

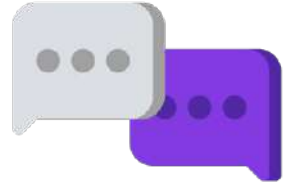
Module 1. Lesson 1.

Review.

Exception processing



Welcome to ProTeam!



Congratulations on starting work at ProTeam!

My name is Alena and I am a *project manager* at ProTeam.

I do the following:

- set tasks for employees and monitor their fulfillment;
- teach the rules for working on projects;
- transfer employees to new positions;
- organize events.



*Alena,
project manager*



Working at an
IT company



Congratulations on starting work at ProTeam!

My name is Cole and I am a *senior python developer* at ProTeam.

I do the following:

- software and application development in Python;
- assess the quality of programs written by others;
- assist interns.

We hope you will quickly feel at ease in the team and be able to show what great developers you are!



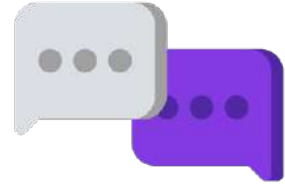
Cole,
senior developer



Working at an
IT company



Welcome back to ProTeam!



Welcome, developers!

We hope you haven't forgotten about your mentors!

My name is Alena and I am a *project manager* at ProTeam.

I do the following:

- set tasks for employees and monitor their fulfillment;
- teach the rules for working on projects;
- transfer employees to new positions;
- organize events.



*Alena,
project manager*



Discussing
work tasks



Welcome, developers!

We hope you haven't forgotten about your mentors!

My name is Cole and I am a *senior python developer* at ProTeam.

I do the following:

- software and application development in Python;
- assess the quality of programs written by other
- assist other developers.



Cole,
senior developer

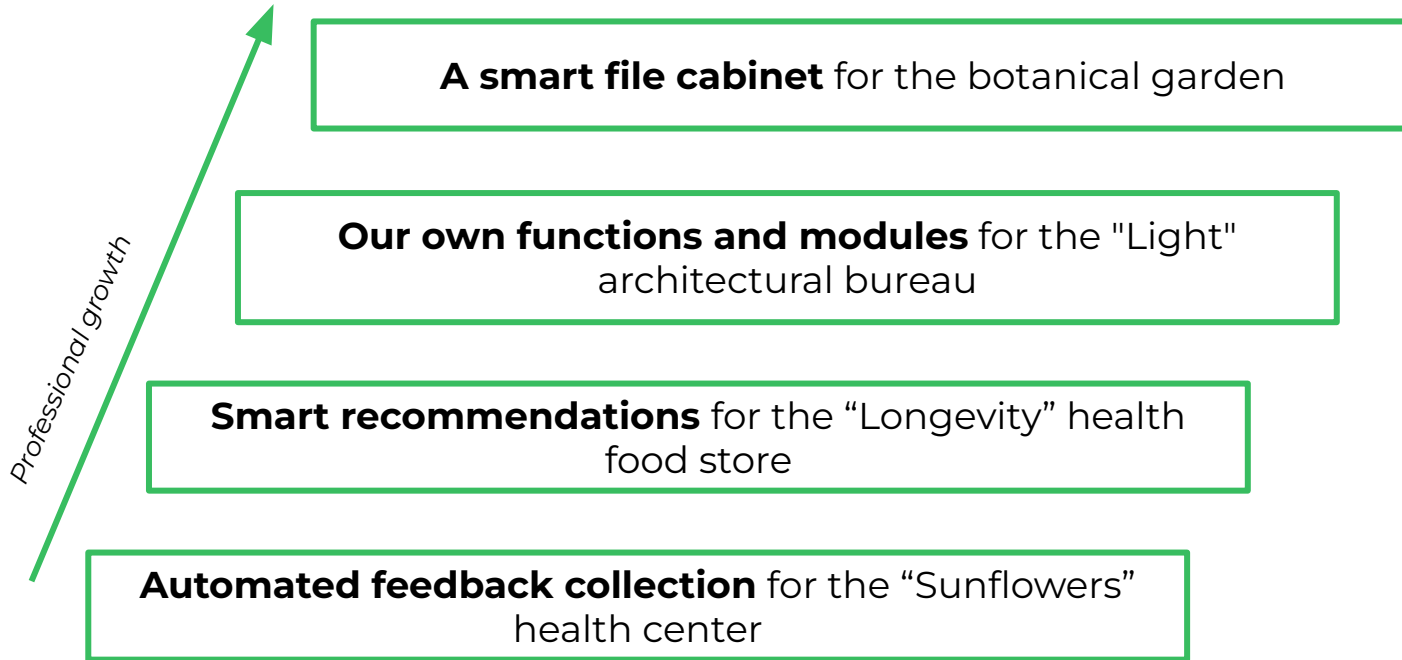


Discussing
work tasks



Welcome, developers!

Remember how many great orders we have already completed?



Discussing
work tasks



Welcome, developers!

A description of several upcoming projects is already stored in the task manager:

<i>Customer</i>	<i>Description</i>
The "Five out of Five" private school	A school knowledge base with information about academic performance, student records, etc.
Scientific Institute of Space Research	Application for smart notes with searching by tags.
The PlayForever game development studio	Create their own game library for interactive 2D novels .

And that's just the beginning!



Discussing
work tasks



Welcome, developers!

Difficult but interesting work tasks await you.

In order to cope with everything and not let your customers down, you need to know not only the basic Python constructions, but some other developer tools as well.

Ready?



*Cole,
senior developer*



**Discussing
work tasks**



The goal of the work day is

to recall the basic tools in Python and apply them to program the first customer request.

Today you will

- go through the qualifications for the Python developer position
- program an automatic knowledge assessment system;
- study the user notification tool about errors in the program.



Discussing
work tasks



Qualifications



**Show that you are ready
for a brainstorming
session!**

**Show your knowledge
of the programming
language
after a long break.**



Qualifications



What **data types** do you know?



Qualifications



What **data types** do you know?

We know four:

- **integers**,
- **decimals**,
- **strings**,
- **boolean** type.

