Module 2. Lesson 4.

Loops. Continued
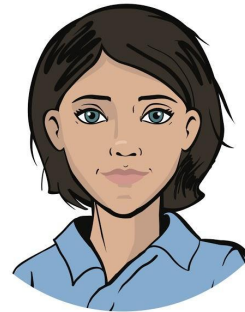
Discussion:

A big order from a social network
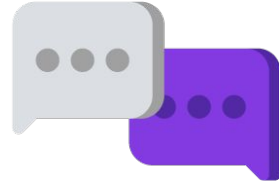
A new order

An emerging **social network called "Friend Around"** is asking for help. Investors are ready to support their project if two conditions are met:

❏ availability of a <u>user authorization</u> system with a login and password;

❏ availability of a <u>user personal account</u> with a smart news feed, music recommendations, and contests.

Let's try to cope with this task?
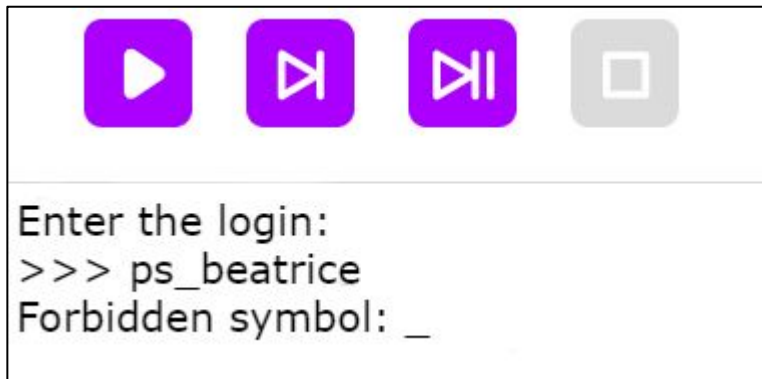
Emily,
Project Manager

User registration

Sample task.
Only letters of the Latin alphabet and numbers can be used in logins. The following symbols are forbidden: **=?*^$№@_,;:#%^&()**. The program must ask for the login and print the forbidden symbols, if any.

```
Enter the login:
>>> ps_beatrice
Forbidden symbol: _
```

How to analyze a login?

User registration

Sample task.
Only letters of the Latin alphabet and numbers can be used in logins. The following symbols are forbidden: **=?*^$№@_,;:#%^&()**. The program must ask for the login and print the forbidden symbols, if any.

=	?	*	^	$	№	@	_	,	;	#	%	&

Deny string

p	s	_	s	m	i	t	h	s	o	n

Login string

A possible solution is to look through all the login characters one by one and, for each of them, check whether it is forbidden (is in the deny string) or not.

```
Enter the login:
>>> ps_beatrice
Forbidden symbol: _
```
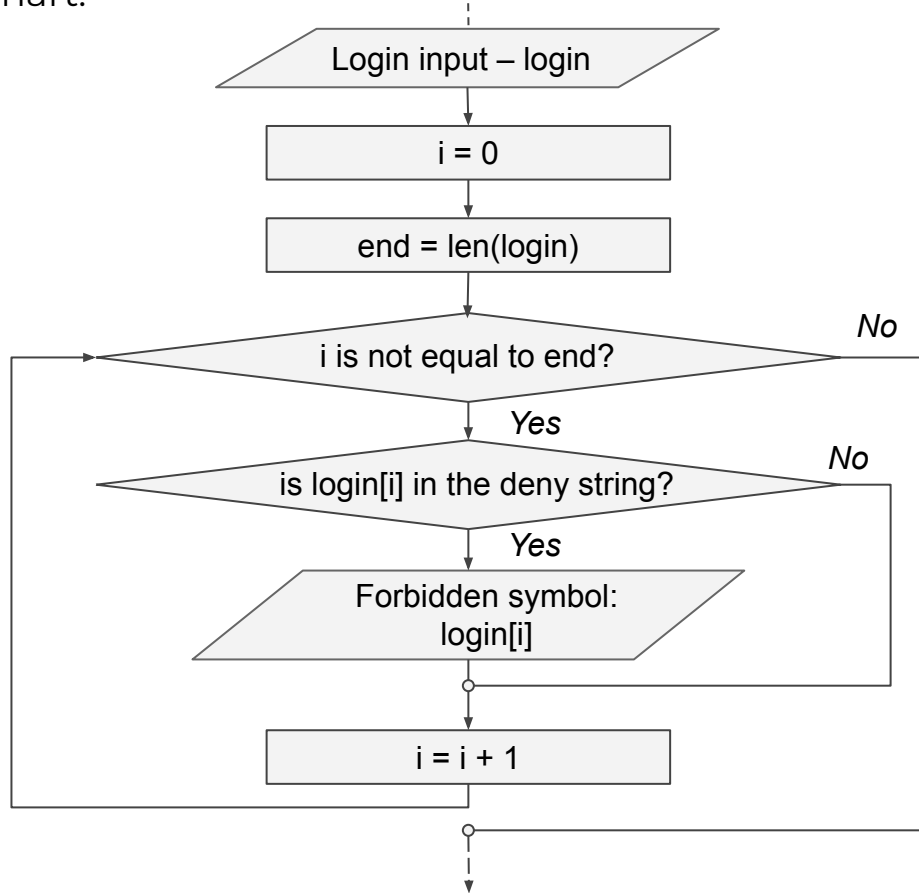
User registration

Sample flowchart:
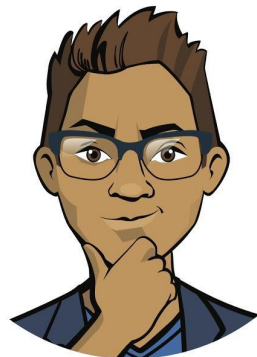
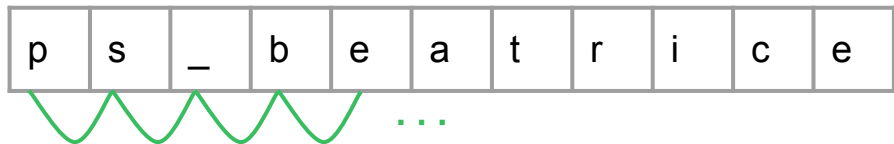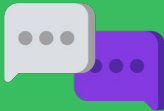# The optimal solution

The use of a counter is necessary to program any action that is repeated n times.

