GCSE

Practical programming skills in Python

Selection and iteration

Topic 2

PG ONLINE

2

Objectives

- Describe the different comparison operators
- Be able to use selection statements
- Be able to use counter controlled (for) loops
- Be able to use condition controlled (while) loops

Starter activity

- Which of these statements are true?
  - 15 == 12
  - 17 != 14
  - 12 < 8
  - 26 >= 10
  - 3 <= 3
  - 3 < 3
  - “Five” == 5

Starter activity Answers

- Which of these statements are true?

15 == 12	False
17 != 14	True
12 < 8	False
26 >= 10	True
3 <= 3	True
3 < 3	False
“Five” == 5	False

Relational operators

- There are six comparison operators commonly used in Python:
  - Equal to
  - Not equal to
  - Less than
  - Less than or equal to
  - Greater than
  - Greater than or equal to

Relational operators

- There are six comparison operators commonly used in Python:

Equal to	==
Not equal to	!=
Less than	<
Less than or equal to	<=
Greater than	>
Greater than or equal to	>=

Equality = / ==

- The ‘assignment’ operator (=) assigns a new value
- The ‘equals’ operator (==) checks if two items have the same value
- Try this code:

```
userName = "Dave"
targetName = "Dave"

if userName == targetName:
    print("Names match")
```

Change the userName and test the result

Inequality !=

- The ‘not equals’ operator (!=) checks if two items have a different value
- Try this code:

```
userName = "Bob"
targetName = "Dave"

if userName != targetName:
    print("Names don’t match")
```

Change the userName and test the result

Greater than >

- The ‘greater than’ operator (>) checks if the first item is bigger than the second
- Try this code:

```
numOne = 12
numTwo = 5

if numOne > numTwo:
    print("Number one is bigger")
```

Change the numbers and test the result

What happens if the numbers are equal?

Greater than or equal >=

- The ‘greater than or equal’ operator (>=) checks if the first item is bigger than **or equal to** the second
- Try this code:

```
numOne = 5
numTwo = 5

if numOne >= numTwo:
    print("Number one not smaller than Number two")
```

Change the numbers and test the result

Less than < (… or equal <=)

- The ‘less than’ operator (<) checks if the first item is less than the second
- The ‘less than or equal’ operator (<=) checks if the first item is smaller than **or equal to** the second
- Try this code:

```python
numOne = 3
numTwo = 5

if numOne < numTwo:
    print("Number one is smaller")
```

Greater than / Less than

- To help remember which is which, read from left to right. If the left side has a bigger opening
    - 12 is greater than 6
    - 12 > 6
- If the left side has a smaller opening
    - 5 < 8
    - 5 is less than 8

Data types

- When comparing two different data types, the values will always be different
- Try this code:

```
numOne = 3
numTwo = "3"

if numOne != numTwo:
    print("Values are different")
```

Comparing strings

- Using greater than or less than on strings will compare them alphabetically
- Try this code:

```
animalOne = "aardvark"
animalTwo = "bird"

if animalOne < animalTwo:
    print("Animal one is earlier in the alphabet")
```

Comparison operators

- Remember the six comparison operators:

==	-	Equal to
!=	-	Not equal to
>	-	Greater than
>=	-	Greater than or equal to
<	-	Less than
<=	-	Less than or equal to

Worksheet 2a

- Complete **Questions 1 & 2**

Selection statements

- Selection statements come in three main types:
  - IF ______ THEN ______
  - IF ______ THEN ______

    ELSE ______
  - IF ______ THEN ______

    ELSE IF ______ THEN ______

    ELSE ______

if …

- Try this code:

```
lives = 0
if lives < 1:
    print("Game Over")
```

- Change lives to 1 and test the program
- The print statement only runs if lives is less than 1
- If the value is bigger then nothing happens

if … else …

- Try this code:

```
lives = 0
if lives < 1:
    print("Game Over")
else:
    print("Carry on")
```

- Now there are two possible outcomes
- There is always some response printed to the screen

if … elif … else …

- Try this code:

```python
lives = 10

if lives < 1:
    print("Game Over")
elif lives >= 10:
    print("Bonus Round!")
else:
    print("Carry on")
```

