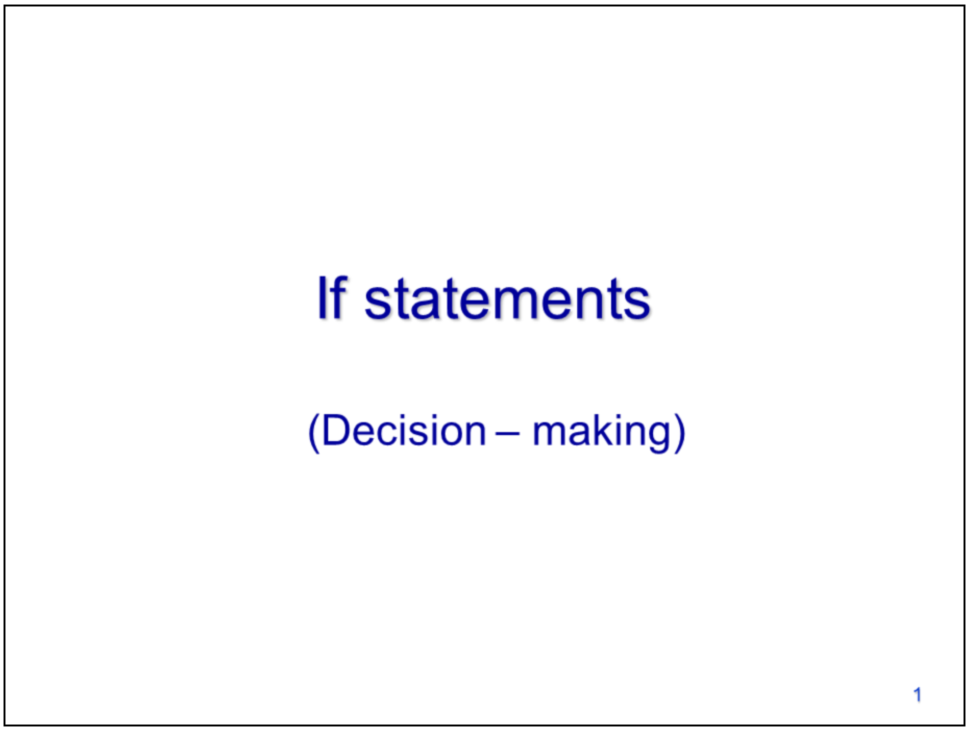
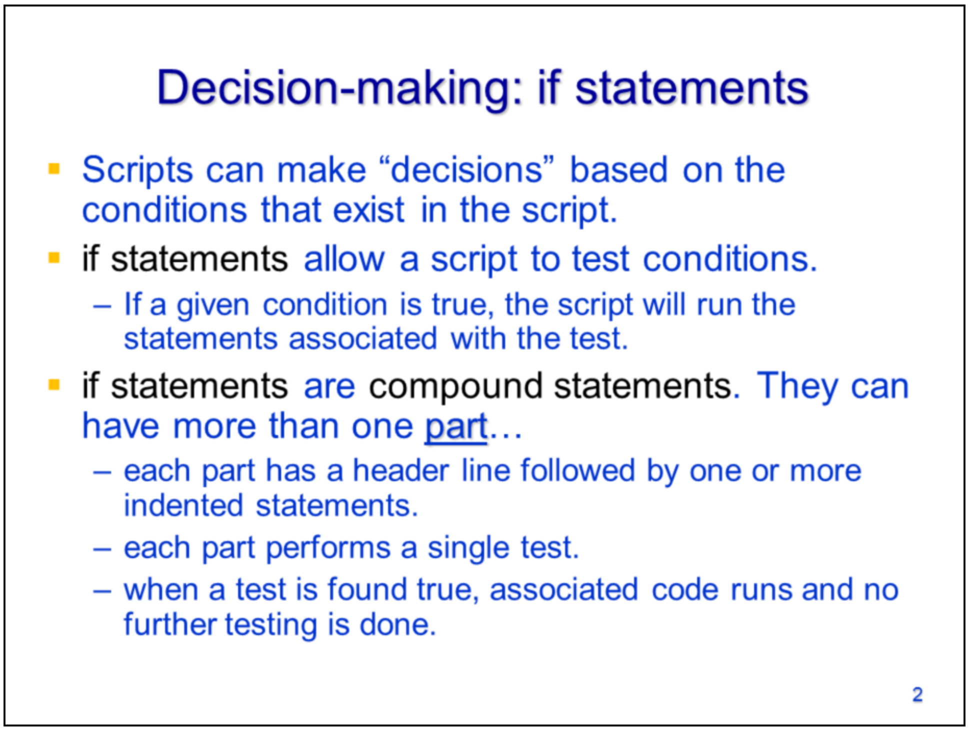


If statements

(Decision – making)

1

This video will discuss how to include decision-making capabilities in a script by using if statements.