But a string is a **sequence of characters**. The interpreter sees that the characters in the string are connected and can <u>iterate over them without a counter</u>!

p	s	_	b	e	a	t	r	i	c	e

...

Cole,
Senior Developer
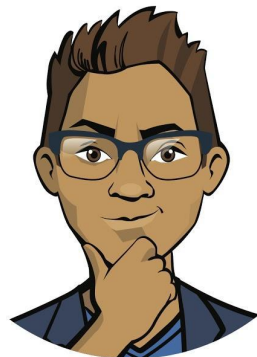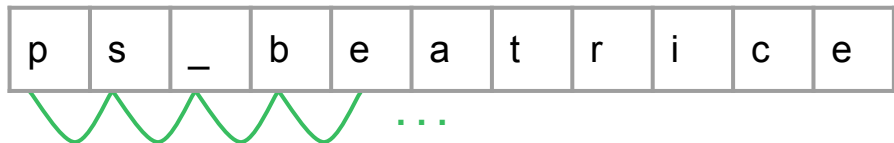
The optimal solution

Iteration over elements in a sequence is a handy and powerful Python tool.

To master it, we will study a new type of loop – the **for** loop.

p	s	_	b	e	a	t	r	i	c	e

...

Cole,
Senior Developer

The goal of the workday is to

program user authorization and personal accounts for a social network.

Use the optimal tool in your work – the for loop.

Today you will:

- learn that the for loop can be used to iterate over the elements of a sequence;

- learn and compare for and while loops;

- introduce new mechanics into the "Friend Around" social network.

Confirmation of qualifications

To get started on the working tasks, demonstrate **your knowledge level** .

Prove that you are ready for the brainstorm!

What is a **loop**?
Which <u>loop operator</u> do you know?

Loop

– a command that executes actions given as long as a certain logical expression (condition) remains true.

```
while  Expression is true  :
```

An action block starts with a colon

Execute action 1

Execute action 2

Execute action 3

Execute action 4

4 spaces

What is a counter ?
What is it used for?

Counter

– a variable storing the number of steps of a certain loop.

The counter can store:

- All the steps of a loop.
- Loop steps for which a certain condition is met.

Counter

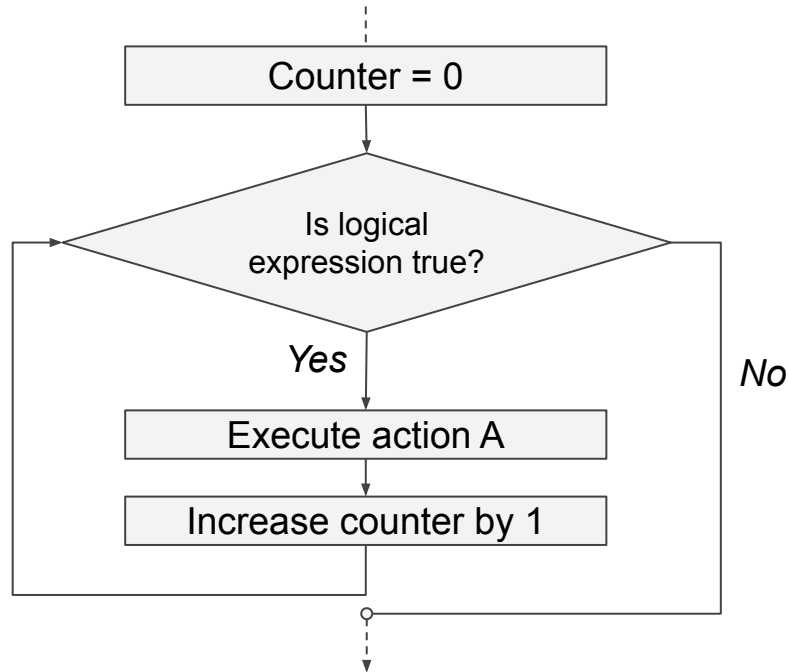**– a variable storing the number of steps of a certain loop.**

Example 1:
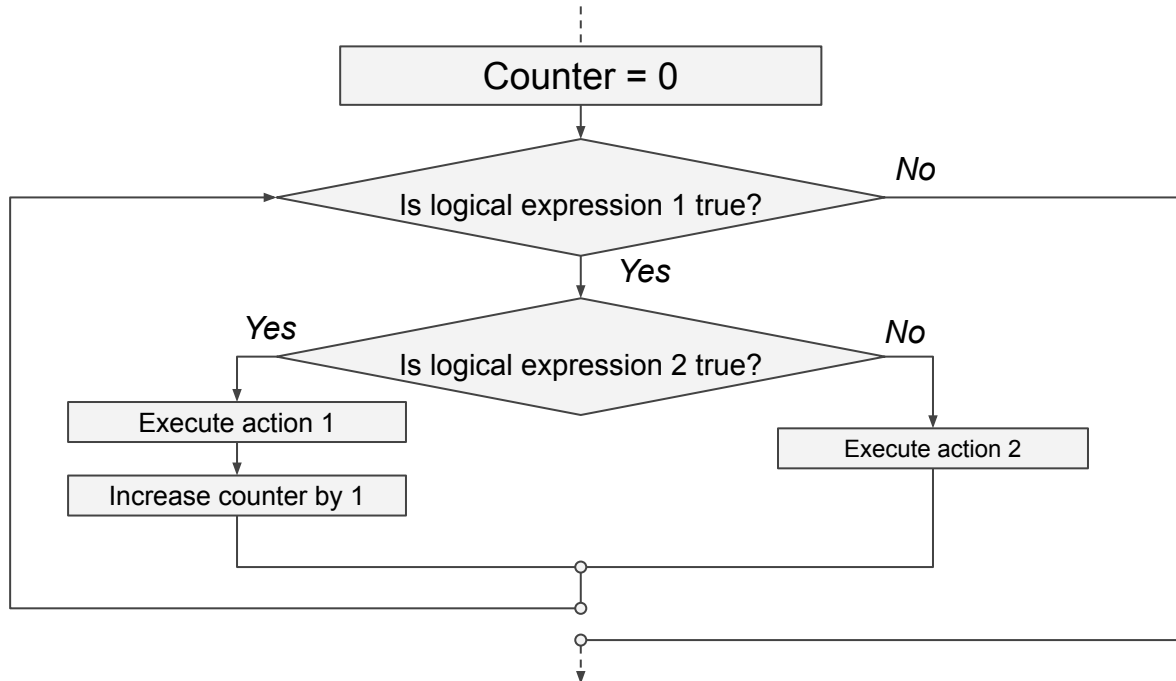Counter storing all the loop steps.