	<i>Numbers</i>	<i>Strings</i>	<i>Boolean type</i>
144	<u>Integer</u> (int)	'John' (str)	True (bool)
48.3	Decimal (float)	'256' (str)	False (bool)
(2*11)	<u>Integer</u> (int)	'15.05.2007' (str)	0 <= 10 (bool)
(4*8.2)	Decimal (float)	'Data received' (str)	name != John (bool)



Qualifications



What is a **function? What **input** and **output** functions do you know?**



Qualifications



A function is an algorithm compiled in a programming language that has a unique name.

Example (data input and output functions):

<i>Function syntax</i>	<i>Purpose</i>	<i>Result</i>
<code>print ('Hello, hackers!')</code>	Information output on the screen	Hello, hackers!
<code>name = input('Your name:')</code>	Getting data from the keyboard and saving it to a variable	Your name: >>> Sasha

Will be saved in name.



Qualifications



What is a conditional operator?
What is it used for?



Qualifications

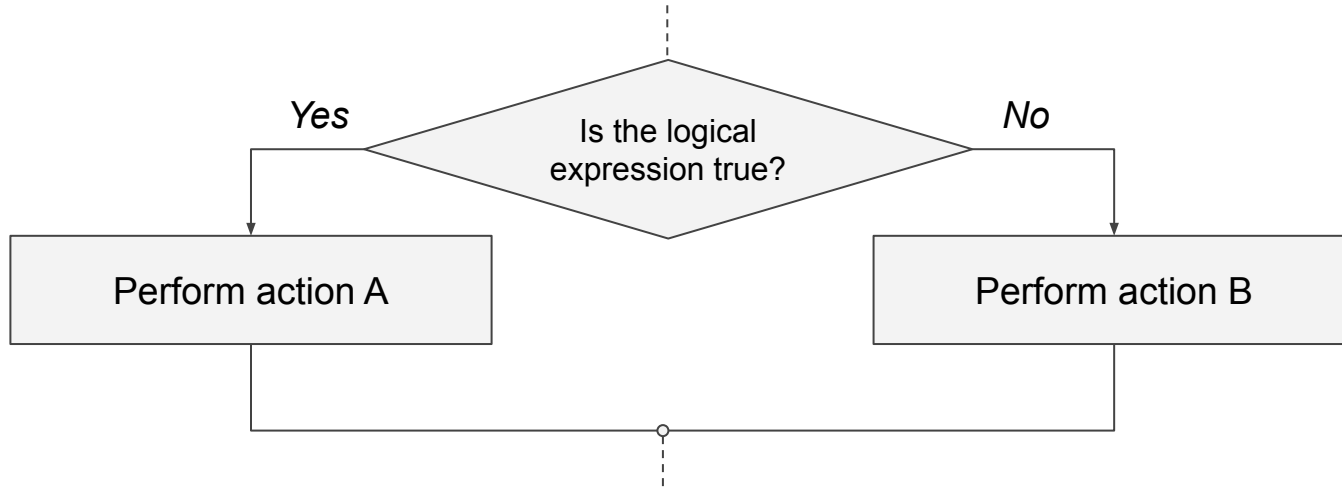


A conditional operator

is a command that performs or does not perform an action depending on the value of the logical expression.

Usage example:

performing action A if the expression is true or action B if it is false.



What types of conditional operators do you know? What words are used to set them?



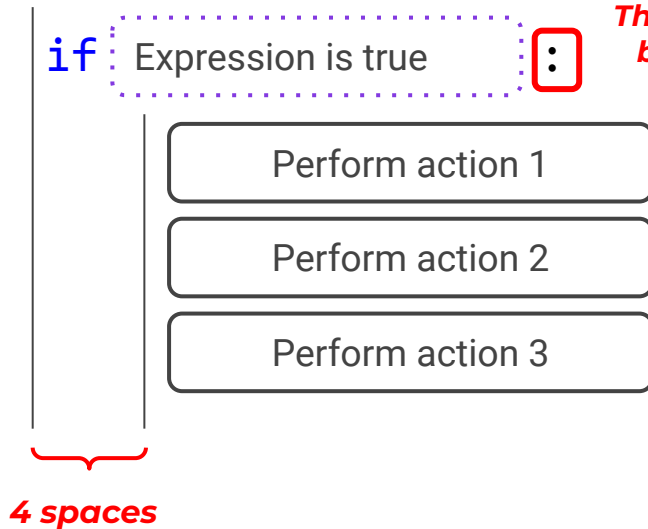
Qualifications



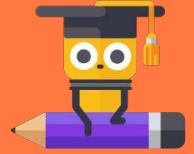
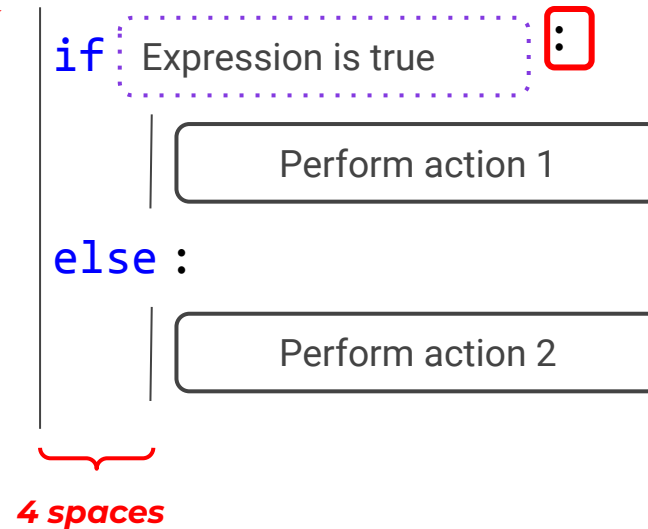
Types of conditional operators:

- basic conditional operator;
- nested conditional operator;
- conditional operator with multiple branches.

