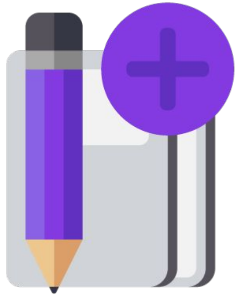
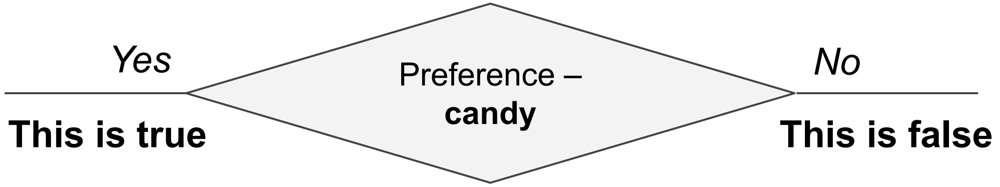
Brainstorm:

Logical data type



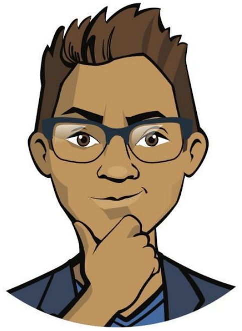
How do we program a condition?

In the previous task, we considered a **condition** as some kind of statement that can be **either true or false.**



`True`

`False`



Cole,
Senior Developer

Brainstorm

Logical data type

Such statements play an important role in programming. A **logical (boolean) data type** was invented for them.

Data type	*Integer*	*Logical*
Values	`-100, 5, 512`	`True, False`
Variables	`days = 31`	`is_correct = True`

Simple expressions	`daily_money * days`
	`price - sale`
		`5 > 2`
		`name != 'John'`

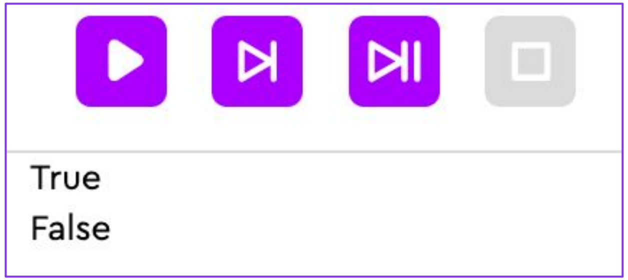
Brainstorm

Variables and simple expressions

Variables and expressions can take values

True or False.

```
checked = True
is_sent = False
print(checked)
print(is_sent)
```



```
True
False
```

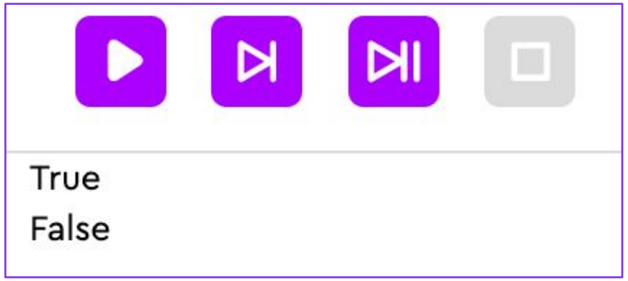
Brainstorm

Variables and simple expressions

Variables and expressions can take values:

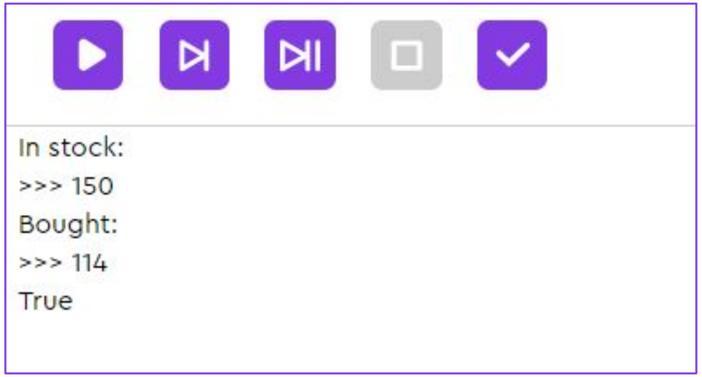
`True` or `False`.

```python
checked = True
is_sent = False
print(checked)
print(is_sent)
```



```
True
False
```

```python
amount_shop = int(input('In stock:'))
booked = int(input('Bought:'))
ok = amount_shop > booked
print(ok)
```



```
In stock:
>>> 150
Bought:
>>> 114
True
```

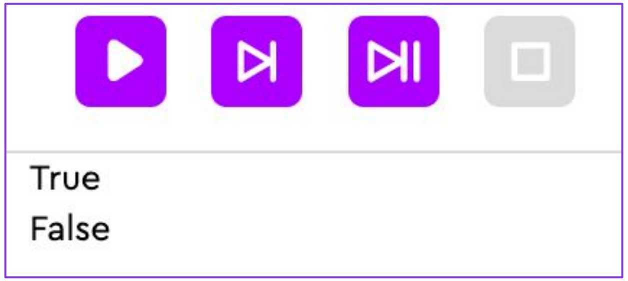
Brainstorm

Variables and simple expressions

Variables and expressions can take values:

True or False.