Counter

– a variable storing the number of steps of a certain loop.

Example 2:
Counter storing all the loop steps where the condition is true.

```
                    Counter = 0
                         │
                         ▼
         ┌──── Is logical expression 1 true? ────► No
         │               │                         │
         │              Yes                        │
         │               ▼                         │
  Yes ◄──── Is logical expression 2 true? ────► No │
         │                                        │ │
         ▼                                        ▼ │
  Execute action 1                        Execute action 2
         │                                        │
         ▼                                        │
  Increase counter by 1                           │
```

Complete the task

Write a program that prompts you to enter three user preferences. After every preference has been entered, the program prints: "Preference taken into account!". After all the preferences have been entered, the program prints: "Recommendation system is set up!".

```
Enter your wish:
>>> tik tokers
Preference taken into account
Enter your wish:
>>> Marvel comics
Preference taken into account
Enter your wish:
>>> games
Preference taken into account
Recommendation system is set up!
```

Complete the task

<u>Write a program</u> that prompts you to enter three user preferences. After every preference has been entered, the program prints: "Preference taken into account!". After all the preferences have been entered, the program prints: "Recommendation system is set up!"

```python
i = 0

while i != 3:

    wish = input('Enter your wish:')

    print('Preference taken into account')

    i += 1

print('Recommendation system is set up!')
```

```
Enter your wish:
>>> tik tokers
Preference taken into account
Enter your wish:
>>> Marvel comics
Preference taken into account
Enter your wish:
>>> games
Preference taken into account
Recommendation system is set up!
```
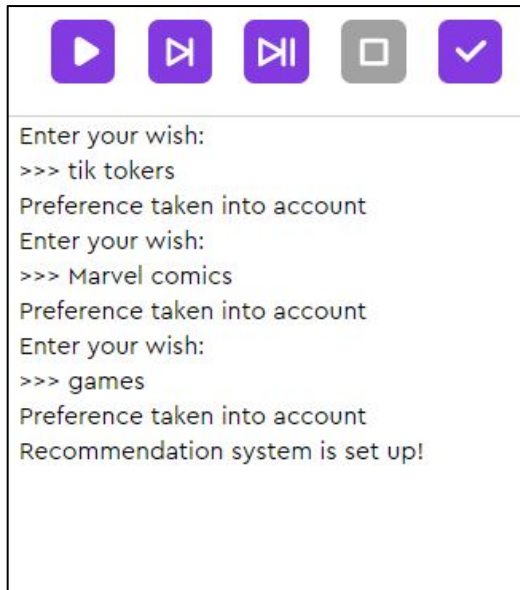
Qualifications confirmed!

Great! You are ready to brainstorm and complete your work task!

Brainstorm:

The for loop
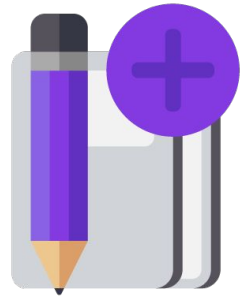
Where is the for loop used?

The **for** loop

Iteration over the sequence elements

Repeating actions in the loop body n times

Important tool:

The **in** operator

Important tool:

The **range()** function

Where is the for loop used?

The **for** loop

Iteration over the sequence elements

Repeating actions in the loop body n times

Important tool:

The **in** operator

Important tool:

The **range()** function
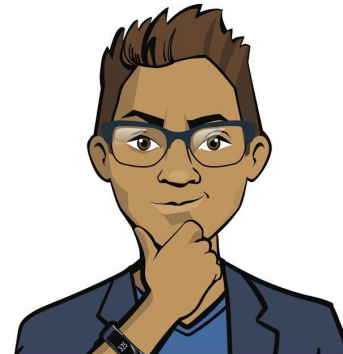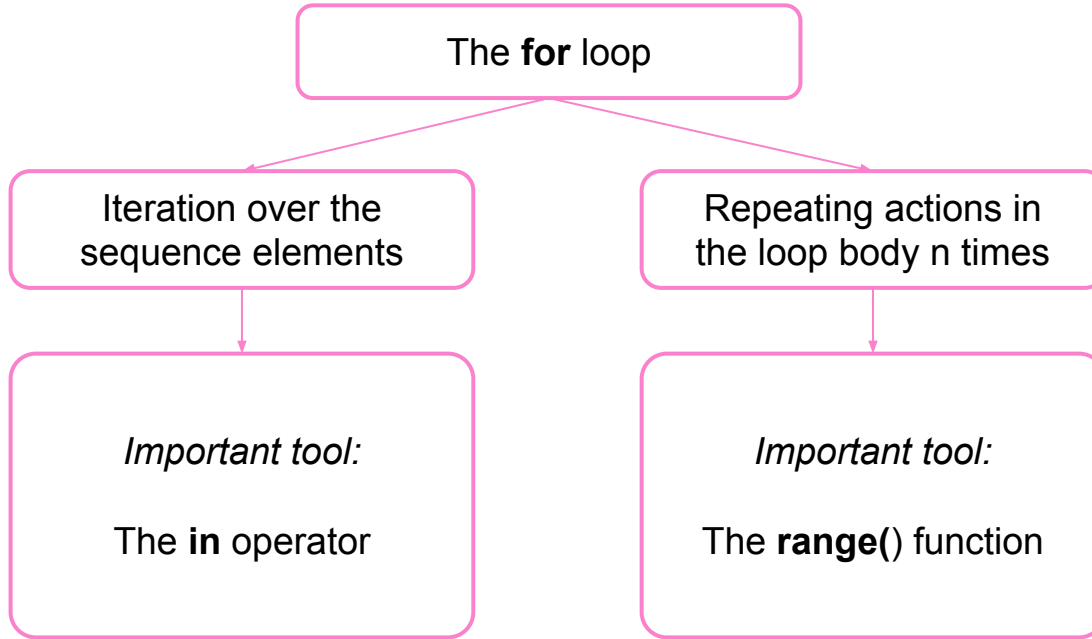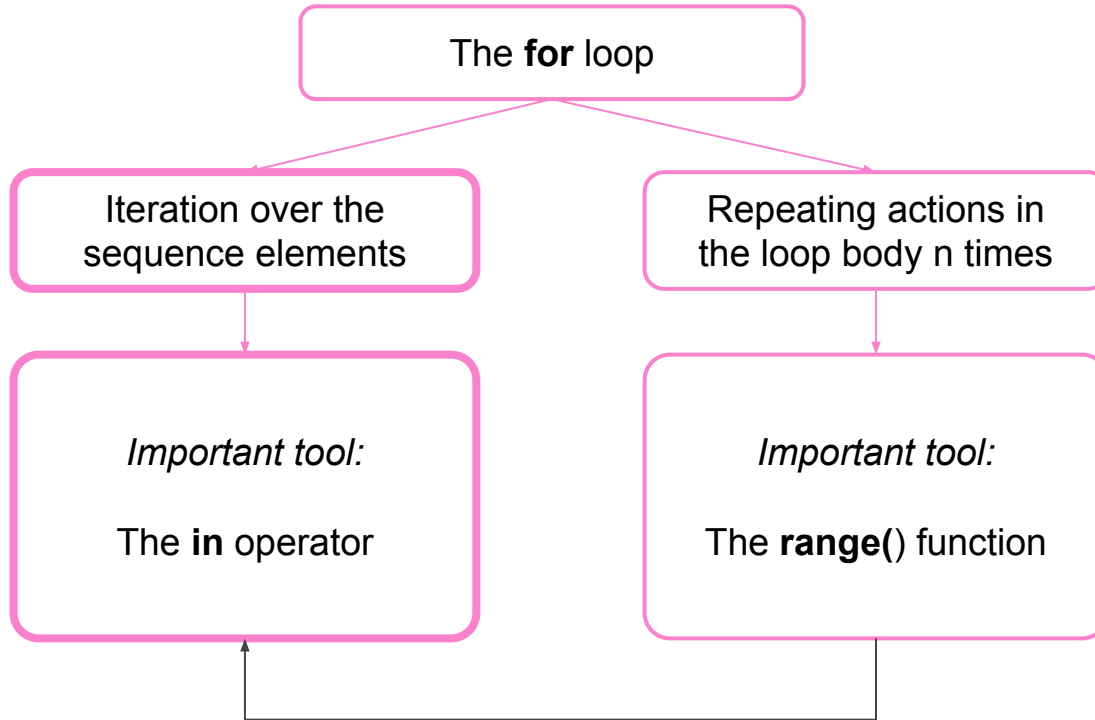
In fact, the second case comes down to the first one!
Let's figure out how the loop works inside.
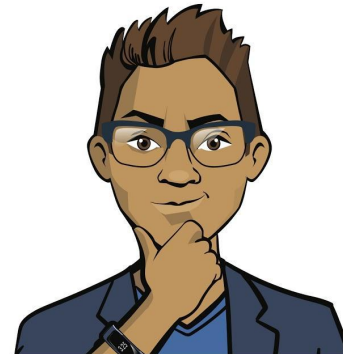
The for loop
— a loop iterating over the elements of a finite sequence.