BASIC:



The action block begins with a colon



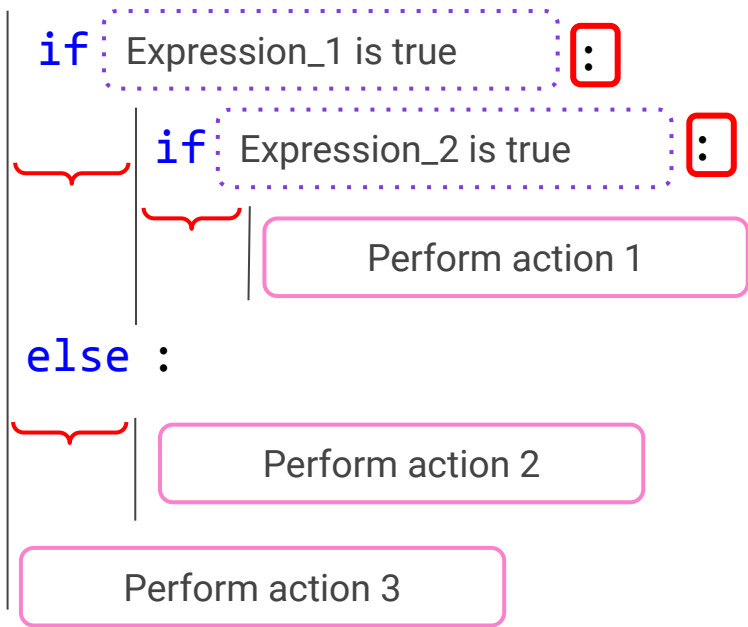
Qualifications



Types of conditional operators:

- basic conditional operator;
- nested conditional operator;
- conditional operator with multiple branches.

NESTED:



If Expression_1 is true,

And if Expression_2 is true,

Perform action_1

In all other cases,

Perform action 2.

(По окончании условного оператора)

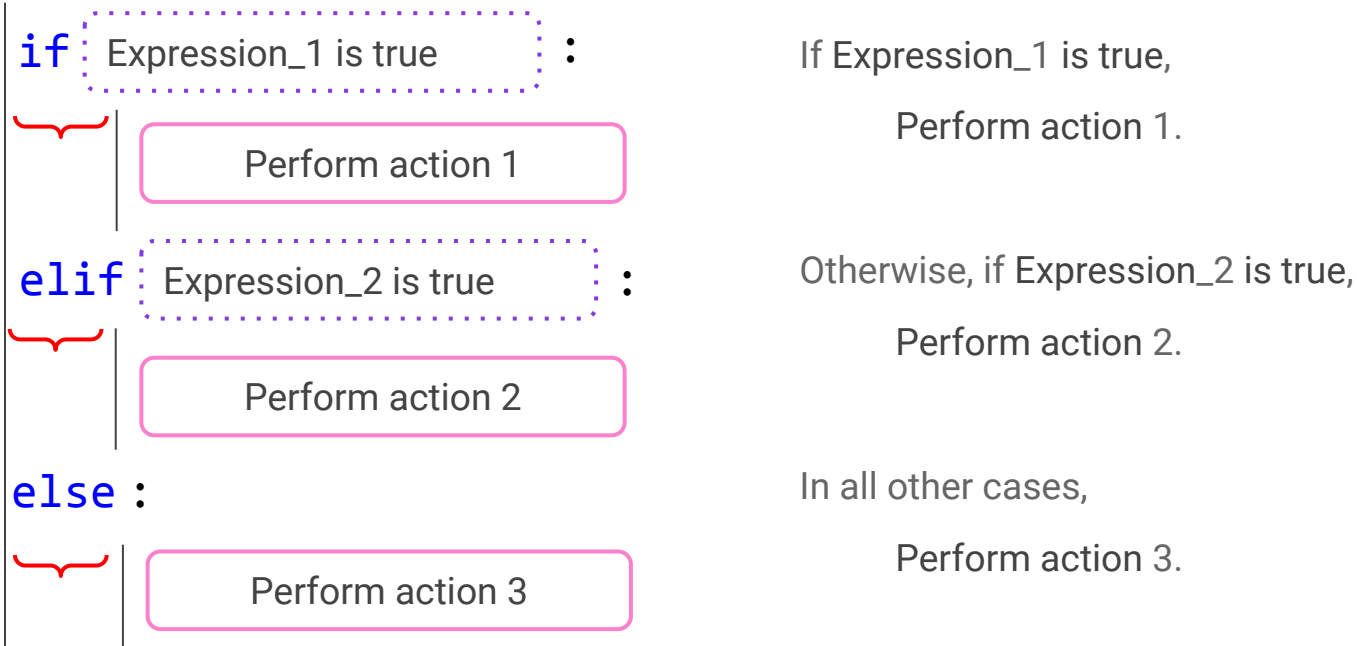
Perform action 3



Types of conditional operators:

- basic conditional operator;
- nested conditional operator;
- conditional operator with multiple branches.

OPERATOR WITH MULTIPLE BRANCHES:



What is a **loop?**
What is it used for?



Qualifications

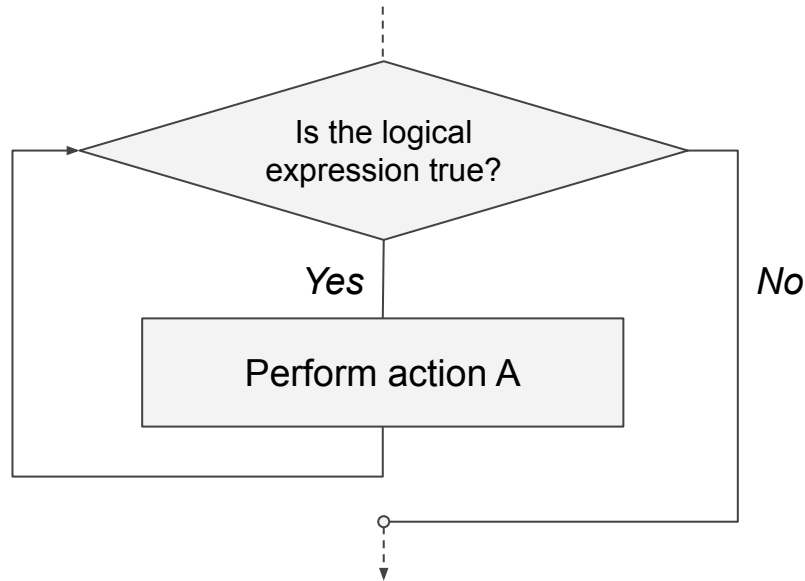


A loop

is a command that performs the indicated actions as long as the logical expression (condition) remains true.