Decision-making: if statements

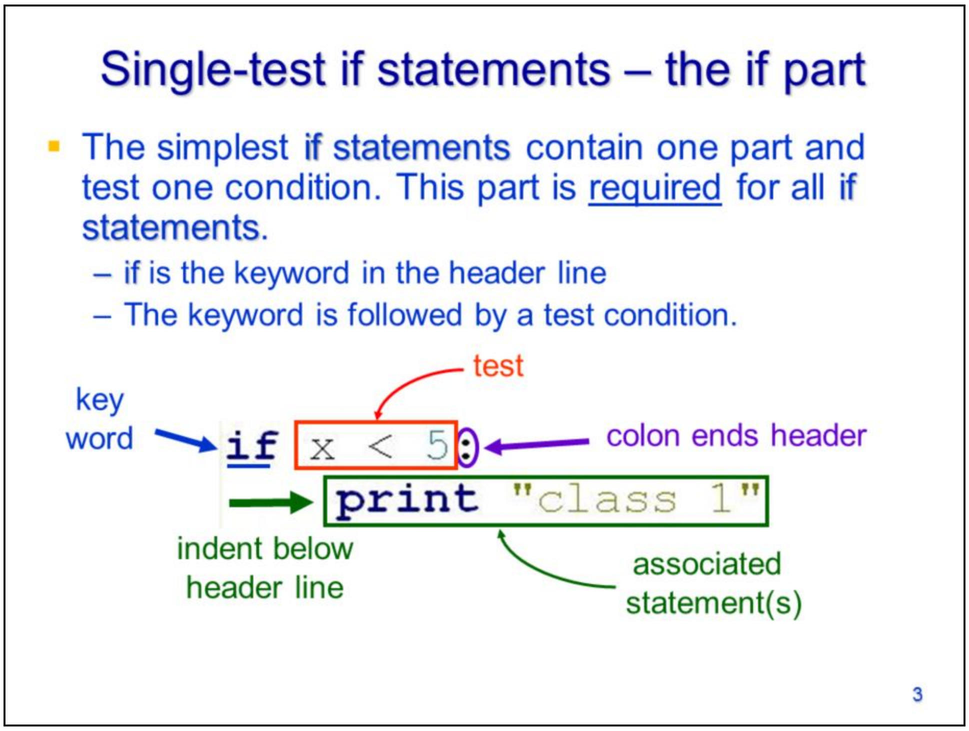
- Scripts can make "decisions" based on the conditions that exist in the script.
- if statements allow a script to test conditions.
  - If a given condition is true, the script will run the statements associated with the test.
- if statements are compound statements. They can have more than one part…
  - each part has a header line followed by one or more indented statements.
  - each part performs a single test.
  - when a test is found true, associated code runs and no further testing is done.

2

Decisions in a script are made by testing the conditions within the script.

If statements allow script conditions to be tested – if a test is true, then the statements associated with the test are run.

If statements are **compound statements** and they can have more than one part. Each part has its own header line, performs a single test, and has one or more statements associated with it.



Single-test if statements – the if part

- The simplest if statements contain one part and test one condition. This part is required for all if statements.
  - if is the keyword in the header line
  - The keyword is followed by a test condition.

```
if x < 5:
    print "class 1"
```

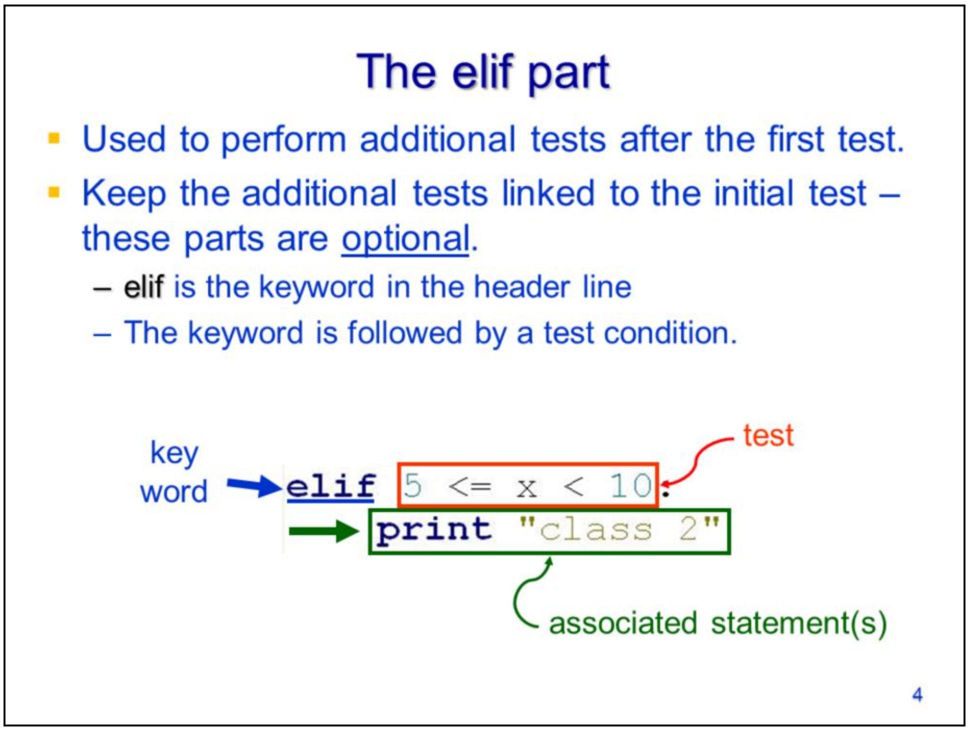
test
key word
colon ends header
indent below header line
associated statement(s)
3