A sequence is an ordered set of elements.

```
for element in sequence :
```

An action block starts with a colon

Execute action 1

Execute action 2

Execute action 3

Execute action 4

4 spaces

The for loop
— **a loop iterating over the elements of a finite sequence.**

A sequence is an ordered set of elements.

```
for  element  in  sequence :
        Execute action 1
        Execute action 2
        Execute action 3
    Execute action 4
```
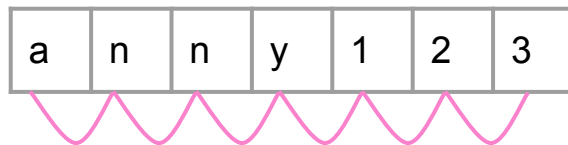
*The interpreter <u>itself</u> determines the **beginning** of the sequence, its **end**, and the **order** of the elements.*

<u>No counter needed</u>!

a	n	n	y	1	2	3

The for loop
— a loop iterating over the elements of a finite sequence.

Encrypting the password by letter number in the alphabet:

```python
password = 'kitty'

alphabet = 'abcdefghijklmnopqrstuvwxyz'

for symbol in password:
    print(alphabet.find(symbol) + 1)
```

```
11
9
20
20
25
```
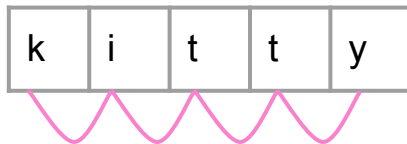
k	i	t	t	y

The for loop
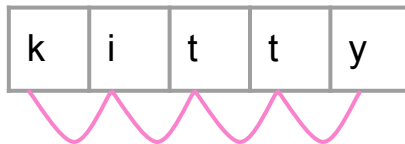— a loop iterating over the elements of a finite sequence.

Encrypting the password by letter number in the alphabet:

```
password = 'kitty'

alphabet = 'abcdefghijklmnopqrstuvwxyz'

for symbol in password:
    print(alphabet.find(symbol) + 1)
```

How many times will the loop work? Why exactly this many?

```
11
9
20
20
25
```

k	i	t	t	y

Brainstorm

Let's go back to the task

Task 1. Only letters of the Latin alphabet and numbers can be used in logins. The following symbols are forbidden: **=?*^$№@_,;:#%^&()**. The program must ask for the login and print the forbidden symbols, if any.

```
Enter your login:
>>> kate_@smith
Forbidden symbol: _
Forbidden symbol: @
```

Let's go back to the task

Task 1. Only letters of the Latin alphabet and numbers can be used in logins. The following symbols are forbidden: **=?*^$№@_,;:#%^&()**. The program must ask for the login and print the forbidden symbols, if any.

```python
login = input('Enter your login:')

wrong = '=?*^$№@_'

for symbol in login:

    if symbol in wrong:

        print('Forbidden symbol:', symbol)
```
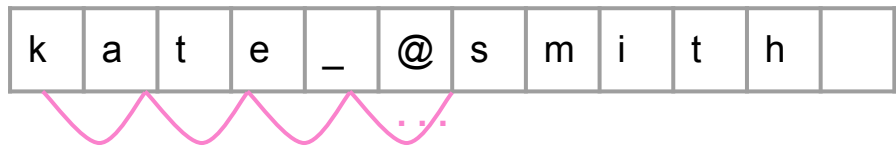
```
Enter your login:
>>> kate_@smith
Forbidden symbol: _
Forbidden symbol: @
```

k	a	t	e	_	@	s	m	i	t	h	

...