Example:

The loop performs action A as long as the logical expression is true



What **types of loops do you know?**
What words are used to set them?



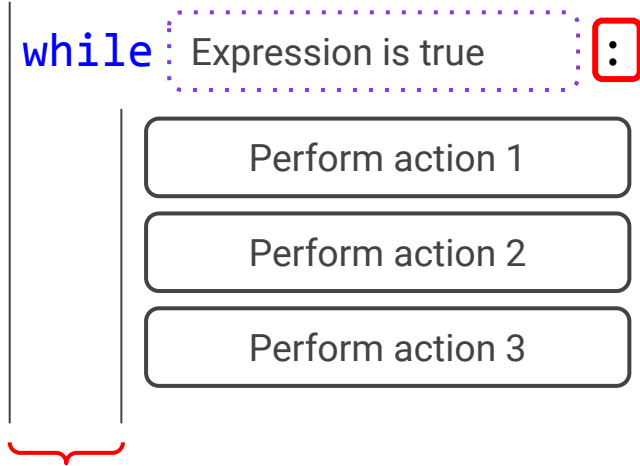
Qualifications



The **while** loop:

- basic loop;
- loop with a counter.

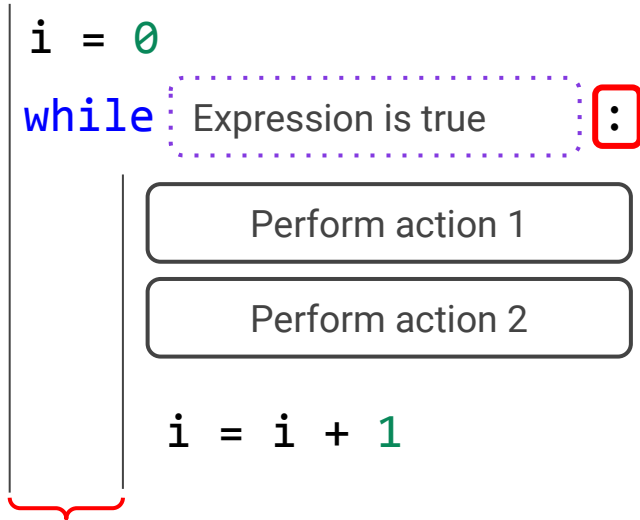
BASIC:



The **while** loop:

- basic loop;
- loop with a counter.

LOOP WITH A COUNTER:



**The counter can be used to count loop repeats.*

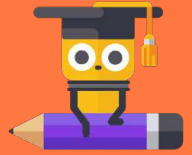
Create a counter variable and set the initial value to 0

While Expression is true,

Perform action 1,

Perform action 2.

Increase the value of the counter to 1



Qualifications



Qualifications confirmed!

Great, you are ready for brainstorming and completing your work tasks!

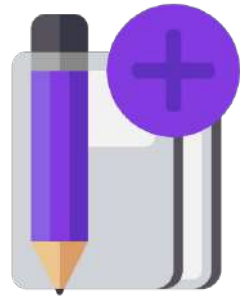


Qualifications



Brainstorming:

The “Five out of Five” private school



Request from the school's director

The director of the “Five out of Five” school contacted ProTeam.

Every quarter, their elementary school students undergo mandatory testing to assess their performance.

It's hard for teachers to check a large number of tests, so the school's leadership decided to order a set of programs from ProTeam for automatic testing of students.



Brainstorming



Planning work tasks

The students' knowledge is assessed using test questions.

The questions can be of different types, for example:

- choose the correct answer from four possible answers (multiple choice);
- free answer (no options).

Let's look at a few tasks sent by teachers.



Brainstorming



Planning work tasks

The task. Write a program that displays a question and answer options. Then the student is asked to enter a response and the result is displayed.

"Which tree is not coniferous: pine, cedar, spruce, maple".

How will this program work?