An if statement must contain an if part which is itself a compound statement. It is indicated by the **if** keyword in the header line.

The keyword is followed by the test condition.

The header line ends with a colon (as with all compound statement header lines).

Indented below the header line are one or more associated statements that will run when the test is true.



The elif part

- Used to perform additional tests after the first test.
- Keep the additional tests linked to the initial test – these parts are optional.
  - elif is the keyword in the header line
  - The keyword is followed by a test condition.

```
elif 5 <= x < 10:
    print "class 2"
```

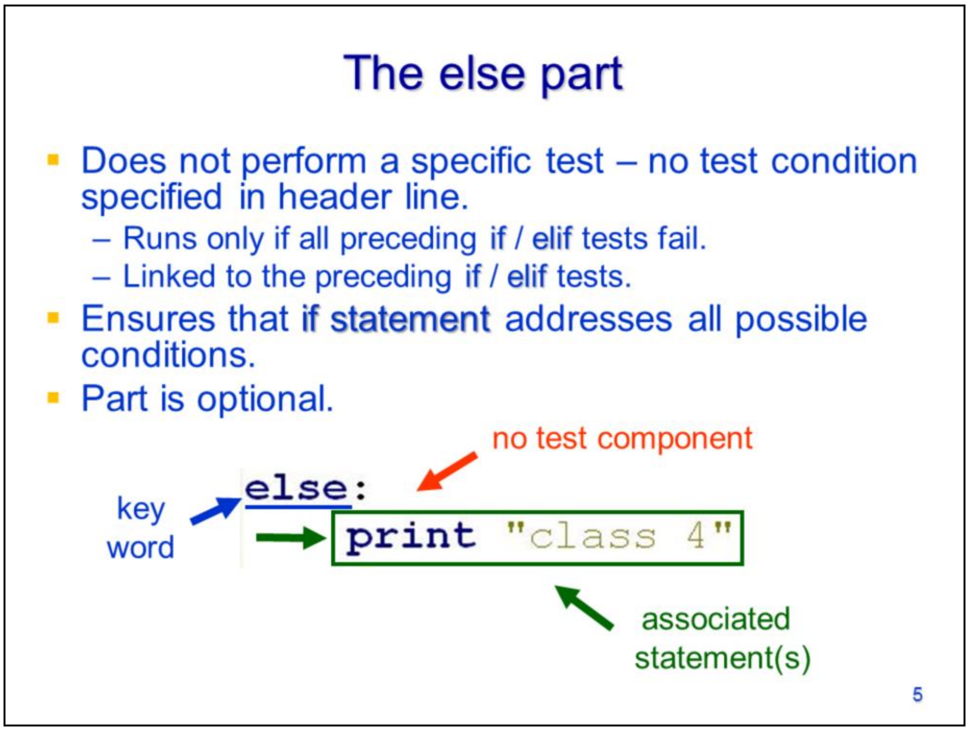
key word
test
associated statement(s)
4

The **elif part** of an if statement is optional and is only included if additional tests are needed.

The elif part is linked to the preceding tests in the if statement. This part is indicated by the **elif** keyword.

The elif is followed by a test condition

And has associated statements that will run if the test is true.



The else part

- Does not perform a specific test – no test condition specified in header line.
  - Runs only if all preceding if / elif tests fail.
  - Linked to the preceding if / elif tests.
- Ensures that if statement addresses all possible conditions.
- Part is optional.

```
else:
    print "class 4"
```

no test component
key word
associated statement(s)
5

The final part for an if statement is the **else part**. This part does not perform a specific test.

