UNIVERSITY OF TWENTE.

Conditionals

The key tools to making decisions inside the code

1

Today's objectives

After this lecture, students will be able to:

- explain the use of conditionals in a program
- apply conditionals to control the program flow

UNIVERSITY OF TWENTE.

2

Contents

- logical operators, comparison operators
- modulo
- conditional execution

UNIVERSITY OF TWENTE.

3

Booleans

In Python, Boolean is a primitive data type with just two values: *True* and *False*

Such values are often used for controlling the program flow

UNIVERSITY OF TWENTE.

4

Equality, inequality

Equality operator ==

Inequality operator !=

Domains: just about anything!

Range: Boolean

print(1 == 1)

True

print('Hello' != 'World')

True

UNIVERSITY OF TWENTE.

6

Modulo: numeric operator

Floor division //

Remainder or modulo %

print(7 // 2)

3

print(7 % 2)

1

Such that 3 * 2 + 1 equals 7.

In general: *x == (x // y) * y + (x % y)*

UNIVERSITY OF TWENTE.

7

The even test

print(0 % 2)
0
print(1 % 2)
1
print(2 % 2)
0
print(3 % 2)
1
print(4 % 2)
0

Observe that the remainder by 2 can be used for a test to see if a number is even or odd.

Is x even?

print(x % 2 == 0)
True #Yes!

UNIVERSITY OF TWENTE.

8

The is-even function

```
def is_even(x):
    return x % 2 == 0
```

This function returns a Boolean value.

Is 23 an even number?

is_even(23)
False

UNIVERSITY OF TWENTE.

9

Last digit

The remainder can be used for determining final digits of a number:

- What is the final digit of 969?

print(969 % 10)

9

- What are the 2 final digits of 969?

print(969 % 100)

69

UNIVERSITY OF TWENTE.

10

The last_digits function

Can we generalize this?

Yes!

```
def last_digits(x, d):
        return x % 10**d

last_digits(83271, 3)
```

271

UNIVERSITY OF TWENTE.

11

Comparisons

The operators <, <=, >=, and > only work if the input domains are ordered!

print (2 < 5)

True

print('hello' > 'world')

False

UNIVERSITY OF TWENTE.

12

Comparisons

- Sequence objects (strings, lists, tuples) may be compared to other objects with the same type
- Comparison uses ***sequential*** ordering: first, the first two items are compared, and if they differ this determines the outcome of the comparison; if they are equal, the next two items are compared, and so on, until either sequence is exhausted.
- For strings, sequential ordering becomes alphabetical ordering. It uses the Unicode codepoint number to order individual characters Thus, uppercase letters come before lowercase letters, and numbers before letters.
- If all items of two sequences are equal, the sequences are considered equal. If one sequence is an initial sub-sequence of the other, the shorter sequence is the smaller (lesser) one.

UNIVERSITY OF TWENTE.

13

Boolean expressions

Operators can be combined in such a way that the result is a Boolean value.

Such expressions are called Boolean expressions.

print(is_even(96) and last_digits(38, 1) < 5)

False

UNIVERSITY OF TWENTE.

14

Conditional execution

Suppose we want to print some text in case *x* is greater than zero.
Python has a special statement for cases like these.

It is called the **if statement**

```
x = 23
if x > 0:
        print( 'x is positive' )
```

x is positive

UNIVERSITY OF TWENTE.

15

Syntax

Observe the general layout of the *if statement*:

if boolean_expression :
    statement
    statement
} body

- If the boolean expression evaluates to True, then the body is executed.
- Normal processing proceeds afterwards

UNIVERSITY OF TWENTE.

16

Alternative execution

What if x is not positive?

```
x = -1
if x > 0:
        print( 'x is positive' )
else:
        print( 'x is not positive' )
```

x is not positive

UNIVERSITY OF TWENTE.

18

Syntax

```
if boolean_expression :
    statements1
else :
    statements2
```

- If the expression is True, then the first body of statements is executed
- If the expression is False, then the second body of statements is executed
- Normal processing proceeds afterwards

UNIVERSITY OF TWENTE.