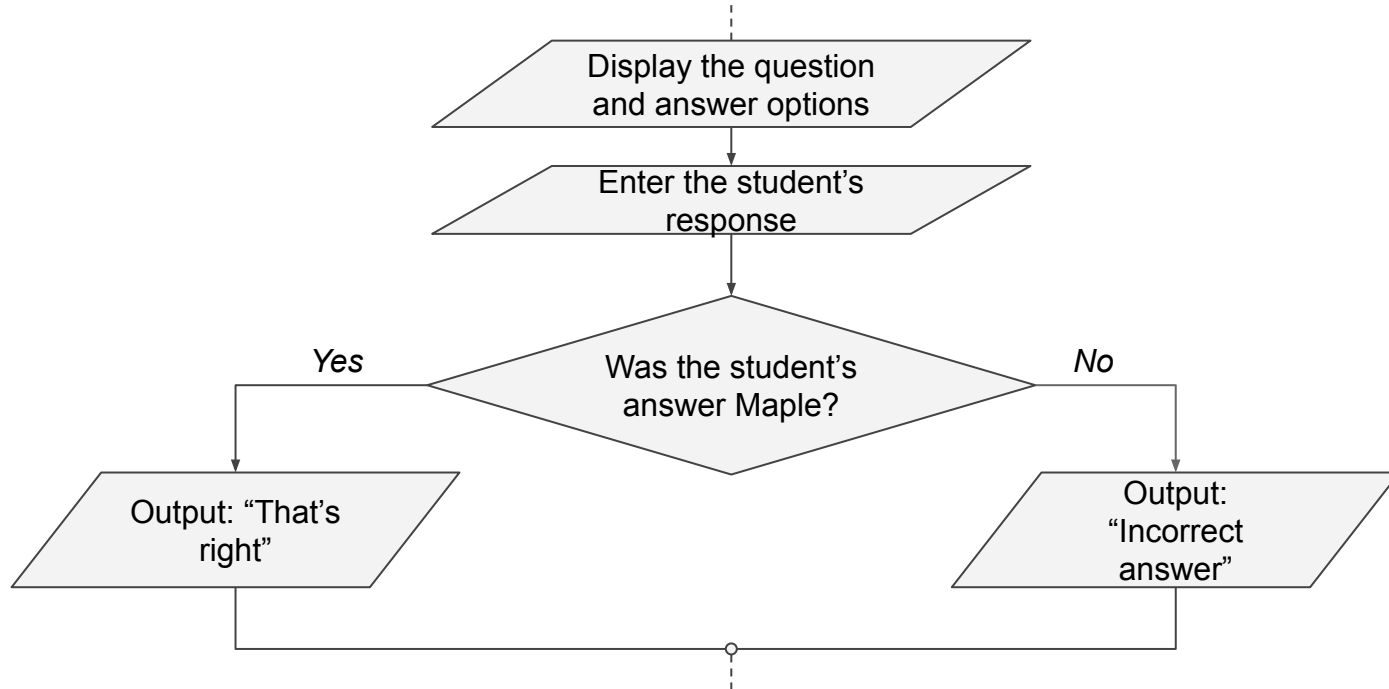
Brainstorming



Planning work tasks

The task. Write a program that displays a question and answer options. Then the student is asked to enter a response and the result is displayed.

"Which tree is not coniferous: pine, cedar, spruce, maple".



Brainstorming



Planning work tasks

The task. Write a program that displays a question and answer options. Then the student is asked to enter a response and the result is displayed.

"Which tree is not coniferous: pine, cedar, spruce, maple".

We will need the following:

- Data input-output — the `print()` and `input()` functions.
- To determine if the response is correct — the conditional operator `if` — `else`.



Brainstorming



Planning work tasks

The task. Write a program that displays a question and answer options. Then the student is asked to enter a response and the result is displayed.

"Which tree is not coniferous: pine, cedar, spruce, maple".

```
print('Which tree is not coniferous: pine, cedar, spruce, maple')
```

?



Brainstorming



Planning work tasks

The task. Write a program that displays a question and answer options. Then the student is asked to enter a response and the result is displayed.

"Which tree is not coniferous: pine, cedar, spruce, maple".

```
print('Which tree is not coniferous: pine, cedar, spruce, maple')
answer = input('Answer:')
if answer == 'maple':
    print('That's right!')
else:
    print('Incorrect answer')
```



```
Which tree is not coniferous: pine, cedar, spruce, maple
Answer:
>>> maple
That's right!
```



Brainstorming



Planning work tasks

The task. Write a program that asks a question with a free answer. If the answer is incorrect, the program displays "Try again!" and the question is asked again. If the answer is correct, the program displays "That's right!" and shuts down.

"A square is given with a side of 3. What is its perimeter?"

How will this program work?



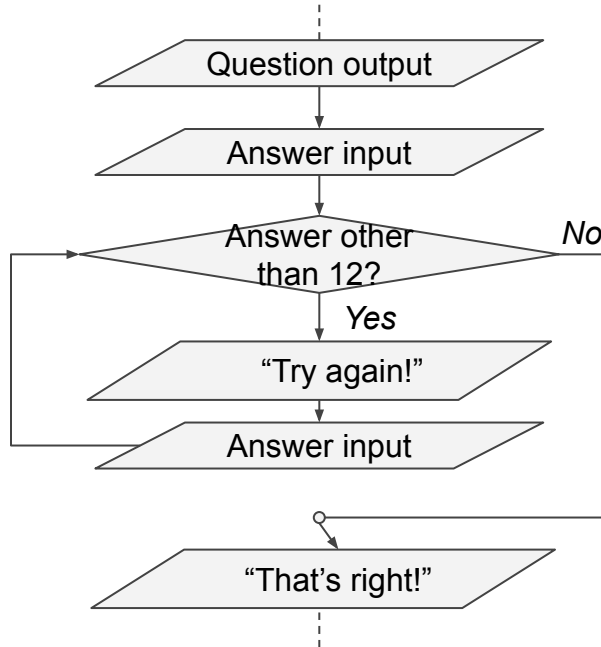
Brainstorming



Planning work tasks

The task. Write a program that asks a question with a free answer. If the answer is incorrect, the program displays "Try again!" and the question is asked again. If the answer is correct, the program displays "That's right!" and shuts down.

"A square is given with a side of 3. What is its perimeter?"



Brainstorming



Planning work tasks

The task. Write a program that asks a question with a free answer. If the answer is incorrect, the program displays "Try again!" and the question is asked again. If the answer is correct, the program displays "That's right!" and shuts down.

"A square is given with a side of 3. What is its perimeter?"

We will need the following:

- Data input-output — the `print()` and `input()` functions.
- To transition to the integer data type — the `int()` function.
- To repeat the steps until the correct answer is given — the `while` loop.



Brainstorming



Planning work tasks

The task. Write a program that asks a question with a free answer. If the answer is incorrect, the program displays "Try again!" and the question is asked again. If the answer is correct, the program displays "That's right!" and shuts down.

"A square is given with a side of 3. What is its perimeter?"

```
print('A square is given with a side of 3. What is its  
perimeter?')  
answer = int(input('Answer:'))  
while answer != 12:  
    answer = int(input('Try again:'))  
print('That's right!')
```



```
A square is given with a side of 3. What is its perimeter?  
Answer:  
>>> 6  
Try again:  
>>> 12  
That's right!
```



Brainstorming



Planning work tasks

The task. Write a program that asks a question with a free answer. If the answer is incorrect, the program displays "Try again!" and the question is asked again. If the answer is correct, the program displays "That's right!" and shuts down.