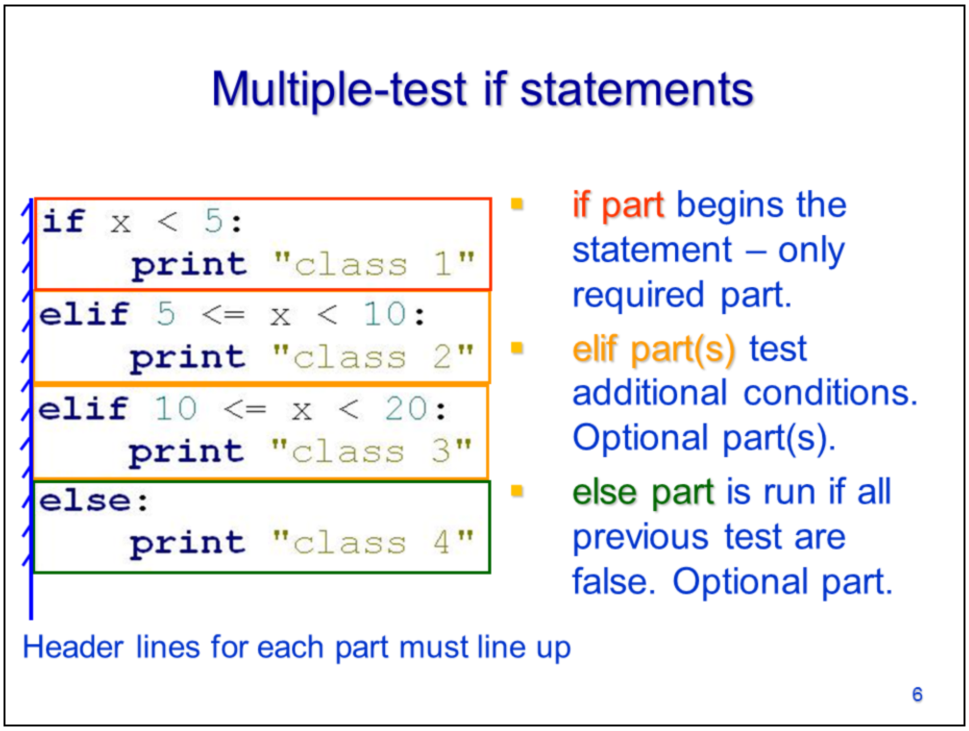
The part is indicated by the **else** keyword and there is no test in the header line.

Associated statements are indented below the header line and will run if the else test is true.

The else test is true if all preceding tests in the if statement were false.

This part ensures that the if statement accounts for all possible conditions – including those not specifically tested.

This part is optional.



Multiple-test if statements

```
if x < 5:
    print "class 1"
elif 5 <= x < 10:
    print "class 2"
elif 10 <= x < 20:
    print "class 3"
else:
    print "class 4"
```

- if part begins the statement – only required part.
- elif part(s) test additional conditions. Optional part(s).
- else part is run if all previous test are false. Optional part.

Header lines for each part must line up

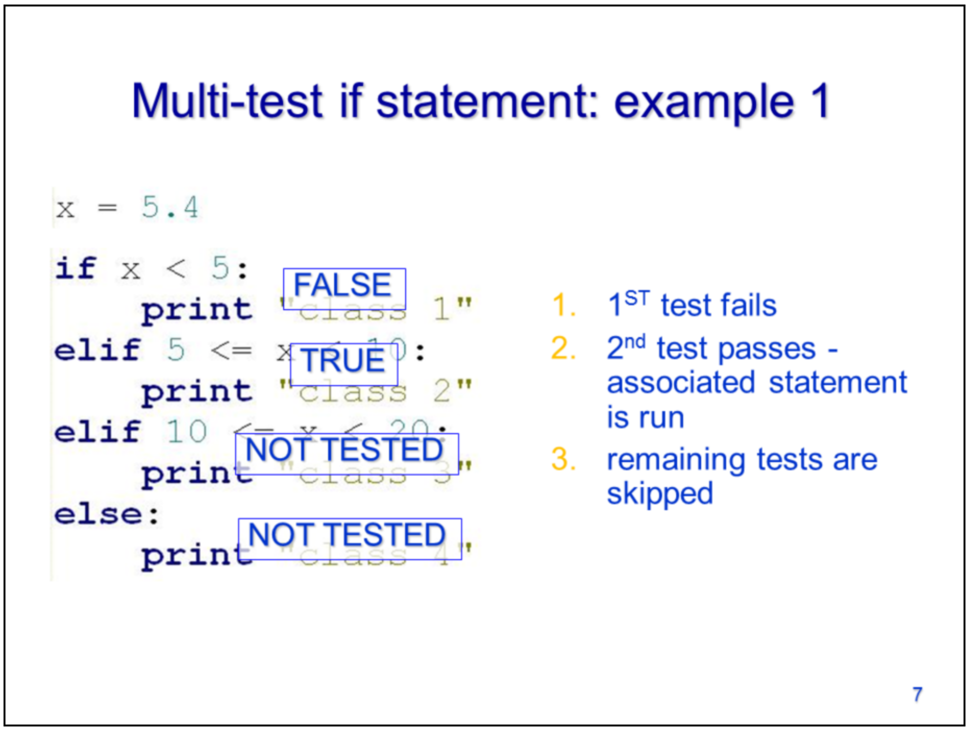
6

To summarize: the **if part** begins the statement and is the only required part.