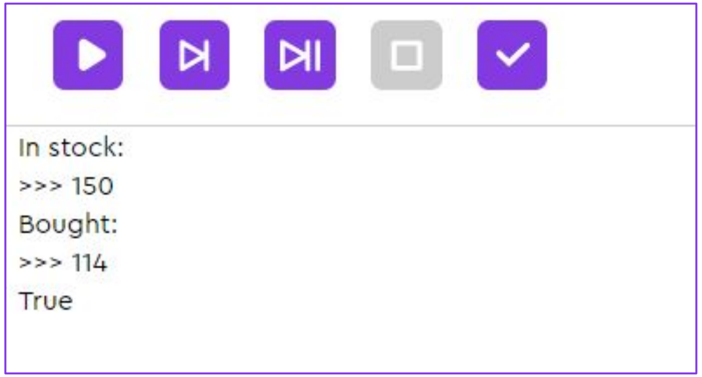
```
checked = True
is_sent = False
print(checked)
print(is_sent)
```



```
True
False
```

```
amount_shop = int(input('In stock:'))
booked = int(input('Bought:'))
ok = amount_shop > booked
print(ok)
```

Logical operation **Logical expression**



```
In stock:
>>> 150
Bought:
>>> 114
True
```

Brainstorm

Simple logical expression: comparison operators

Comparison operators can be used to make logical expressions.

Integer type					
*	/	%	//	+	-
Multiplication	Division	Remainder	Quotient	Addition	Subtraction

Brainstorm

Simple logical expression: comparison operators

Comparison operators can be used to make logical expressions.

Integer type					
*	/	%	//	+	-
Multiplication	Division	Remainder	Quotient	Addition	Subtraction

Logical type					
>	<	==	!=	<=	>=
Greater than	Less than	Equal	Not equal	Less than or equal	Greater than or equal

Brainstorm

Simple logical expression: comparison operators

Task. Write a program that asks for the stock balance of chocolates and determines if the stock needs to be replenished. The minimum amount of sweets in stock is 50 kg.

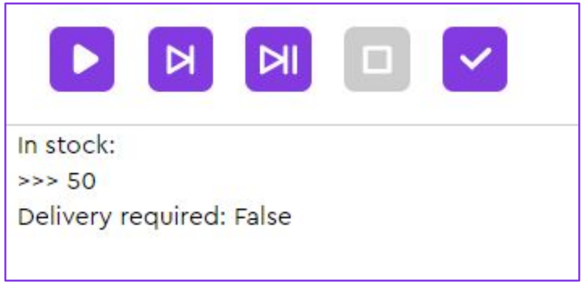
You might want to set up the delivery requirement using a logical expression.

Brainstorm

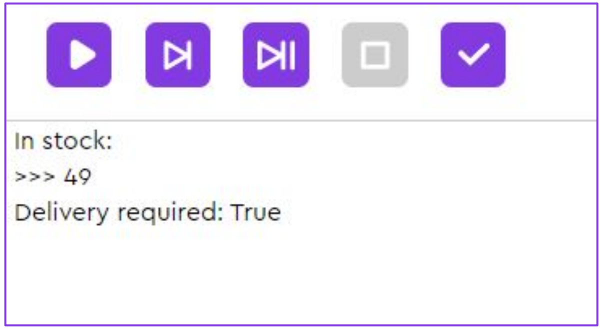
Simple logical expression: comparison operators

Task. Write a program that asks for the stock balance of chocolates and determines if the stock needs to be replenished. The minimum amount of sweets in stock is 50 kg.

```python
amount_store = int(input('In stock:'))

amount_min = 50

delivery = amount_store < amount_min

print('Delivery required:', delivery)
```



```
In stock:
>>> 50
Delivery required: False
```



```
In stock:
>>> 49
Delivery required: True
```

Brainstorm

Compound logical expression

A **compound** logical expression can be made up of **simple expressions** by linking them using logical operators:

Operator	*Name*	*Used when needed:*
and	Logical AND	Require two simple conditions to be met at the same time
or	Logical OR	Require at least one of two simple conditions to be met

order of execution

Brainstorm

**Subexpressions connected by logical AND are executed first, then those linked by logical OR.*

Compound logical expression

Task. Write a program that notifies the user about an error in the stock of chocolates.

A stock error occurs when the storage is almost empty (less than 50 kg) or when it is full (more than 300 kg).

Try to program a stock error using a compound logical expression.