19

Nested if statements

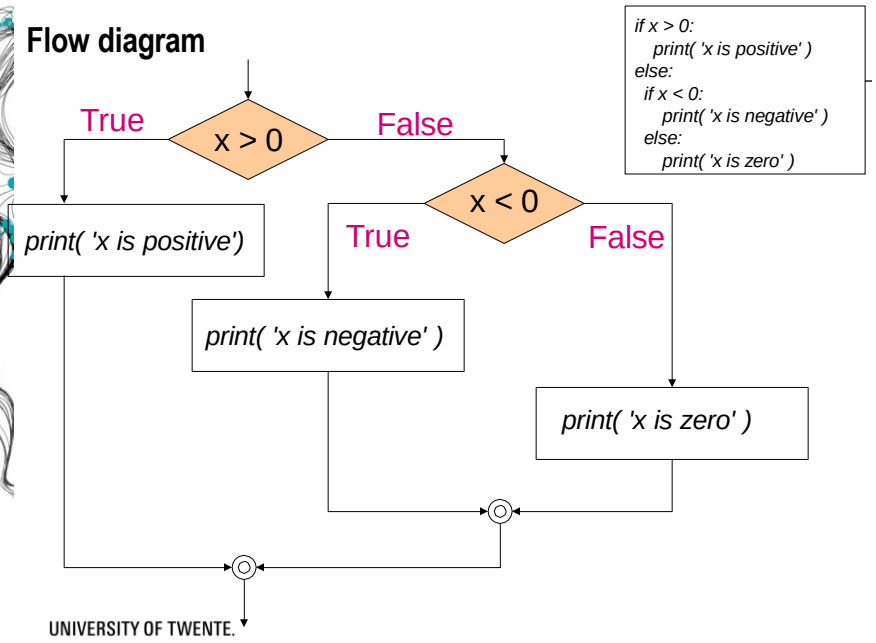
Yes, but what if x is zero?

```
if x > 0:
        print( 'x is positive')
else:
        if x < 0:
                print( 'x is negative' )
        else:
                print( 'x is zero' )
```

UNIVERSITY OF TWENTE.

21

Flow diagram




```
if x > 0:
   print( 'x is positive' )
else:
  if x < 0:
     print( 'x is negative' )
  else:
     print( 'x is zero' )
```

UNIVERSITY OF TWENTE.

22

Chained conditionals

Suppose we have numbered the days of the week and we want to map the number to a string:

- 0 → 'Monday'
- 1 → 'Tuesday'
- 2 → 'Wednesday'
- and so on ...

UNIVERSITY OF TWENTE.

23

Chained conditionals

Let us try with nested conditionals

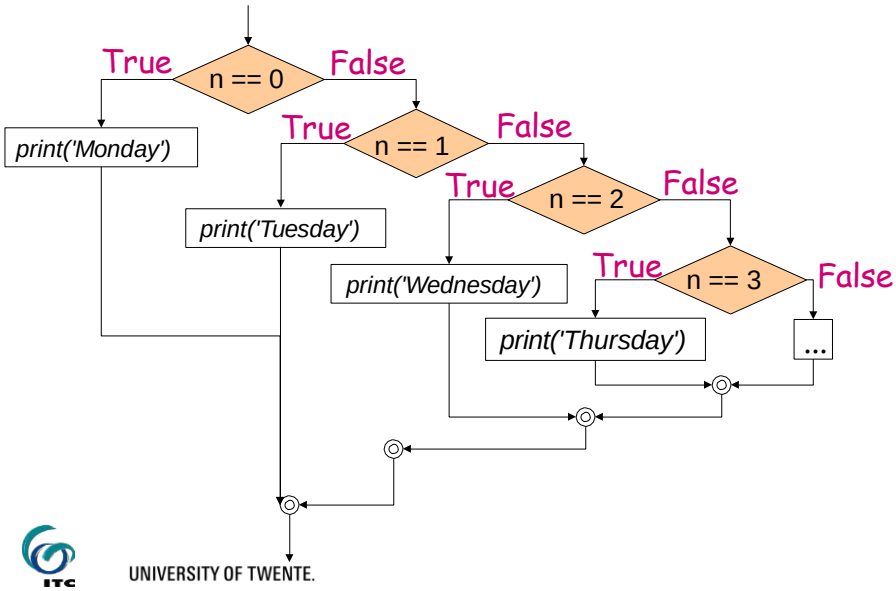
```
if n == 0:
    print('Monday)
else:
    if n == 1:
        print('Tuesday')
    else:
        if n == 2:
            print('Wednesday')
        else:
            if n == 3:
                print('Thursday')
            else:
                if n == 4:
                    print('Friday')
                else:
                    if n == 5:
                        print('Saturday')
                    else:
                        if n == 6:
                            print('Sunday')
                        else:
                            print('Error!')
```

```
if n == 0:
    print('Monday')
else:
    if n == 1:
        print('Tuesday')
    else:
        if n == 2:
            print('Wednesday')
        else:
            if n == 3:
                print('Thursday')
            ...
```

UNIVERSITY OF TWENTE.