The **elif parts** provide additional tests, if needed, and are optional.

The **else part** is the final part and is optional. It will run if all previous tests in the statement have been false.

Note that the header lines for all parts of the **if statement** have the same level of indentation.



Multi-test if statement: example 1

```
x = 5.4

if x < 5:
    print "class 1"
elif 5 <= x < 10:
    print "class 2"
elif 10 <= x < 20:
    print "class 3"
else:
    print "class 4"
```

FALSE
TRUE
NOT TESTED
NOT TESTED

1. 1ST test fails
2. 2nd test passes - associated statement is run
3. remaining tests are skipped

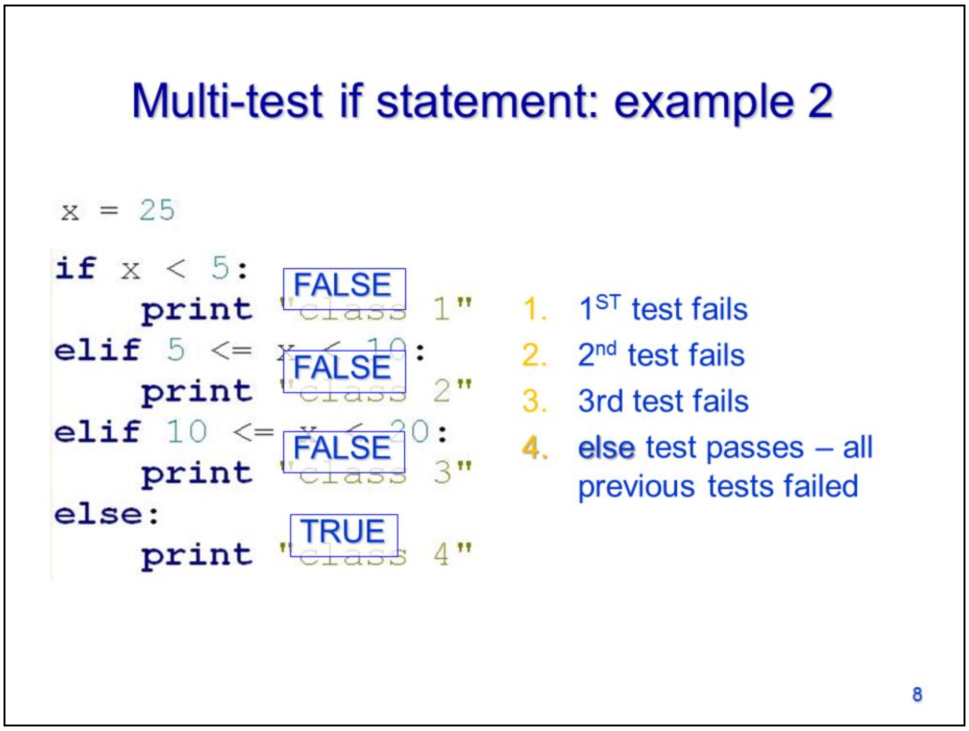
7

Let's look at an example of an if statement in operation.

The 1st test is run and fails so the testing continues to the next part.

The 2nd test passes and the associated statements are run.

All further tests in the statement are skipped.



Multi-test if statement: example 2

```
x = 25

if x < 5:
    print "class 1"
elif 5 <= x < 10:
    print "class 2"
elif 10 <= x < 20:
    print "class 3"
else:
    print "class 4"
```

FALSE

FALSE

FALSE

TRUE

1. 1ST test fails
2. 2nd test fails
3. 3rd test fails
4. else test passes – all previous tests failed