"A square is given with a side of 3. What is its perimeter?"

```
print('A square is given with a side of 3. What is its  
perimeter?')  
answer = int(input('Answer:'))  
while answer != 12:  
    answer = int(input('Try again:'))  
print('That's right!')
```

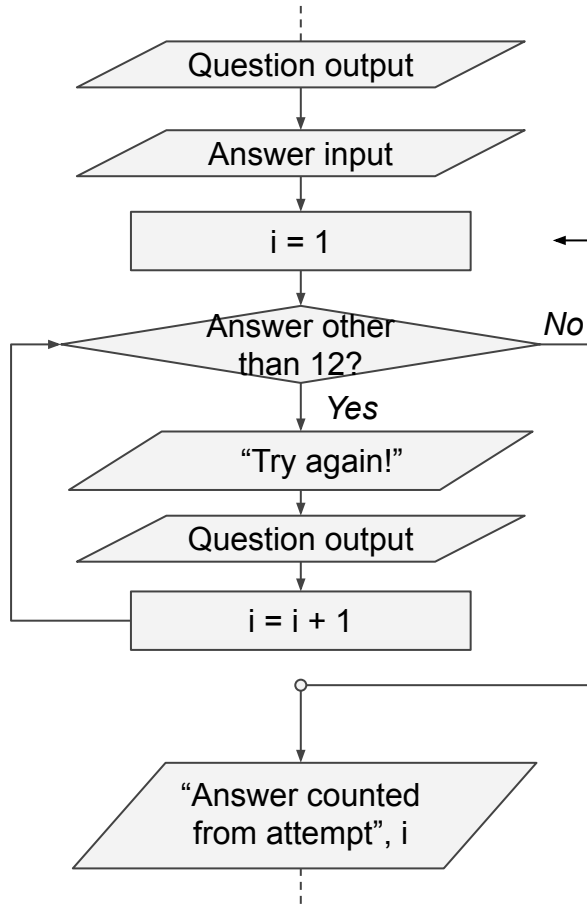
How can we modify the program so that it displays the number of attempts made?



Brainstorming



Possible modification:



Add a counter for the number of responses.



A square is given with a side of 3. What is its perimeter?

Answer:

```
>>> 3
```

Try again:

```
>>> 4
```

Try again:

```
>>> 12
```

Answer counted from 3 attempt



Brainstorming



Planning work tasks

The task. Write a program that asks a question with a free answer. If the answer is incorrect, the program displays "Try again!" and the question is asked again. If the answer is correct, the program displays "That's right!" and shuts down.

"A square is given with a side of 3. What is its perimeter?"

```
print('A square is given with a side of 3. What is its  
perimeter?')  
answer = int(input('Answer:'))  
i = 1  
while answer != 12:  
    answer = int(input('Try again:'))  
    i = i + 1  
print('Answer counted from', i, 'attempt')
```



Brainstorming



Before proceeding to the tasks:

1. What will the previous program display if the student enters: 12? 24?
2. How can we program improvement: if the student answers correctly on the first try, the program displays: “Five for paying attention!”
3. How can we program a question with three free answer attempts?
4. A four-answer-choice question is given. How should the program respond if the student enters an answer that does not match any of the options?



Brainstorming

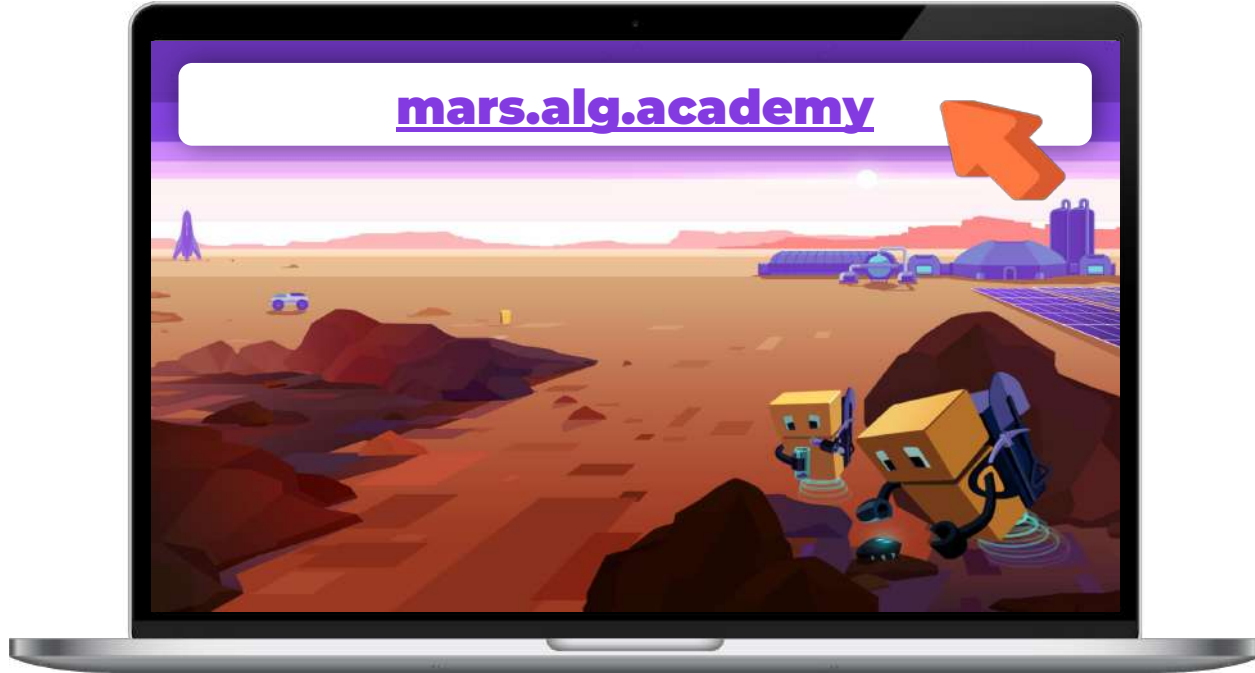


“Five out of Five”: Knowledge check



Complete tasks on the platform

➔ Five out of Five: check



Five out of Five:
Knowledge check



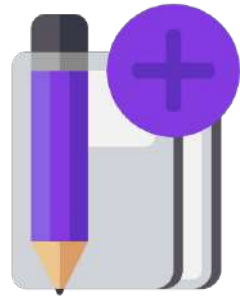
Break



Module 1. Lesson 1. Review. Exception processing

Five out of Five:

Exception processing



Let's look at the task

The task. Write a program that checks the formula for calculating the area of a rectangle. There is one answer attempt. The user enters the lengths of the two sides of a rectangle. The program should display its area.



Brainstorming



It seems like we have already solved a similar task. What will the program look like?

Let's look at the task

The task. Write a program that checks the formula for calculating the area of a rectangle. There is one answer attempt. The user enters the lengths of the two sides of a rectangle. The program should display its area.