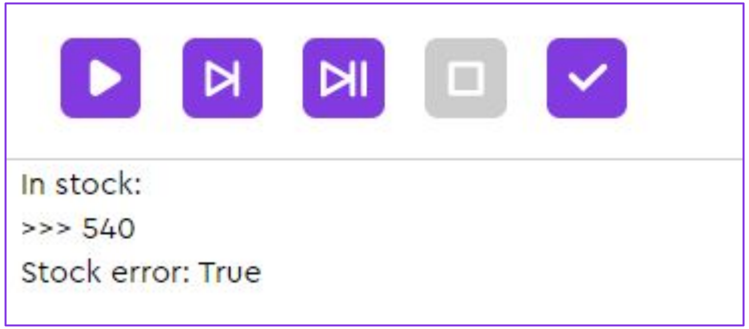
Brainstorm

Compound logical expression

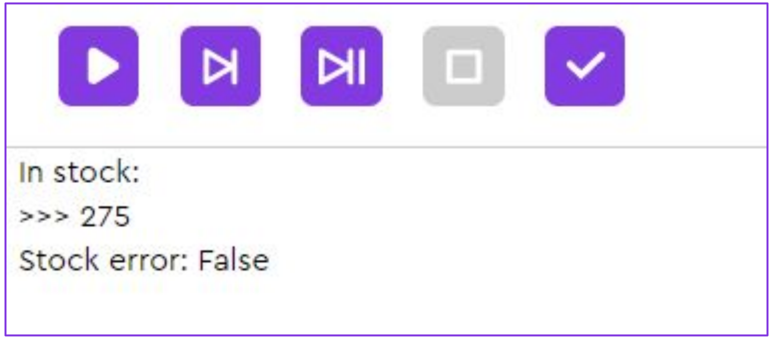
Task. Write a program that notifies the user about an error in the stock of chocolates.

A stock error occurs when the storage is almost empty (less than 50 kg) or when it is full (more than 300 kg).

```python
amount_store = int(input('In stock:'))

error = amount_store < 50 or amount_store > 300

print('Stock error:', error)
```



```
In stock:
>>> 540
Stock error: True
```



```
In stock:
>>> 275
Stock error: False
```

Brainstorm

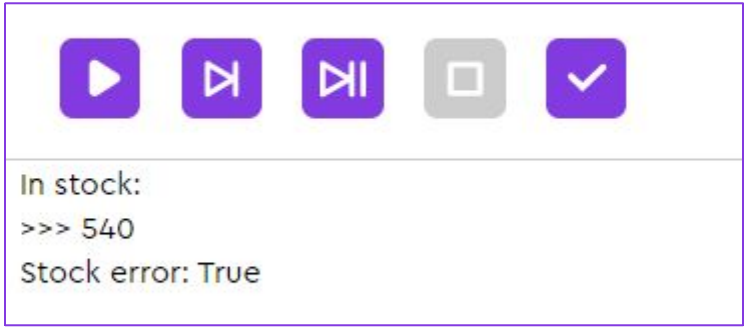
Compound logical expression

Task. Write a program that notifies the user about an error in the stock of chocolates.

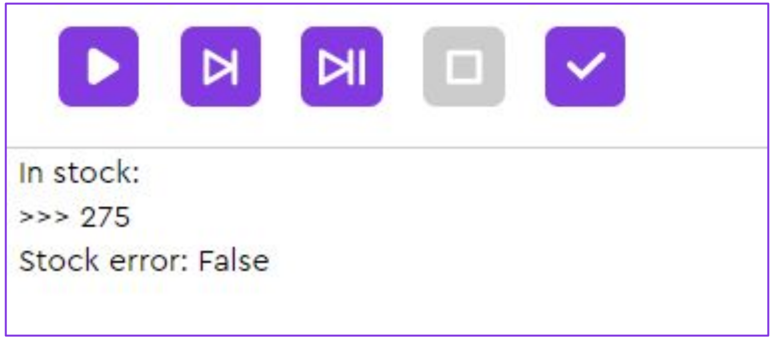
A stock error occurs when the storage is almost empty (less than 50 kg) or when it is full (more than 300 kg).

```python
amount_store = int(input('In stock:'))
error = amount_store < 50 or amount_store > 300
print('Stock error:', error)
```

← The values of simple expressions are calculated first. Then, they are followed by the values of compound expressions.



```
In stock:
>>> 540
Stock error: True
```

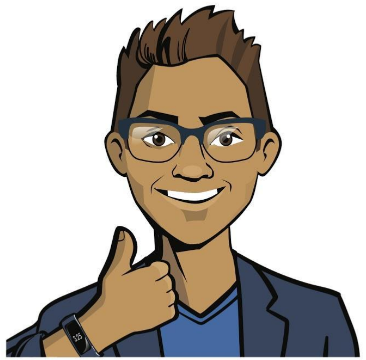


```
In stock:
>>> 275
Stock error: False
```

Brainstorm

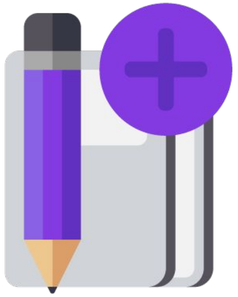
Conclusions:

1. A **logical data type** is a data type used to program statements that can be true or false.
2. **Simple logical expressions** can be constructed using <u>comparison operators</u>.
3. **Compound logical expressions** can be constructed from simple logical expressions and <u>logical operators</u>.