8

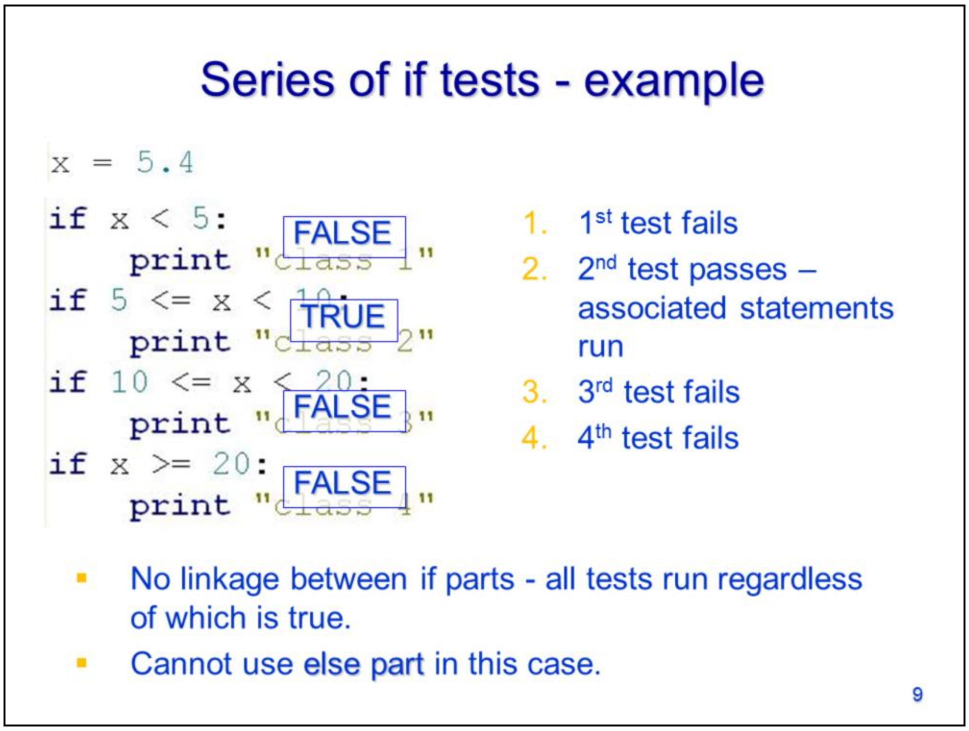
In a second example,

The 1st test fails,

The 2nd test fails,

And the 3rd test fails.

The else test passes because all previous tests in the statement have failed – the statements in the else part are run.


FALSE
TRUE
FALSE
FALSE

Series of if tests - example

```
x = 5.4
if x < 5:
    print "class 1"
if 5 <= x < 10:
    print "class 2"
if 10 <= x < 20:
    print "class 3"
if x >= 20:
    print "class 4"
```

1. 1st test fails
2. 2nd test passes – associated statements run
3. 3rd test fails
4. 4th test fails

- No linkage between if parts - all tests run regardless of which is true.
- Cannot use else part in this case.

9

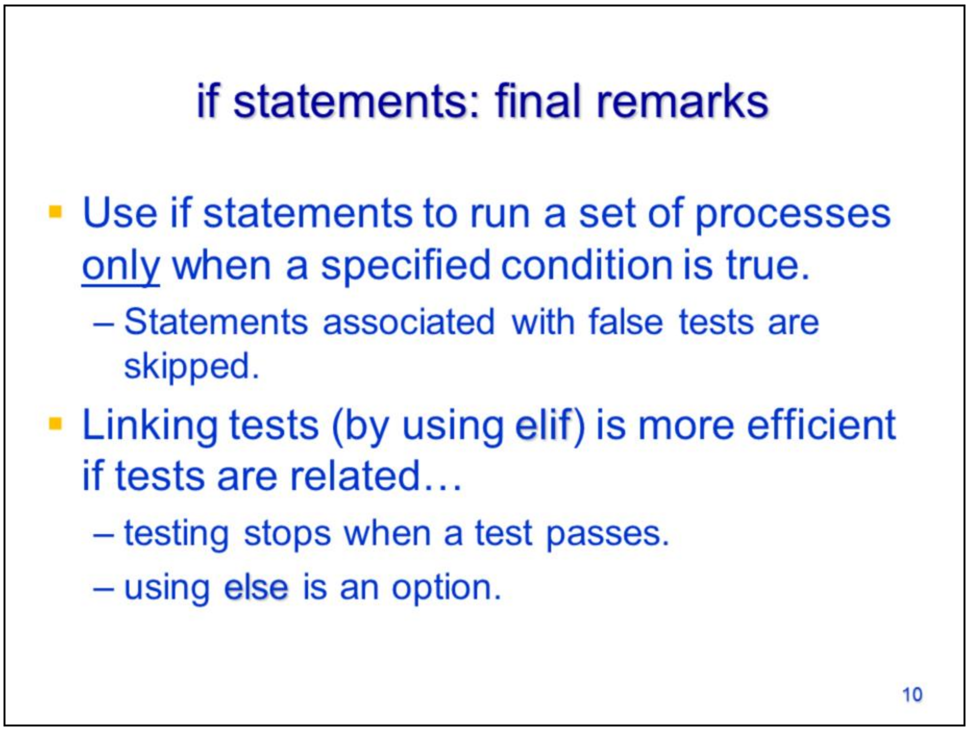
Let’s look at what happens if we use a series of if tests rather than a multi-part if statement.

The 1st test fails…

The 2nd test passes and the associated statements are run.