Before we continue:

1. How many steps will the loop take if you enter the login: "ag.sidorov"? Why?

2. What will the program print if you enter the login: "$tep@n555"?

3. Suppose we enter a login that has no forbidden characters. How many times will the loop work?

The for loop

can be programmed as a counter loop.

Since for iterates over the elements of a sequence, the for loop must iterate over numbers in the range from 0 to n-1 to repeat the actions n times.

```
for  element  in  range from 0 to 4  :
```

Repeating actions
of the loop body 5 times.

Execute action 1

Execute action 2

Execute action 3

Execute action 4

Brainstorm

The <small>for</small> loop

can be programmed as a counter loop.

Since for iterates over the elements of a sequence, the for loop must iterate over numbers in the range from 0 to n-1 to repeat the actions n times.

`for` | element | `in` | range from 0 to 4 | :

Repeating actions
of the loop body 5 times.

Execute action 1

Execute action 2

Execute action 3

Execute action 4

To create a time range for the for loop, the range() function is used.

The range() function

creates a sequence of numbers within the specified range.

range(n) — creates the sequence of numbers 0, 1, 2... n-1.

range(a, b) — creates the sequence of numbers a, a+1, a+2... b-1.

```
for element in range( range ):
```

Execute action 1

Execute action 2

Execute action 3

Execute action 4

The range() function

creates a sequence of numbers within the specified range.

range(n) — creates the sequence of numbers 0, 1, 2... n-1.

range(a, b) — creates the sequence of numbers a, a+1, a+2... b-1.

Prompting the user to enter three preferences:

```python
for i in range(3):
    wish = input('Enter your wish:')
    print('Preference taken into account')
print('Recommendation system is set up!')
```

Enter your wish:
>>> tik tokers
Preference taken into account
Enter your wish:
>>> Marvel comics
Preference taken into account
Enter your wish:
>>> games
Preference taken into account
Recommendation system is set up!

Brainstorm

Let's go over a task

Task 2. Write a program to create a group chat. The number of people is entered from the keyboard. Then, one by one, the names of the users to be added to the chat are entered. In response to each name, the program prints: "Welcome, <name>!" After all the names have been entered, a message is displayed: "Group chat created!"

```
Number of participants:
>>> 3
Enter name:
>>> Kate
Welcome, Kate
Enter name:
>>> John
Welcome, John
Enter name:
>>> Bess
Welcome, Bess
Group chat created!
```
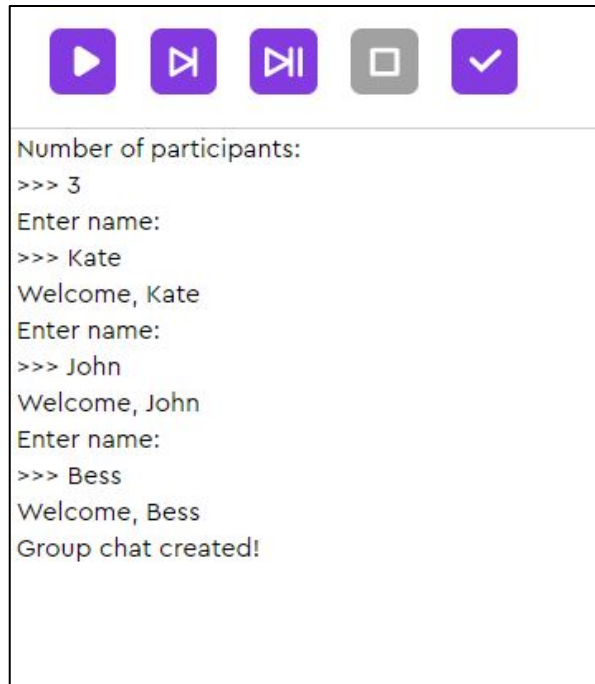
Brainstorm

Let's go over a task

Task 2. Write a program to create a group chat. The number of people is entered from the keyboard. Then, one by one, the names of the users to be added to the chat are entered. In response to each name, the program prints: "Welcome, <name>!" After all the names have been entered, a message is displayed: "Group chat created!"

```python
amount = int(input('Number of participants:'))
for i in range(amount):
    name = input('Enter name:')
    print('Welcome,', name)
print('Group chat created!')
```

0	1	2	3

```
Number of participants:
>>> 3
Enter name:
>>> Kate
Welcome, Kate
Enter name:
>>> John
Welcome, John
Enter name:
>>> Bess
Welcome, Bess
Group chat created!
```
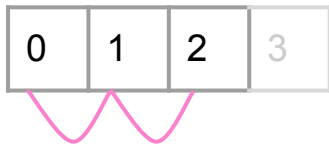
Brainstorm

Before we continue:
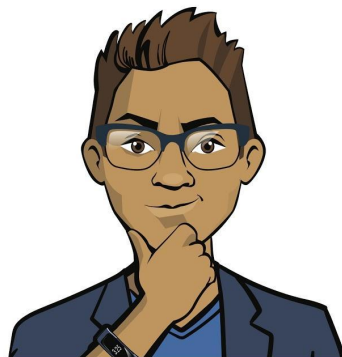
1. How many steps will the loop take if you enter 5 participants? 10? 7? Why?

2. How does the solution change if you specify a range of values as `range(0, amount)`? `range(0, 3)`? `range(amount-1)`?

Platform:

Friend Around

Do the task on the platform

➡️ **"Friend Around: Authorization"**

mars.alg.academy
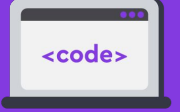
Friend Around: Personal account

Break

Brainstorm:

Nested constructs

A user's personal account

You need to program a personal account for the social network "Friend Around". The user will be able to use it, for example:

- to receive movie and music recommendations according to their mood
- to participate in contests and win prizes

and much more.

Let's go over a task

Task 1. Create a program to authorize the social network administrator by login and password. There are **three attempts** to enter. If the data is entered correctly (login: admin, password: kjf3g), the program prints: "Authorization at attempt <number>".
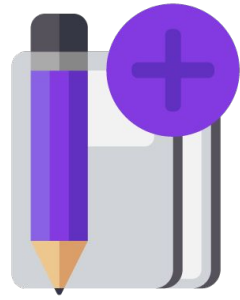
```
Login:
>>> admin
Password:
>>> ksjdk
Login:
>>> admin
Password:
>>> kjf3g
Authorization at attempt 2
```

How do we complete this task?