Brainstorm

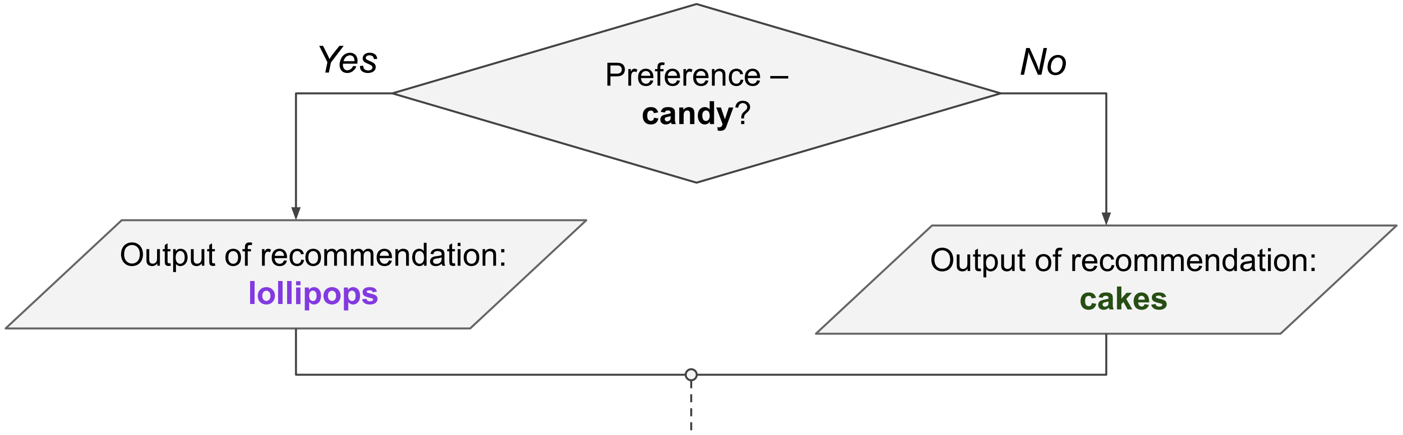
conditional statements



How do we program a selection?

We learned how to program a condition – a statement that can be true or false.

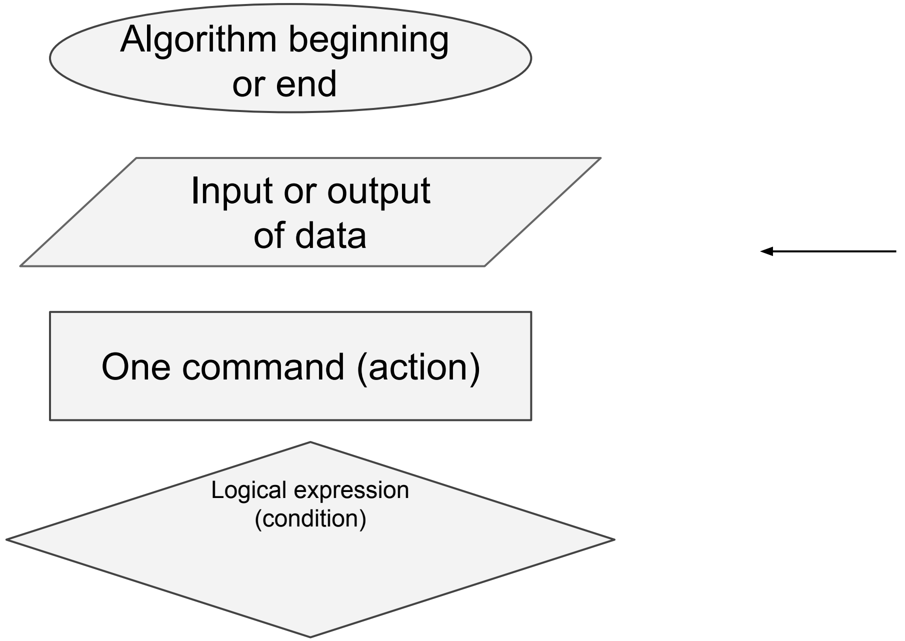
Now, let's become familiar with a construct that selects a command to execute depending on whether the condition is true.


Yes
Preference – **candy**?
No
Output of recommendation: **lollipops**
Output of recommendation: **cakes**

Writing an algorithm as a flowchart

From now on, when we analyze algorithmic constructs, we will use flowcharts.

This is a universal method of writing an algorithm that every programmer knows.