In this case, the 3rd and 4th tests are also run and they fail.

When a series of if tests are used, there is no linkage among the tests so all tests are performed. In this case, the else test cannot be used.

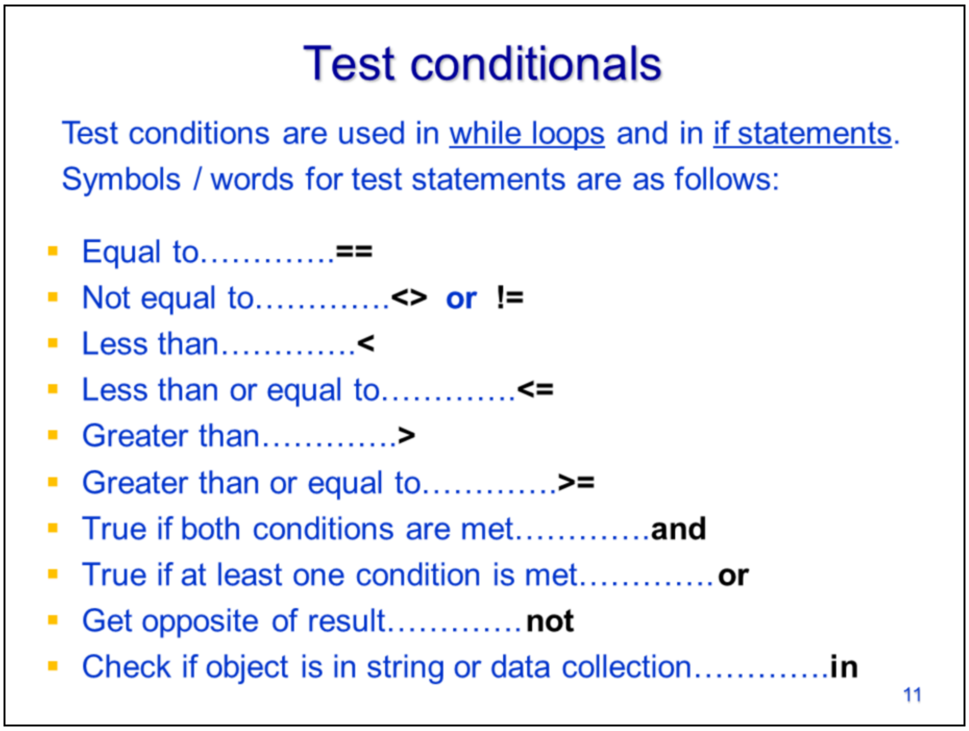


if statements: final remarks

- Use if statements to run a set of processes only when a specified condition is true.
  - Statements associated with false tests are skipped.
- Linking tests (by using elif) is more efficient if tests are related…
  - testing stops when a test passes.
  - using else is an option.

10

To conclude: if statements run a set of processes only if a specified condition is met. If a series of tests are related, then it is more efficient to use the elif parts to link the tests together. When tests are linked, then the else test can be used.



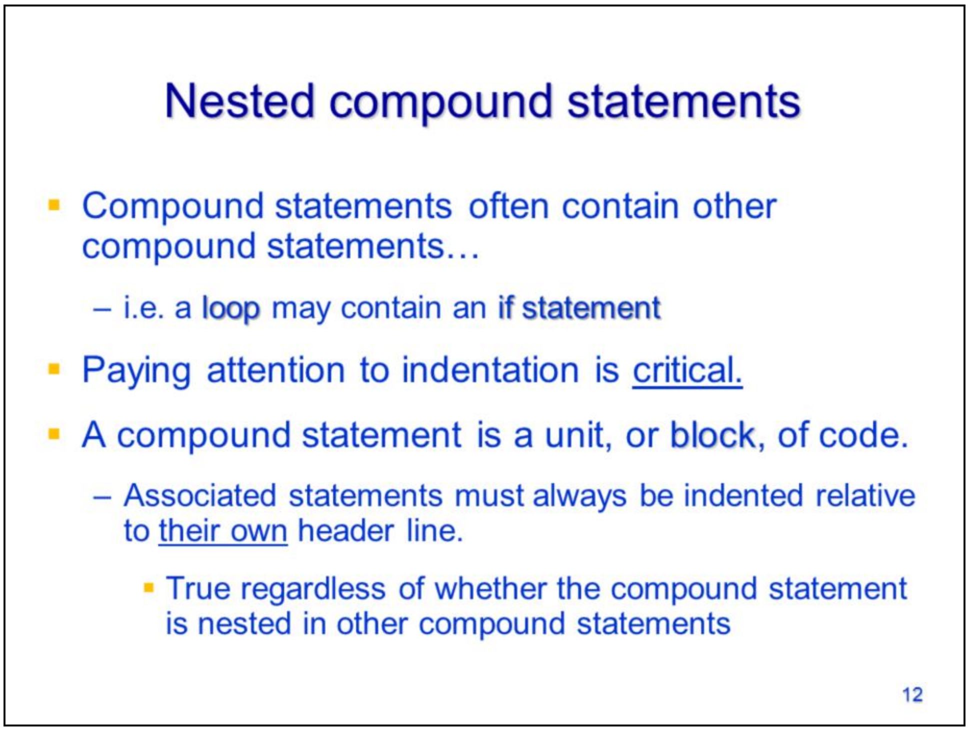
Test conditionals

Test conditions are used in while loops and in if statements.
Symbols / words for test statements are as follows:

- Equal to............**==**
- Not equal to............**<> or !=**
- Less than............**<**
- Less than or equal to............**<=**
- Greater than............**>**
- Greater than or equal to............**>=**
- True if both conditions are met............**and**
- True if at least one condition is met............**or**
- Get opposite of result............**not**
- Check if object is in string or data collection............**in**

11

This slide lists some of the test conditionals that can be used in Python. Note that tests for equality are made with a double equal sign (a single equal sign is only used to assign values to variables). Also note the symbols used for inequality tests.



Nested compound statements

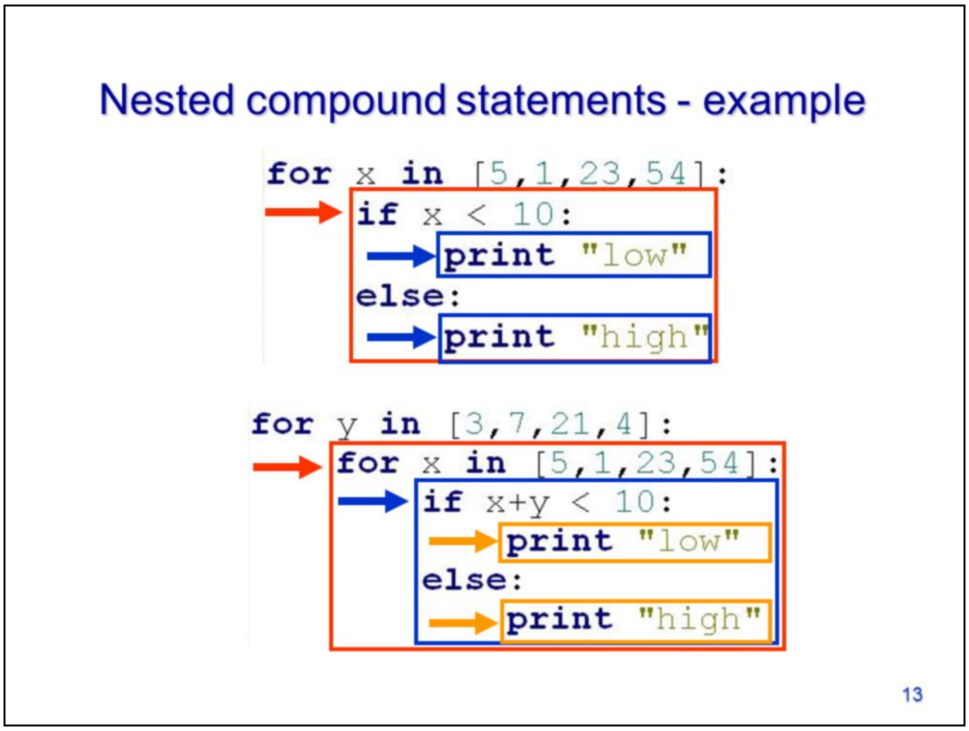
- Compound statements often contain other compound statements…
  - i.e. a **loop** may contain an **if statement**
- Paying attention to indentation is <u>critical.</u>
- A compound statement is a unit, or **block**, of code.
  - Associated statements must always be indented relative to <u>their own</u> header line.
    - True regardless of whether the compound statement is nested in other compound statements

12

Compound statements can contain other compound statements. It is common to include an if statement in a loop or a loop within a loop.

When working with nested compound statements, it is critical to pay attention to indentation.

In these cases, a given compound statement should be treated as a block of code. The associated statements should be indented relative to their corresponding header line.



Nested compound statements - example

```
for x in [5,1,23,54]:
    if x < 10:
        print "low"
    else:
        print "high"

for y in [3,7,21,4]:
    for x in [5,1,23,54]:
        if x+y < 10:
            print "low"
        else:
            print "high"
```

13

In this example,

The if statement is a block of code that is associated with the for loop.

Within the if statement, the lines associated with the if part

And else part are indented relative to their respective header lines.

In the 2nd example,

The for loop block is indented relative to the main loop.

The if statement block is indented relative to the nested for loop.

The lines associated with the if

and else parts are indented relative to their respective header lines.