Let's go over a task

Task 1. Create a program to authorize the social network administrator by login and password. There are **three attempts** to enter. If the data is entered correctly (login: admin, password: kjf3g), the program prints: "Authorization at attempt <number>".

```python
for i in range(3):

    login = input('Login:')

    password = input('Password:')

    if login == 'admin' and password == 'kjf3g':

        print('Authorization at attempt', i+1)
```
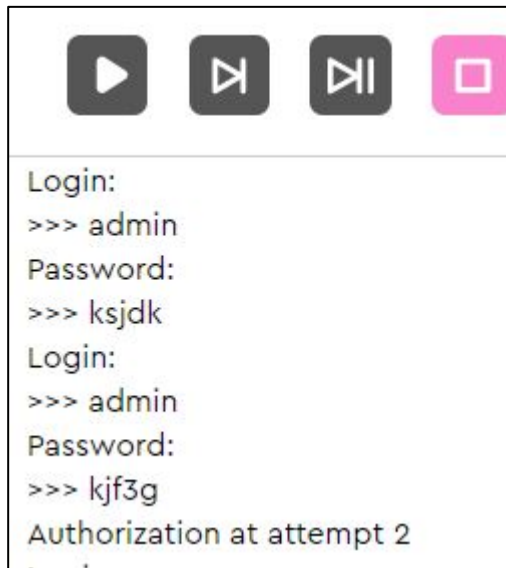
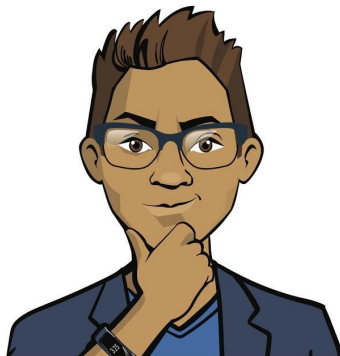*Let's go over a solution that uses for. Is it correct?*

Let's go over a task

Task 1. Create a program to authorize the social network administrator by login and password. There are **three attempts** to enter. If the data is entered correctly (login: admin, password: kjf3g), the program prints: "Authorization at attempt <number>".

```python
for i in range(3):

    login = input('Login:')

    password = input('Password:')

    if login == 'admin' and password == 'kjf3g':

        print('Authorization at attempt', i+1)
```

No! The loop will run 3 times even if correct data is received.

Describe a tool that could remedy the situation.

Login:
>>> admin
Password:
>>> hhasd
Login:
>>> admin
Password:
>>> kjf3g
Authorization at attempt 2
Login:
>>> admin
Password:
>>> kjf3g
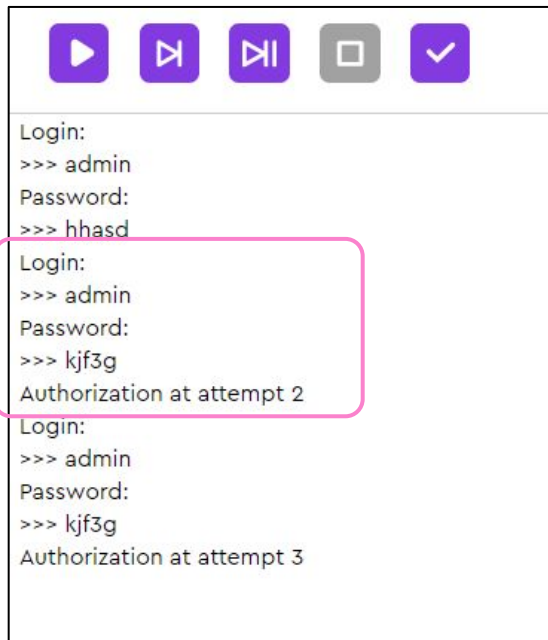Authorization at attempt 3

Brainstorm

Let's go over a task

Task 1. Create a program to authorize the social network administrator by login and password. There are **three attempts** to enter. If the data is entered correctly (login: admin, password: kjf3g), the program prints: "Authorization at attempt <number>".

break —— operator that terminates the loop ahead of schedule.

```
Login:
>>> admin
Password:
>>> hhasd
Login:
>>> admin
Password:
>>> kjf3g
Authorization at attempt 2
Login:
>>> admin
Password:
>>> kjf3g
Authorization at attempt 3
```
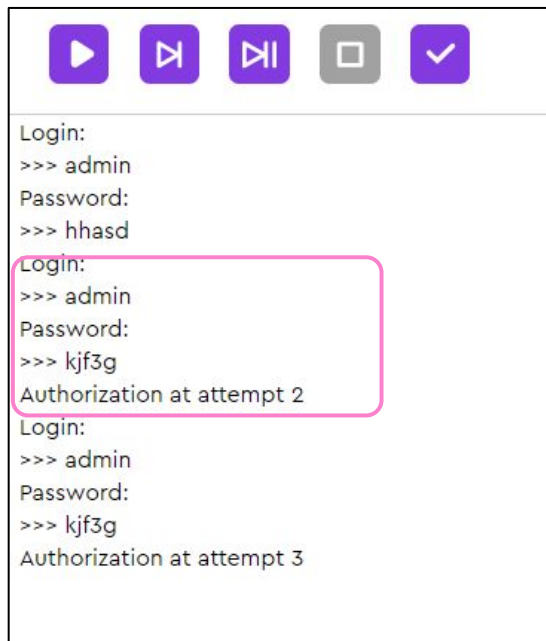
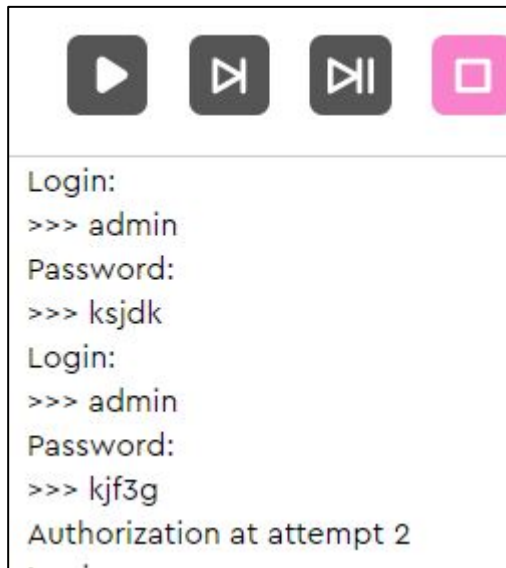*How can we fix the previous solution using the break operator?*