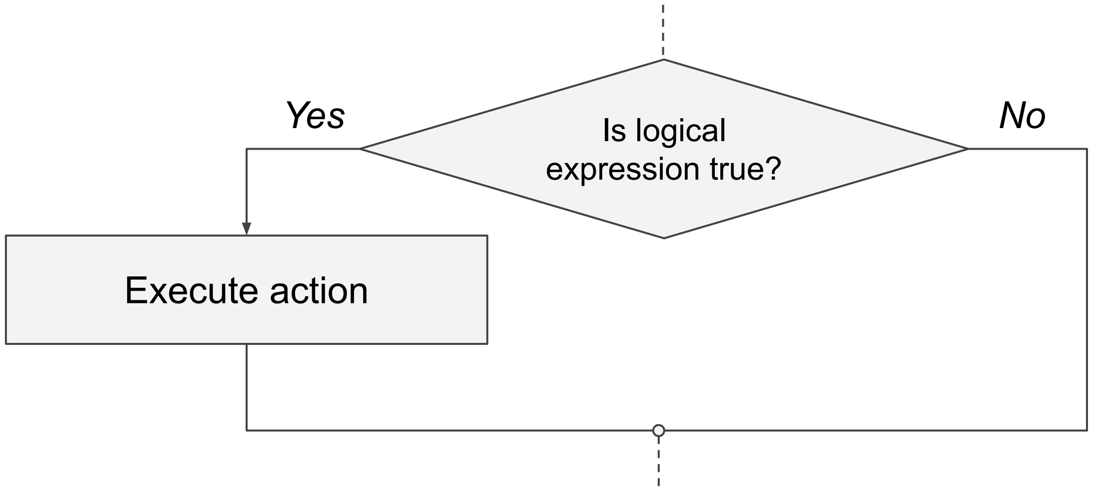
Block types

Brainstorm

A conditional statement

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

Usage example:
executing some action only if the expression is true.

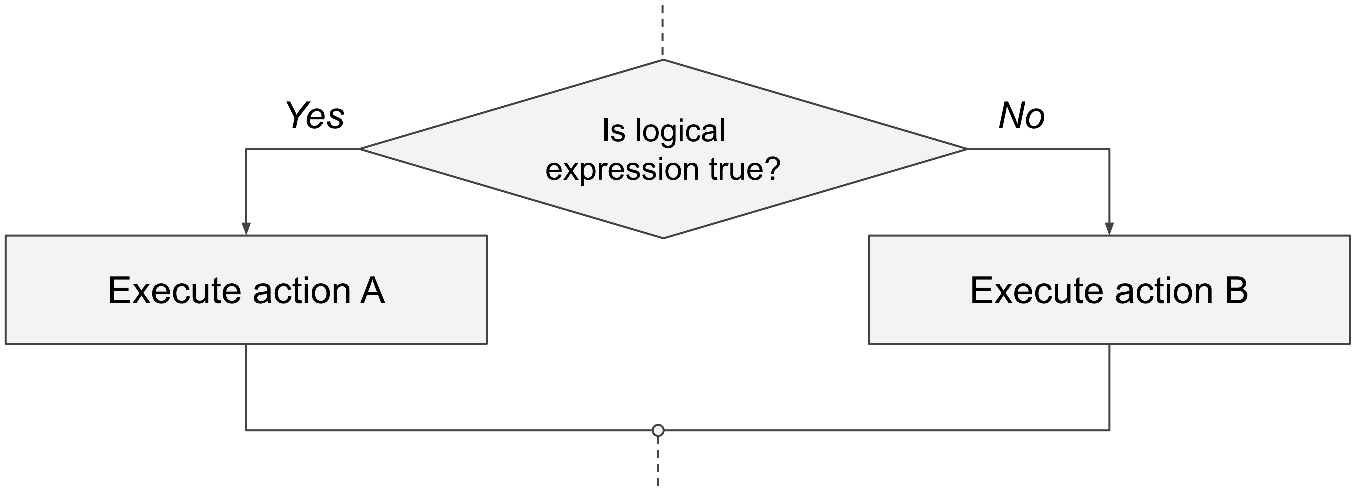


A conditional statement,

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

Usage example:

executing action A if the expression is true and action B is false.



Brainstorm

Conditional statement

Task 1. Create an algorithm that checks if it is possible to make a purchase with a card.

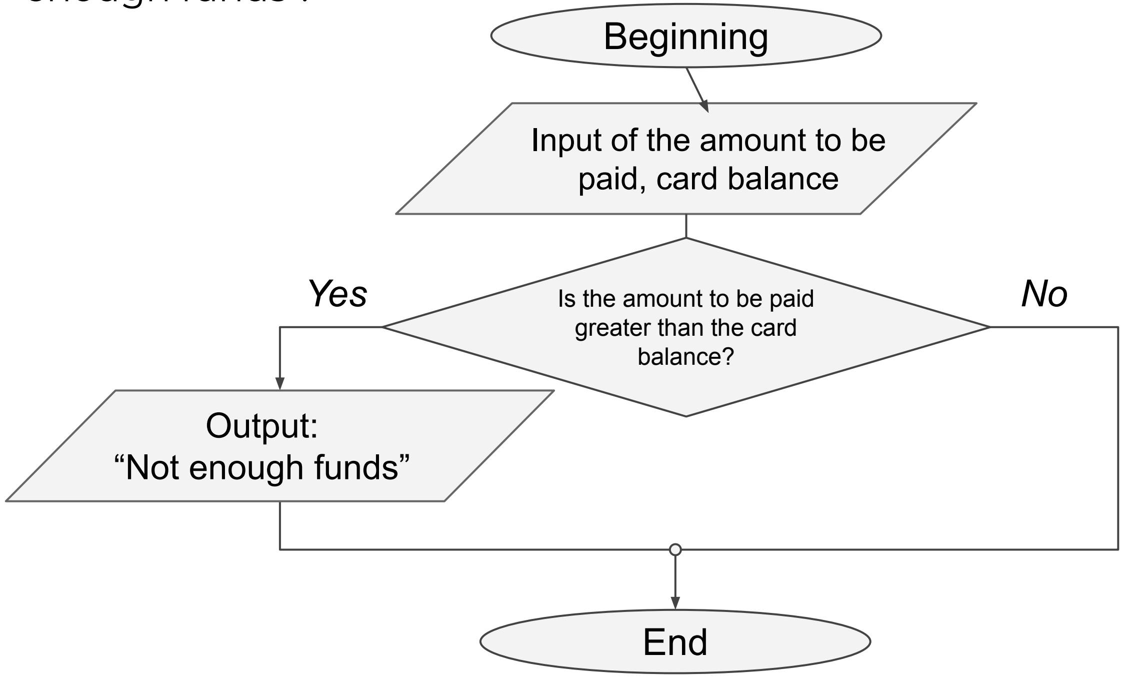
If the cost of goods is more than the amount of money on the card, output: “Not enough funds”.

