Control Flow

- Conditional Statements
- Loops

Conditional Statements

- Using if tests
- Nesting if tests
- Using the if-else operator
- Doing nothing
- Testing a value is in a set of values
- Testing a value is in a range

Using if tests

Basic if tests

```
if expression:
  body
```

if-else tests

```
if expression:
  body1
else:
  body2
```

if-elif tests

```
if expression1 :
  body1

elif expression2 :
  body2

elif expression3 :
  body3
…
else :
  lastBody
```

Notes:

- Test conditions can be any type of expression
- Use indentation to indicate the extent of a block, i.e. don’t use {}

Nesting if tests

You can nest if tests inside each other

- Use indentation to indicate level of nesting

Example:

```
age = int(input("Please enter your age: "))
gender = input("Please enter your gender [M/F]: ").low

if age < 18:
  if gender == "m":
    print("Boy")
  else:
    print("Girl")

else:
  if age >= 100:
    print("Centurion")

  if gender == "m":
    print("Man")
  else:
    print("Woman")

print("The End")
```

Using the if-else operator

The if-else operator is an in-situ test

- trueResult if condition else falseResult

Example:

```
1   isMale = …
2   age    = …
3
4   togo = (65 - age) if isMale else (60 - age)
5
6   print("%d years to retirement" % togo)
```

Doing nothing

If you're not sure what to do if a test is true…

- You can use the pass statement
- Equivalent to a null statement in other languages

Example:

```
team = input("Who is your favourite rugby team? ")

if team == "Ireland":
  pass     # Eeek. We'll need to do something about this!

print("Your favourite team is %s " % team)
```

Testing a value is in a set of values

You can test if a value is in a set of allowable values

- Use the in operator

Example:

```
country = input("Please enter your country: ")

if country in ("Netherlands", "Belgium", "Luxembourg"):
  print("Lowlands country")

elif country in ("Norway", "Sweden", "Denmark", "Finland", "Iceland"):
  print("Nordic country")

elif country in ("England", "Scotland", "Wales", "Northern Ireland"):
  print("UK country")

else:
  print("%s isn't classified in this particular application!" % country)
```

Testing a value is in a range

You can test if a value is in a range of allowable values

- Call range(start,end) to return a range
- The range is inclusive at start, exclusive at the end

Example:

```
number = int(input("Enter a football jersey number [1 to 11]: "))

if number == 1:
  print("Goalie")

elif number in range(2, 6):
  print("Defender")

elif number in range(6, 10):
  print("Midfielder")

else:
  print("Striker")
```

Loops

- Using while loops
- Using for loops
- Using for loops with a range
- Unconditional jumps
- Using else in a loop
- Simulating do-while loops

Using while loops

The while loop is the most straightforward loop construct

```
while expression:
  loopBody
```

- Test expression is evaluated
- If true, loop body is executed
- Test expression is re-evaluated
- Etc…

Note:

- Loop body will not be executed if test is false initially

Example:

```
print("Numbers from 1-5 inclusive")
i = 1
while i ⩽ 5:
  print(i)
  i = i + 1
```

Using for loops

The for loop is different than in most languages

- In Python, a for loop iterates over items in a sequence

```
for item in sequence:
  loopBody
```

Example:

```
lottonumbers = [2, 7, 3, 12, 19, 1]

for item in lottonumbers:
  print(item)
```

Using for loops with a range

You can also use a for loop to iterate over a numeric range

- Use range() to create a range of numbers
- The for loop will iterate over these numbers

Example:

```
print("Numbers from 0-4 inclusive")
for i in range(5):
  print(i)

print("Numbers from 6-10 inclusive")
for i in range(6, 11):
  print(i)

print("First 5 odd numbers")
for i in range(0, 9, 2):
  print(i + 1)
```

Unconditional jumps

Python provides two ways to perform an unconditional jump in a loop

- break
- continue

Example:

```
1   magicnumber = int(input("What is the magic number? "))
2
3   print("This loop terminates if it hits the magic number")
4   for i in range(1, 21):
5     if i == magicnumber:
6       break
7     print(i)
8   print("End")
9
10  print("\nThis loop skips the magic number")
11  for i in range(1, 21):
12    if i == magicnumber:
13      continue
14    print(i)
15  print("End")
```

Using else in a loop

You can define an else clause at the end of a loop

- Same kind of syntax as if…else
- The else branch is executed if the loop terminates naturally (i.e. if it didn't exit because of a break)

Example

```
1   magicnumber = int(input("What is the magic number? "))
2
3   print("This loop does some processing if it doesn't detect the magic number")
4   for i in range(1, 21):
5     if i == magicnumber:
6       break
7     print(i)
8   else:
9     print("The magic number %d was not detected" % magicnumber)
10
11  print("End")
```

Simulating do-while loops

Many languages have a do-while loop

- Guarantees at least one iteration through the loop body
- The test is at the end, to determine whether to repeat

Python doesn't have a do-while loop, but you can emulate it as follows

```
while True:
  exammark = int(input("Enter a valid exam mark: "))
  if exammark ⩾ 0 and exammark ⩽ 100:
    break

print("Your exam mark is %d" % exammark)
```

Any questions?