Let's go over a task

Task 1. Create a program to authorize the social network administrator by login and password. There are **three attempts** to enter. If the data is entered correctly (login: admin, password: kjf3g), the program prints: "Authorization at attempt <number>".

```python
for i in range(3):

    login = input('Login:')

    password = input('Password:')

    if login == 'admin' and password == 'kjf3g':

        print('Authorization at attempt', i+1)

        break
```

Login:
>>> admin
Password:
>>> ksjdk
Login:
>>> admin
Password:
>>> kjf3g
Authorization at attempt 2

Brainstorm

Let's go over a task

Task 2. Write a program for the entertainment section. When you enter "game", the game "Guess the Number" must start (the correct answer is 5). There are three attempts to answer. If the answer is correct, the program prints: "You have won a concert ticket!". The game can be played any number of times. When you enter "off", the program must exit.
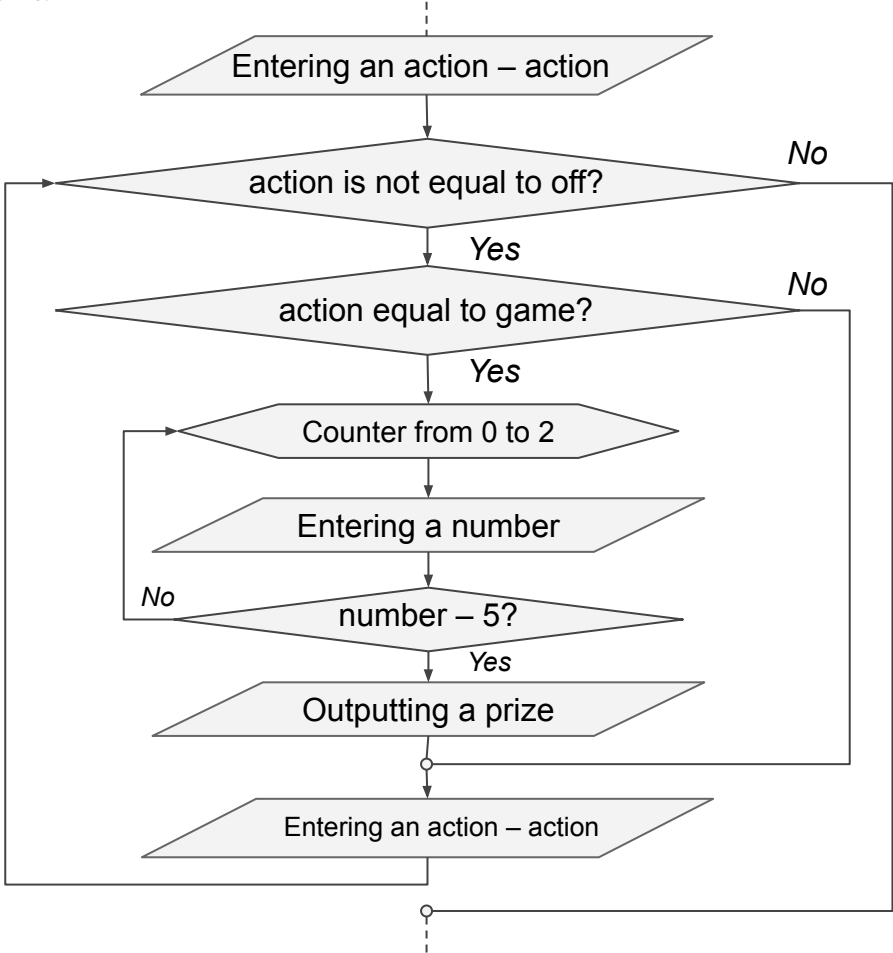
```
Enter game – "Guess the Number" game, off – exit
>>> 3
Enter game – "Guess the Number" game, off – exit
>>> 1
Enter game – "Guess the Number" game, off – exit
>>> 2
Enter game – "Guess the Number" game, off – exit
>>> game
Enter number
>>> 5
You have won a concert ticket!
Enter game – "Guess the Number" game, off – exit
>>> off
```
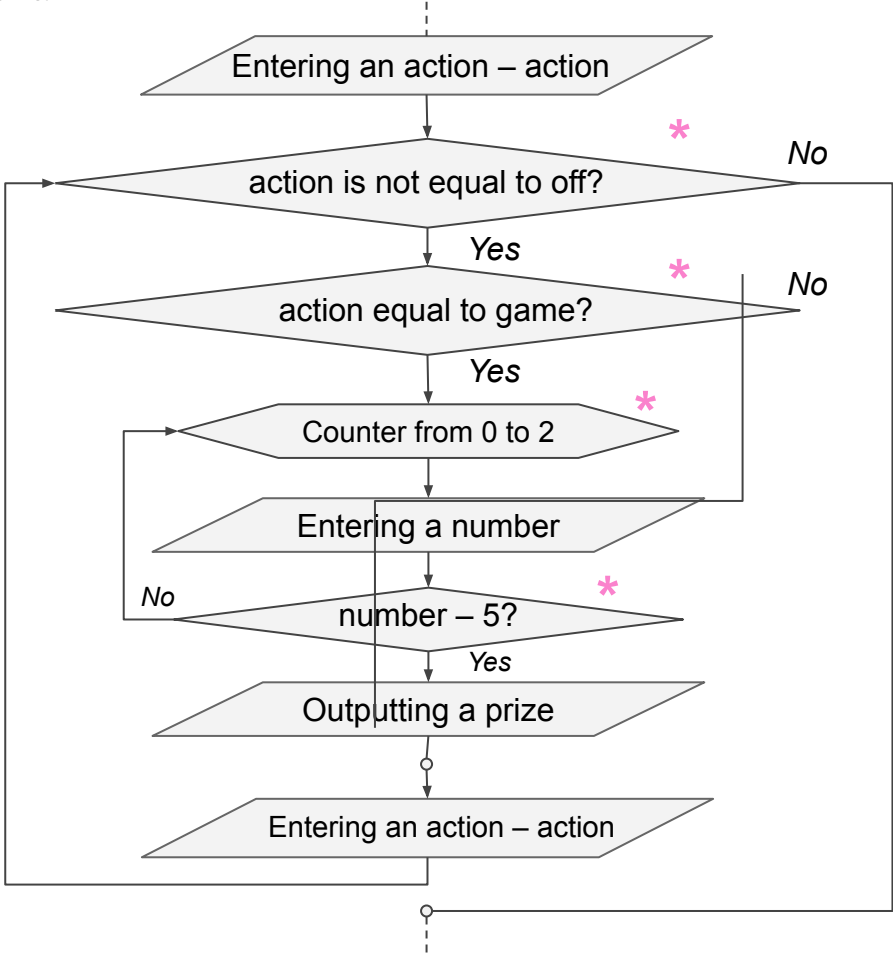
Sample flowchart:

Entering an action – action

action is not equal to off? — No

Yes

action equal to game? — No

Yes

Counter from 0 to 2

Entering a number

number – 5?