Brainstorm

Conditional statement

Task 1. Create an algorithm that checks if it is possible to make a purchase with a card.

If the cost of goods is more than the amount of money on the card, output: “Not enough funds”.


Beginning
Input of the amount to be paid, card balance
Is the amount to be paid greater than the card balance?
Yes
No
Output: “Not enough funds”
End

Brainstorm

Conditional statement

Task 2. Create an algorithm that checks if it is possible to make a purchase with a card.

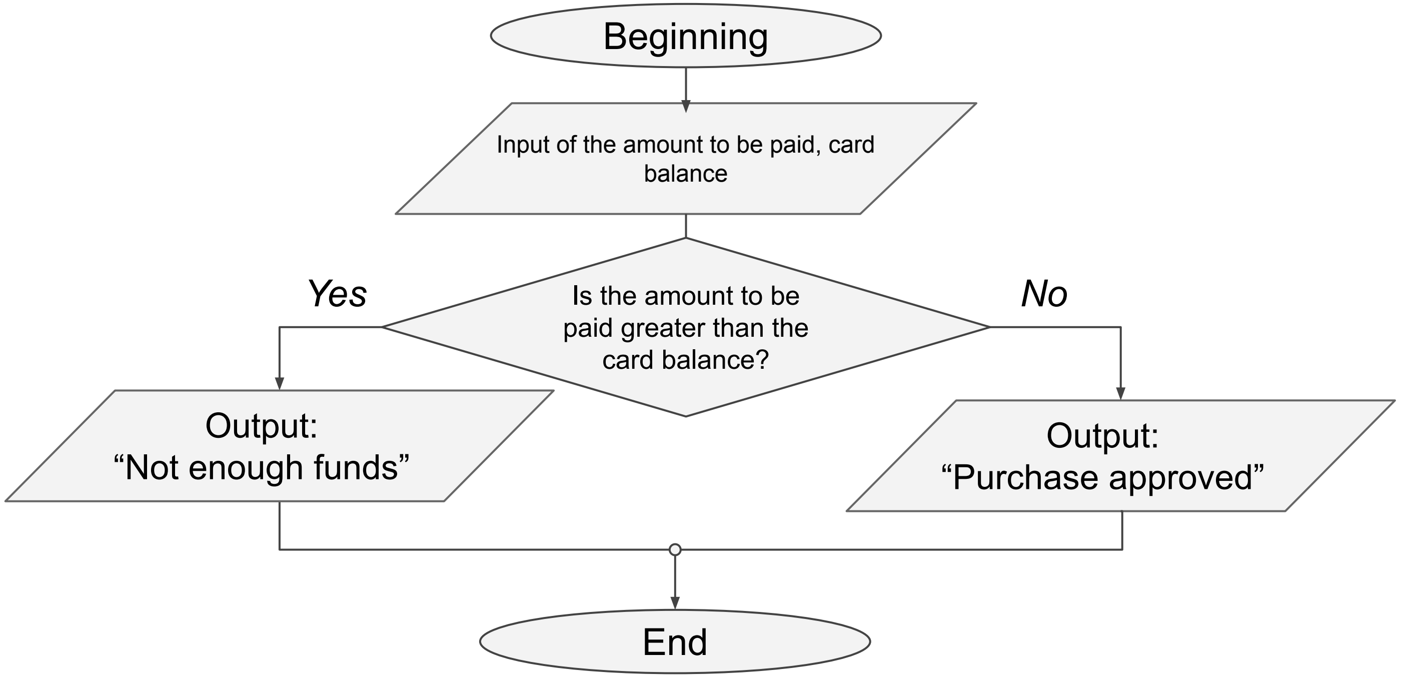
If the cost of goods is more than the amount of money on the card, output: “Not enough funds”. Otherwise, output “Purchase approved”

Brainstorm

Conditional statement

Task 2. Create an algorithm that checks if it is possible to make a purchase with a card.