24

Flow diagram



UNIVERSITY OF TWENTE.

25

Chained conditionals

```
if n == 0:
    print( 'Monday' )
elif n == 1:
    print( 'Tuesday' )
elif n == 2:
    print( 'Wednesday' )
elif n == 3:
    print( 'Thursday' )
elif n == 4:
    print( 'Friday' )
elif n == 5:
    print( 'Saturday' )
elif n == 6:
    print( 'Sunday' )
else:
    print( 'Error!' )
```

We will later see that this behaviour of code can also be obtained by putting all the names of the weekdays in a text list and use the variable *n* as an index over that list.

UNIVERSITY OF TWENTE.

26

Chained conditionals syntax

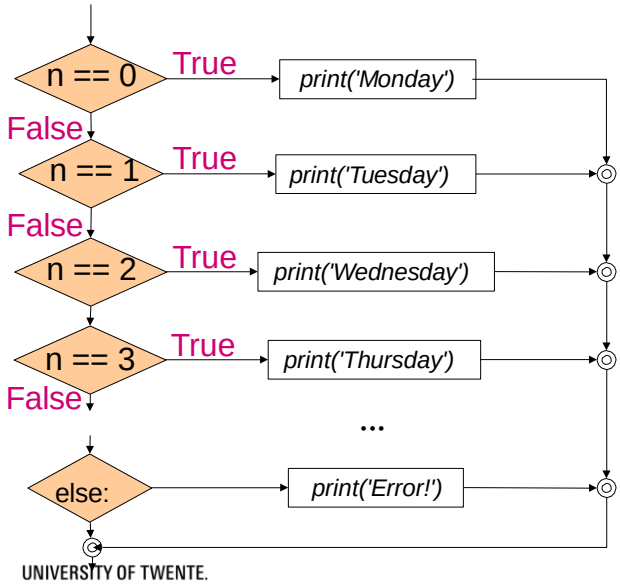
```
if boolean expression1 :
    statements1
elif boolean expresion2 :
    statements2
else:
    statementsN
```

- The IF statement body of the first expression that evaluates to True is executed
- Otherwise, the statements of the else clause at end.

UNIVERSITY OF TWENTE.

27

Flow diagram



UNIVERSITY OF TWENTE.

28

Indentation

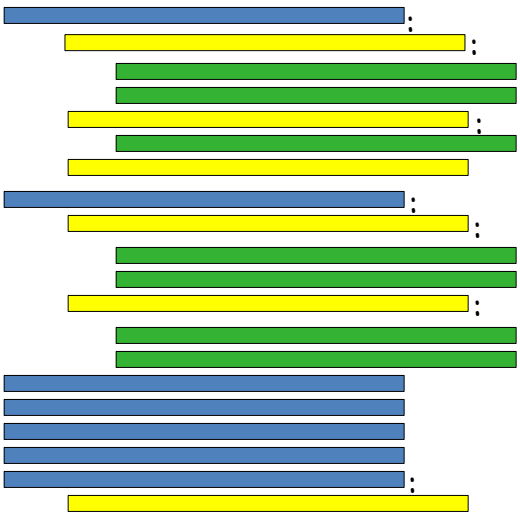
Rule of thumb:

1. A block (or body) starts after a colon :
2. Everything to the lower-right belongs to the same block
3. All the statements within the same block *must* have the same indentation!

UNIVERSITY OF TWENTE.

29

Example



UNIVERSITY OF TWENTE.

30

Common mistakes

```
if 1 == 1:
dothis()
dothat()
```

IndentationError: expected an indented block

```
if 1 + 1 == 2:
dothis()
    dothat()
```

IndentationError: expected an indented block

```
if 1 + 1 == 2
    dothis()
```

SyntaxError: invalid syntax

UNIVERSITY OF TWENTE.

31

Basic control flow

if / elif / else (test condition)

- *if x > 5 : … do something*

while (loop until condition changes)

- *while x < 5 : … do something*

for (iterate over an object)

- *for x in range(10) : … do something*

UNIVERSITY OF TWENTE.

32

Summary

- Modulo, boolean expressions
- Conditional execution *if/elif/else*

33