Possible solution:

```
a = int(input('Enter the length of the first
side:'))
b = int(input('Enter the length of the second
side:'))
s = a * b
print('The area of a rectangle:', s)
```



```
Enter the length of the first side:
>>> 10
Enter the length of the second side:
>>> 5
The area of a rectangle: 50
```



Brainstorming



Let's look at the task

The task. Write a program that checks the formula for calculating the area of a rectangle. There is one answer attempt. The user enters the lengths of the two sides of a rectangle. The program should display its area.

Possible solution:

```
a = int(input('Enter the length of the first
side:'))
b = int(input('Enter the length of the second
side:'))
s = a * b
print('The area of a rectangle:', s)
```

Hmm, this program won't always work correctly.

In what cases?



Brainstorming



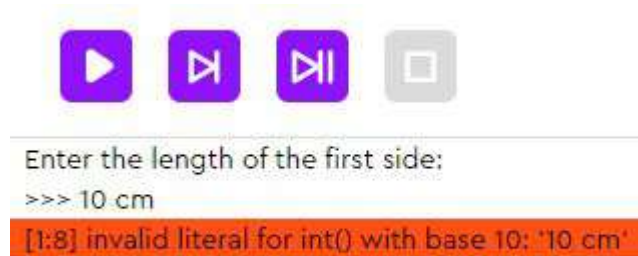
Let's look at the task

The task. Write a program that checks the formula for calculating the area of a rectangle. There is one answer attempt. The user enters the lengths of the two sides of a rectangle. The program should display its area.

Possible solution:

```
a = int(input('Enter the length of the first
side:'))
b = int(input('Enter the length of the second
side:'))
s = a * b
print('The area of the rectangle:', s)
```

A string with letters and numbers cannot be converted to an integer type. What can we do?




Brainstorming



We note the following:

- The last time the program ran, the Python **interpreter did not notice the error!**
- A user without programming skills **will not understand the cause** of this error and will not be able to use the program.



```
Enter the length of the first side:  
>>> 10 cm  
[1:8] invalid literal for int() with base 10: '10 cm'
```

We need a tool to detect unwanted program behavior and correctly notify the user of the cause of the error.



Brainstorming



Exception processing

is identifying undesirable program behavior and describing the program's reaction to that error.

Based on the previous task, what kind of errors can be detected and corrected by an exception processor?



Brainstorming



Exception processing

is identifying undesirable program behavior and describing the program's reaction to that error.

Types of programming errors:

- critical (detected by the interpreter and corrected by programmers);
- hidden (not visible to the interpreter and they hinder the user).



They can be recognized by the processor and a fix can be suggested.



Brainstorming



Exception processing

is identifying undesirable program behavior and describing the program's reaction to that error.

Getting potentially problematic **data**

try:

Attempt to perform an action with the **data**

except:

Action-reaction if an error occurs

Try to perform:

Action

If it doesn't work out:

Action-reaction



Brainstorming



Let's modify the task

The task. Write a program that checks the formula for calculating the area of a rectangle. There is one answer attempt. The user enters the lengths of the two sides of a rectangle. The program should display its area.

```
a = int(input('Enter the length of the first side:'))
b = int(input('Enter the length of the second
side:'))
s = a * b
print('The area of the rectangle:', s)
```

How can we improve the program with exception processing?



Brainstorming



Let's modify the task

```
a = input('Enter the length of the first side:')
try:
    a = int(a)
except:
    print('Error! Enter the length as an integer without
letters')
    a = int(input('Enter the length of the first side:'))
b = input('Enter the length of the second side:')
try:
    b = int(b)
except:
    print('Error! Enter the length as an integer without
letters')
    b = int(input('Enter the length of the second side:'))
s = a * b
print('The area of the rectangle:', s)
```

How will this program react if the user enters "10 cm"?



Brainstorming



Let's modify the task

```
a = input('Enter the length of the first side:')
try:
    a = int(a)
except:
    print('Error! Enter the length as an integer without
letters')
    a = int(input('Enter the length of the first side:'))
b = input('Enter the length of the second side:')
try:
    b = int(b)
except:
    print('Error! Enter the length as an integer without
letters')
    b = int(input('Enter the length of the second side:'))
s = a * b
print('The area of the rectangle:', s)
```



```
Enter the length of the first side:
>>> 10 cm
Error! Enter the length as an integer without letters
Enter the length of the first side:
>>> 10
Enter the length of the second side:
>>> 5
The area of the rectangle: 50
```



Brainstorming



Let's modify the task

```
a = input('Enter the length of the first side:')
try:
    a = int(a)
except:
    print('Error! Enter the length as an integer without
letters')
    a = int(input('Enter the length of the first side:'))
b = input('Enter the length of the second side:')
try:
    b = int(b)
except:
    print('Error! Enter the length as an integer without
letters')
    b = int(input('Enter the length of the second side:'))
s = a * b
print('The area of the rectangle:', s)
```



```
Enter the length of the first side:
>>> 10 cm
Error! Enter the length as an integer without letters
Enter the length of the first side:
>>> 10
Enter the length of the second side:
>>> 5
The area of the rectangle: 50
```

How will the program react if the user doesn't pay attention to the notification and re-enters "10 cm"?



Brainstorming



Let's modify the task

```
a = input('Enter the length of the first side:')
try:
    a = int(a)
except:
    print('Error! Enter the length as an integer without
letters')
    a = int(input('Enter the length of the first side:'))
b = input('Enter the length of the second side:')
try:
    b = int(b)
except:
    print('Error! Enter the length as an integer without
letters')
    b = int(input('Enter the length of the second side:'))
s = a * b
print('The area of the rectangle:', s)
```



```
Enter the length of the first side:
>>> 10 cm
Error! Enter the length as an integer without letters
Enter the length of the first side:
>>> 10 cm
[6:11] invalid literal for int() with base 10: '10 cm'
```

Error again!
How can we anticipate that too?