If the cost of goods is more than the amount of money on the card, output: “Not enough funds”. Otherwise, output “Purchase approved”


Beginning
Input of the amount to be paid, card balance
Is the amount to be paid greater than the card balance?
Yes
No
Output: “Not enough funds”
Output: “Purchase approved”
End

Brainstorm

Conditional statement

To program a conditional statement, the following commands are used:

```
if

else
```

Brainstorm

Conditional statement

To program a conditional statement, the following commands are used:

```
if
else
```

```
if Expression is true :
    Execute action 1
    Execute action 2
    Execute action 3
```

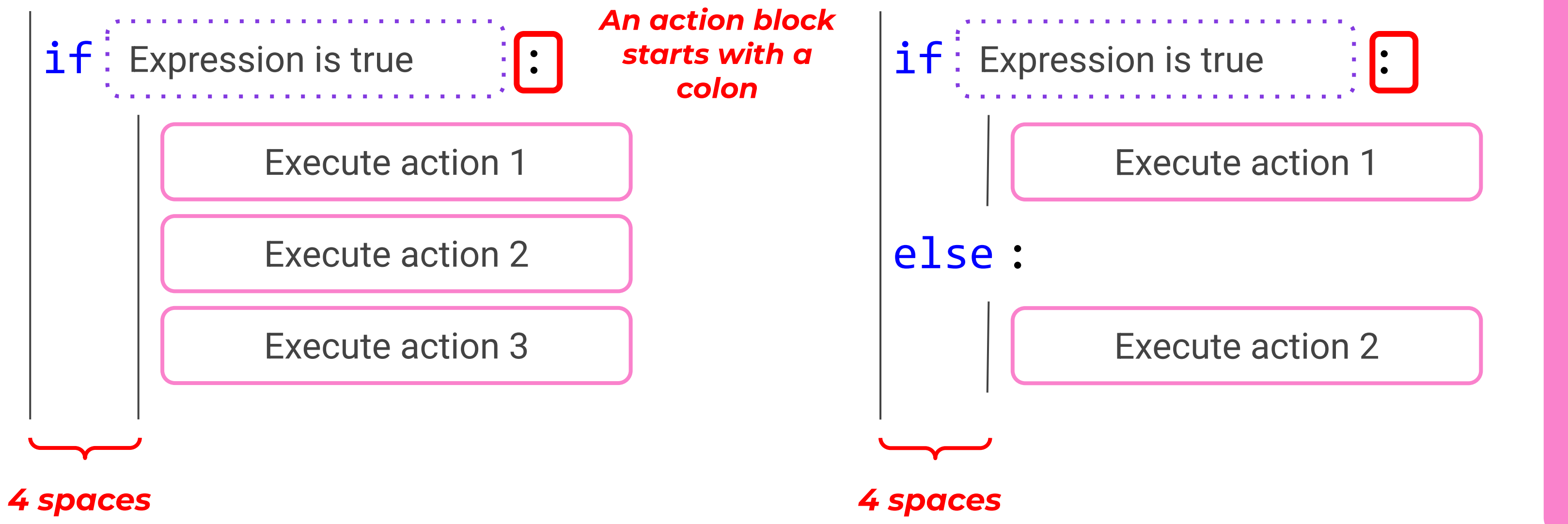
```
if Expression is true :
    Execute action 1
else :
    Execute action 2
```

Brainstorm

Conditional statement

To program a conditional statement, the following commands are used:

```
if
else
```



```
if Expression is true :
    Execute action 1
    Execute action 2
    Execute action 3
```

```
if Expression is true :
    Execute action 1
else :
    Execute action 2
```