No

Yes

Outputting a prize

Entering an action – action

Brainstorm

Sample flowchart:

Name the operators that can be used in the marked blocks.

Brainstorm

Sample solution

Task 2. Write a program for the entertainment section. When you enter "game", the game "Guess the Number" must start (the correct answer is 5). There are three attempts to answer. If the answer is correct, the program prints: "You have won a concert ticket!". The game can be played any number of times. When you enter "off", the program must exit.

```python
action = input('Enter game - "Guess the Number" game, off - exit')
while action != 'off':
    if action == 'game':
        for i in range(3):
            if input('Enter number') == '5':
                print('You have won a concert ticket!')
                break
    action = input('Enter game - "Guess the Number" game, off - exit')
```

Before we continue:

1. How many steps will the <u>while loop</u> take if you sequentially enter "jokes", "game", "3", "5", "off"?

2. How many steps will the <u>for loop</u> take if you sequentially enter "game", "5", "off"?

3. How will the program work if you run out of all attempts in the "Guess the Number" game?
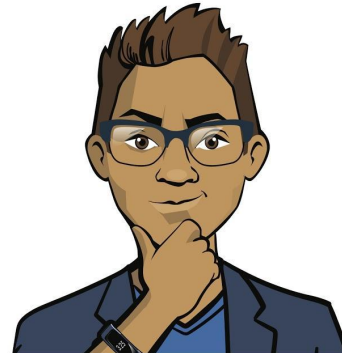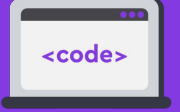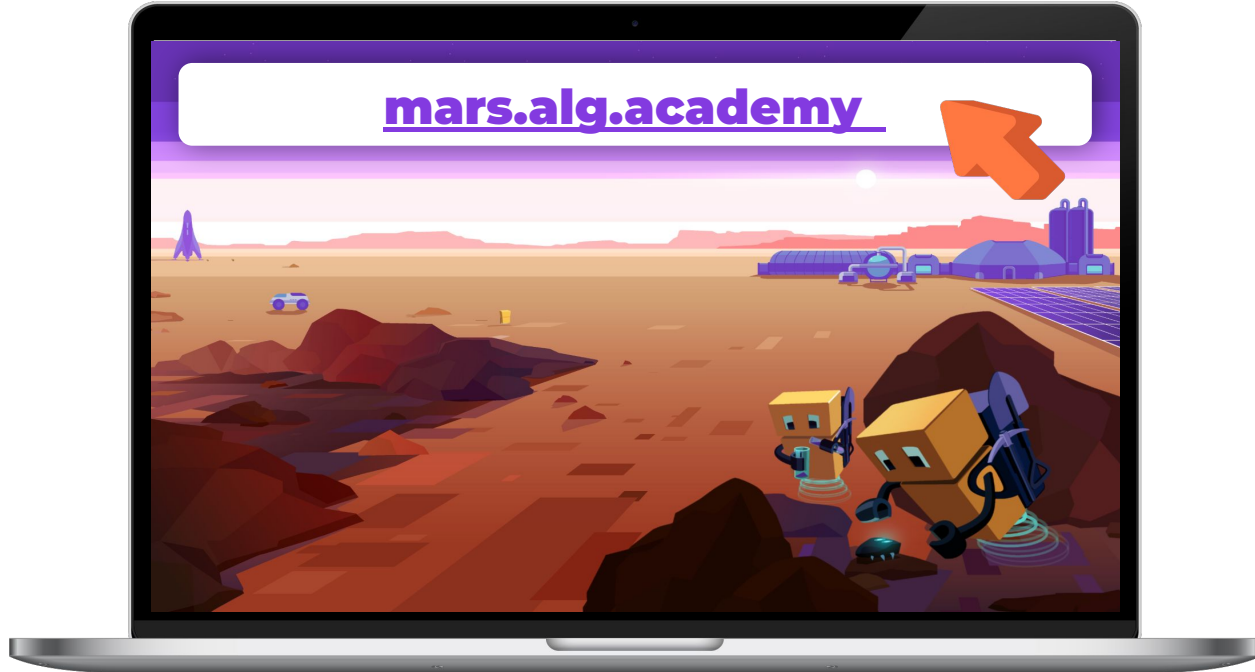
Platform:

Friend Around

Do the task on the platform

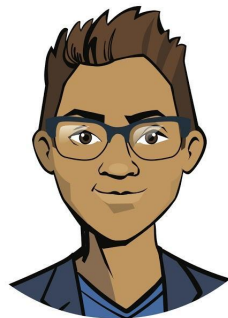➡️ **"Friend Around: Personal Account"**

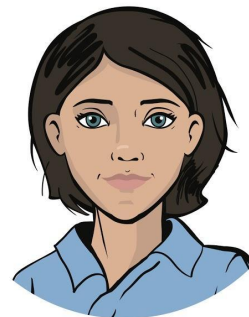mars.alg.academy

End of the workday

To complete the workday, pass a technical interview

1. What is the special feature of the for loop? In what cases is it especially useful?

2. Which operator allows you to interrupt the execution of the loop? When is it used?

Cole,
Senior Developer

Emily,
Project Manager

Performance review

Answer the questions together with your colleagues:

1. What was the best thing you managed to do?

2. What didn't work out the way you wanted?

3. What should you do next time to ensure success?

Additional tasks to improve efficiency

12/13

7

A friend around: authorization
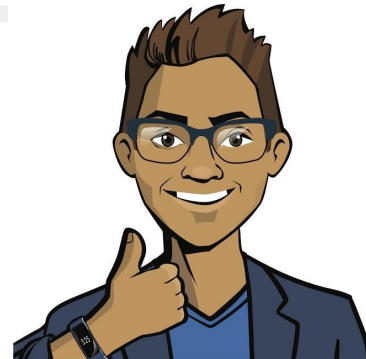
8/9

8

A friend around: personal account

★

A friend around: add'l tasks

"Friend Around: add'l exercises"

Theory: the for loop

Wrapping up the workday