- Now there are many possible outcomes
- Keep testing different values for `lives`

if vs elif

- Try this code:

```
score = 92

if score >= 90:
    print("Grade 8!")
if score >= 80:
    print("Grade 7!")
if score >= 70:
    print("Grade 6!")
```

if vs elif

- Try this code:

```
score = 92

if score >= 90:
    print("Grade 8!")
if score >= 80:
    print("Grade 7!")
if score >= 70:
    print("Grade 6!")
```

- The program prints out all three possible grades
  - Why?

if vs elif

- Try this code:

```
score = 92

if score >= 90:
    print("Grade 8!")
if score >= 80:
    print("Grade 7!")
if score >= 70:
    print("Grade 6!")
```

- The program prints out all 3 possible grades
  - Why? **All three if statements are true**

if vs elif

- Try this code:

```
score = 92

if score >= 90:
    print("Grade 8!")
elif score >= 80:
    print("Grade 7!")
elif score >= 70:
    print("Grade 6!")
```

- Try different values for `score`

if vs elif

- When using `if … elif …` , the selection statement stops checking when it finds a positive match
    - This means fewer unnecessary responses
    - This also means a faster program (fewer comparisons need to be made)
- It is a good habit to always include an `else` statement to catch any unexpected results

Worksheet 2a

- Complete **Questions 3 and 4**

Count controlled loops (`for`)

- Try this code:

```
for count in range(5):
    print("Repeat:", count)
```

Count controlled loops (`for`)

- Try this code:

```
for count in range(5):
    print("Repeat:", count)
```

- `for` loops are useful when you know how many times you want to repeat a block of code
- The number in brackets controls how many repetitions there are
- `count` increases from 0 to 4, so the loop is repeated 5 times

Count controlled loops (`for`)

- Try this code:

```
for count in range(1,5):
    print("Repeat:", count)
```

- `count` starts at 1 and ends at 4 (not 5)
- Try:
  - Making the loop repeat 8 times
  - Displaying the numbers 1 to 8, instead of 0 to 7

Running Total

- Try this code:

```
total = 0
for count in range(5):
    newValue = int(input("New number: "))
    total = total + newValue
print("Total:", total)
```

- Notice:
  - The total must be set **before** the loop starts
  - Printing the total happens only **after** the loop has completed

Worksheet 2b

- Complete **Questions 1 and 2**

Condition controlled loops (`while`)

- Try this code:

```
target = 8
guess = 0
while guess != target:
    guess = int(input("Guess a number: "))
print("You got it right!")
```

- What would happen if `guess` was assigned the value 8 instead of `0` in the second line?

Condition controlled loops (`while`)

- Try this code:

```
target = 8
guess = 0
while guess != target:
    guess = int(input("Guess a number: "))
print("You got it right!")
```

- `While` loops are useful when you don’t know how many times to repeat a block of code
- Set a value so that the loop runs at least once

Running total

- Try this code:

```
total = 0
while total < 10:
    newValue = int(input("New number: "))
    total = total + newValue
print("Total:", total)
```

- Notice:
  - Total is set to 0 so the loop will run at least once
  - Printing the total happens only **after** the loop has completed

Running total

- Try this code:

```
total = 0
newValue = 999
while newValue != 0:
    newValue = int(input("New number: "))
    total = total + newValue
print("Total:", total)
```

- Notice:
  - `newValue` is set to 999 so the loop will run at least once

Worksheet 2b

- Complete **Questions 3 and 4**

Plenary

- Identify the right construct for each option:
  - Collecting cricket scores (6 balls to an over)
  - Collecting snooker scores (play until someone wins 3 frames)
  - Deciding which team has won a match after 90 minutes

Plenary

- Identify the right construct for each option:
  - Collecting cricket scores (6 balls to an over)
    - – For loop (must repeat 6 times)
  - Collecting snooker scores (play until someone wins 3 frames)
    - – While loop (repeat while frames won < 3)
  - Deciding which team has won a match after 90 minutes
    - – If statement (if teamAScore > teamBScore)

Copyright

Licence agreement