Brainstorm

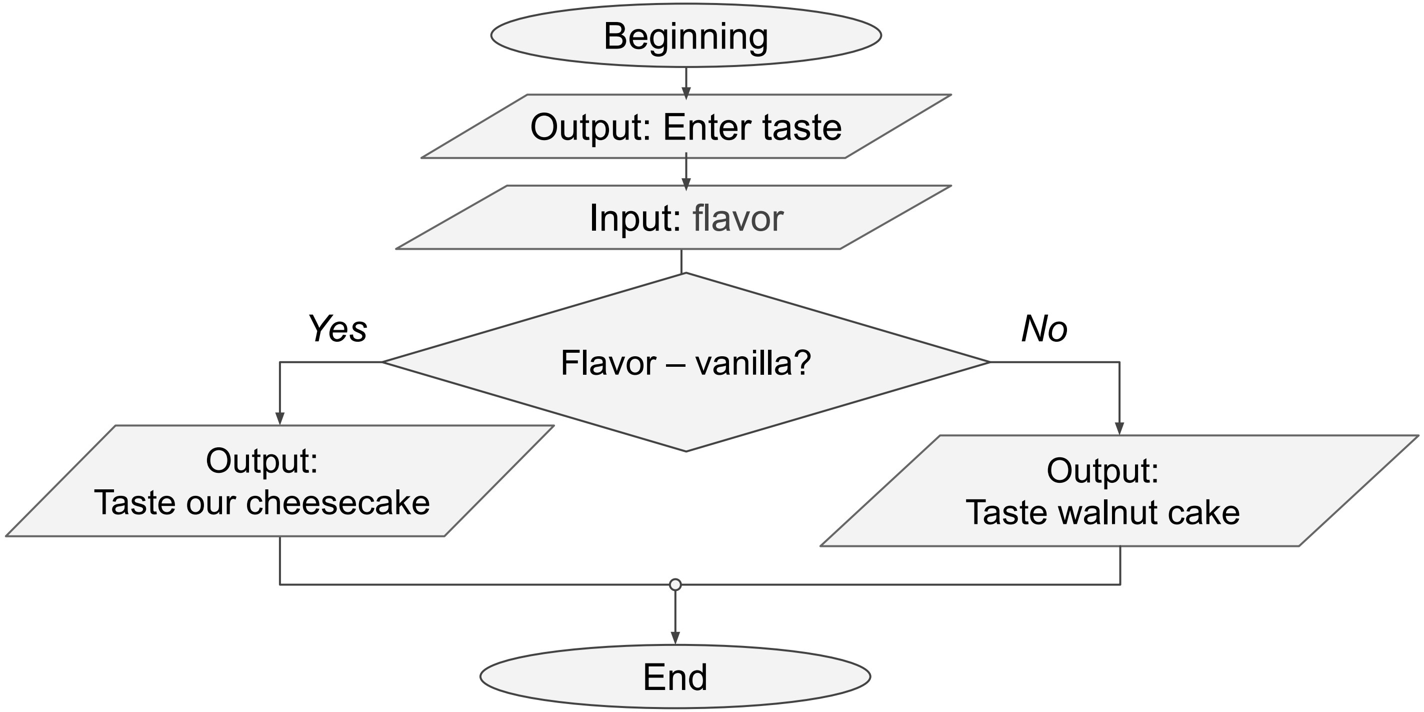
Conditional statement

Task. Write a program that offers a product according to taste preferences. The program asks what flavor the user likes. If it’s vanilla, recommend cheesecake. Else – walnut cake.

Brainstorm

Conditional statement

Task. Write a program that offers a product according to taste preferences. The program asks what flavor the user likes. If it's vanilla, recommend cheesecake. Else – walnut cake.



Brainstorm

Conditional statement

Task. Write a program that offers a product according to taste preferences. The program asks what flavor the user likes. If it's vanilla, recommend cheesecake. Else – walnut cake.

```
taste = input('Enter your favorite taste:')

taste = taste.lower()
```

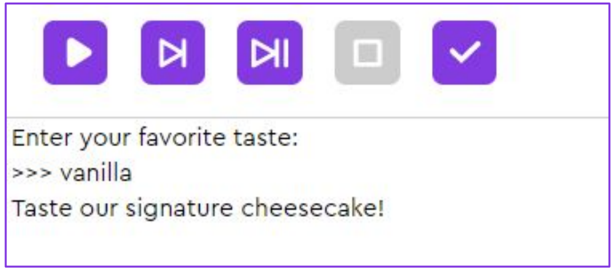
```
?
```

Brainstorm

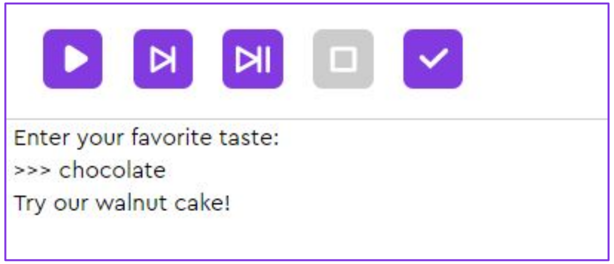
Conditional statement

Task. Write a program that offers a product according to taste preferences. The program asks what flavor the user likes. If it's vanilla, recommend cheesecake. Else – walnut cake.

```python
taste = input('Enter your favorite taste:')

taste = taste.lower()

if taste == 'vanilla':

    print('Taste our signature cheesecake!')

else:

    print('Try our walnut cake!')
```



```
Enter your favorite taste:
>>> vanilla
Taste our signature cheesecake!
```

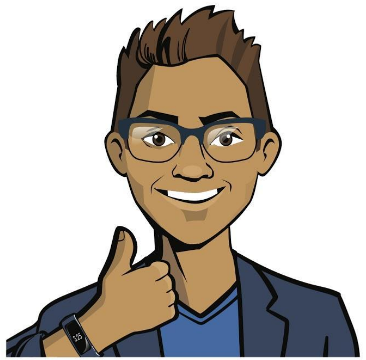


```
Enter your favorite taste:
>>> chocolate
Try our walnut cake!
```

Brainstorm

Conclusions:

1. A conditional statement is a command that executes or does not execute an action depending on the value of a logical expression.
2. To program a conditional statement, the `if` `and` `else` statements are used.
3. Actions within a conditional statement begin with a colon and are indented by 4 spaces.



Brainstorm