Brainstorming



Let's modify the task

```
while True:  
    a = input('Enter the length of the first side:')  
    try:  
        a = int(a)  
        break  
    except:  
        print('Error! Enter the length as an integer without letters')
```

```
while True:  
    b = input('Enter the length of the second side:')  
    try:  
        b = int(b)  
        break  
    except:  
        print('Error! Enter the length as an integer without letters')
```

```
s = a * b  
print('The area of the rectangle:', s)
```

We make the input of the length endless until it is received in the desired format.



Brainstorming



Let's modify the task

```
while True:  
    a = input('Enter the length of the first side:')  
    try:  
        a = int(a)  
        break  
    except:  
        print('Error! Enter the length as an integer without letters')
```

```
while True:  
    b = input('Enter the length of the second side:')  
    try:  
        b = int(b)  
        break  
    except:  
        print('Error! Enter the length as an integer without letters')
```

```
s = a * b
```

```
print('The area of the rectangle:', s)
```



```
Enter the length of the first side:  
>>> 10 cm  
Error! Enter the length as an integer without letters  
Enter the length of the first side:  
>>> 10 cm  
Error! Enter the length as an integer without letters  
Enter the length of the first side:  
>>> 10  
Enter the length of the second side:  
>>> 5 cm  
Error! Enter the length as an integer without letters  
Enter the length of the second side:  
>>> 5  
The area of the rectangle: 50
```



Brainstorming



Conclusions:

1. **An exception processor** detects undesirable program behavior and describes the reaction to that error.
2. To program exception processing, we use the **try** and **except operators**.
3. If necessary, exception processing can be put into a loop.



Brainstorming



“Five out of Five: Check 2



Complete tasks on the platform

➡ Five out of Five: check 2



Sunflowers:
Feedback

End of the work day



Let's end the work day by answering these technical questions

1. What algorithmic constructs do you know? What operators are they set by?
2. What is exception processing? What is it used for?



*Cole,
senior developer*



*Alena,
project manager*



Summing up
the work day

Congratulations on completing the work day!

Today you did the following:

- 1) went through skills qualifications;
- 2) programmed a knowledge assessment system;
- 3) learned and implemented an error detection tool.



Summing up
the work day

Evaluating your performance

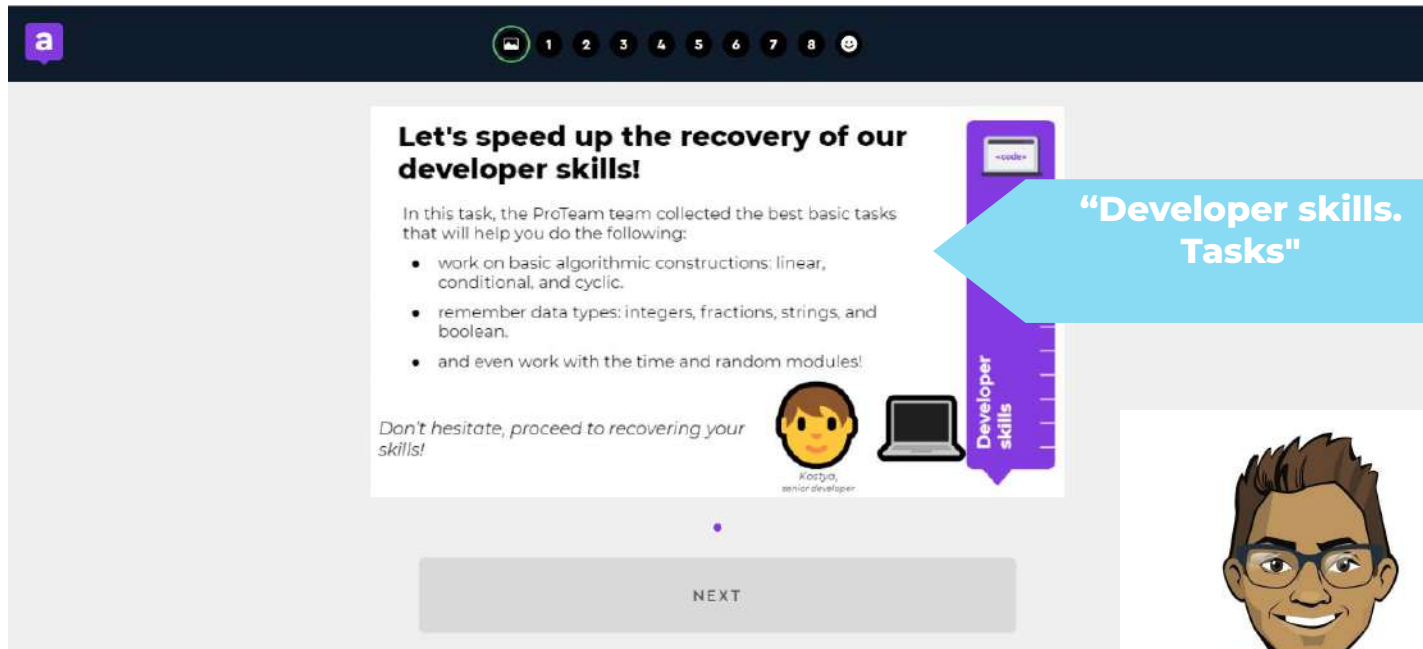
Answer these questions with your colleagues:

1. What turned out the best?
2. What didn't turn out the way you wanted?
3. What should you do next time to avoid any issues?



Summing up
the work day

Additional tasks to improve your performance



The screenshot shows a task interface with a dark header containing a purple 'a' icon and a progress bar with steps 1 through 8. The main content area has a white background with the following text:

Let's speed up the recovery of our developer skills!

In this task, the ProTeam team collected the best basic tasks that will help you do the following:

- work on basic algorithmic constructions: linear, conditional, and cyclic.
- remember data types: integers, fractions, strings, and boolean.
- and even work with the time and random modules!

Don't hesitate, proceed to recovering your skills!

Below the text is a profile icon for 'ProTigo, senior developer' and a laptop icon. A purple badge on the right side of the content area says 'Developer skills' and has a '-code-' icon at the top. A light blue arrow points from the badge area towards the right.

At the bottom of the content area is a grey button labeled 'NEXT'.

“Developer skills. Tasks”



Cole,
senior developer



Summing up
the work day