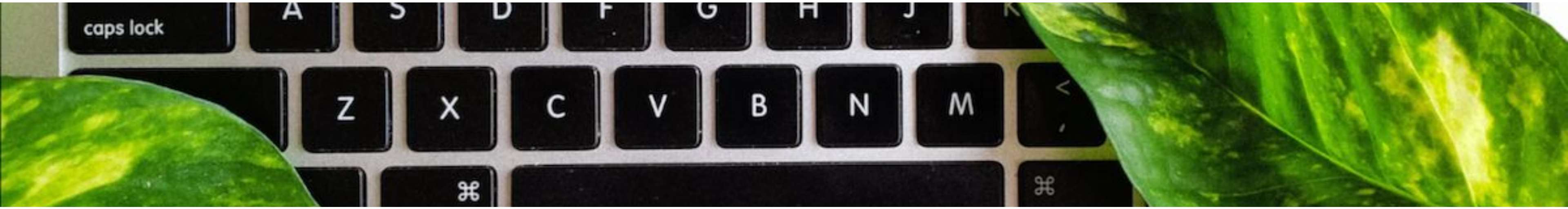
Human traits (including programming skills) are ***fixed/innate***.

Fixed mindset about programming: You have a certain amount of programming ability and *can't do anything to change it.*

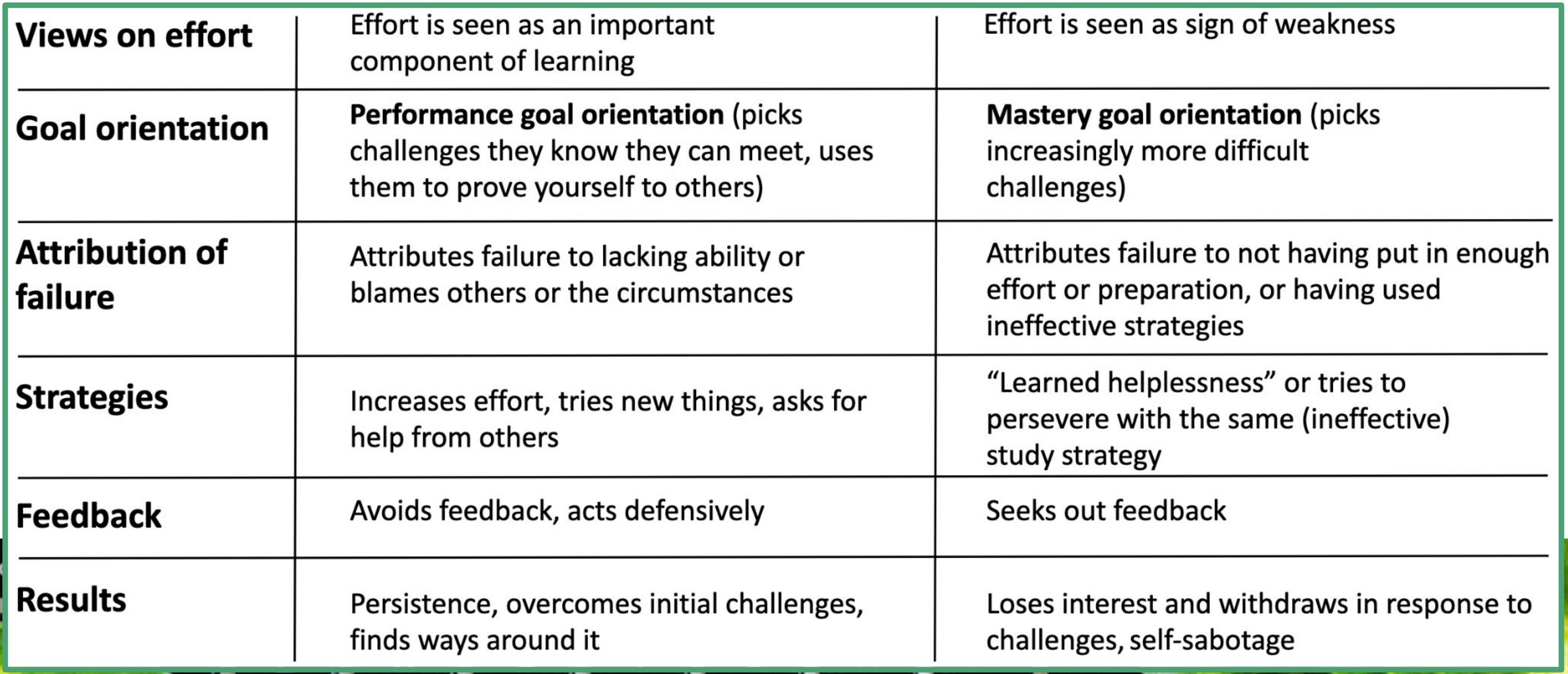
Growth mindset

Human traits (such as programming skills) are ***malleable and can be shaped/developed***.

Programming skill can be developed through personal effort, good learning strategies, and feedback.



Which of these are indicative of a growth mindset?



Views on effort	Effort is seen as an important component of learning	Effort is seen as sign of weakness
Goal orientation	**Performance goal orientation** (picks challenges they know they can meet, uses them to prove yourself to others)	**Mastery goal orientation** (picks increasingly more difficult challenges)
Attribution of failure	Attributes failure to lacking ability or blames others or the circumstances	Attributes failure to not having put in enough effort or preparation, or having used ineffective strategies
Strategies	Increases effort, tries new things, asks for help from others	“Learned helplessness” or tries to persevere with the same (ineffective) study strategy
Feedback	Avoids feedback, acts defensively	Seeks out feedback
Results	Persistence, overcomes initial challenges, finds ways around it	Loses interest and withdraws in response to challenges, self-sabotage

How do these individuals demonstrate growth mindsets?

Learn more about their beginnings in programming!

Topics from this lecture & corresponding notebook

- Syntax of **for** and **while** loops
- How to iterate through strings, lists, and dictionaries
- Using a counter to count loop iterations
- Looping over lists of indices
- Calling functions within functions
- Using **`break`** to interrupt a loop, and **`continue`** to skip a loop
- Functions we learned: **`range(), enumerate()`**

Resources

[Stepik Introduction to Python book, Chapter 3]

[Whirlwind Tour of Python: Control Flow]