COMP110: Principles of Computing

3: Flowcharts and pseudocode

Learning outcomes

- Outcome 1
- Outcome 2
- Outcome 3

FALMOUTH
UNIVERSITY

Loops (from last time)

For loops and ranges

```
for i in xrange(5):
    print i
```

For loops and ranges

```
for i in xrange(5):
    print i
```

- `xrange(n)` is the **sequence** $0, 1, 2, \ldots, n-1$

For loops and ranges

```
for i in xrange(5):
    print i
```

- `xrange(n)` is the **sequence** $0, 1, 2, \ldots, n-1$
- So `xrange(5)` is the **sequence** $0, 1, 2, 3, 4$

For loops and ranges

```
for i in xrange(5):
    print i
```

- `xrange(n)` is the **sequence** $0, 1, 2, \ldots, n-1$
- So `xrange(5)` is the **sequence** $0, 1, 2, 3, 4$
- Note: `xrange(n)` **does not include** n

For loops and ranges

```
for i in xrange(5):
    print i
```

- `xrange(n)` is the **sequence** $0, 1, 2, \ldots, n-1$
- So `xrange(5)` is the **sequence** $0, 1, 2, 3, 4$
- Note: `xrange(n)` **does not include** n
- The `for` loop iterates through the items in a sequence **in order**

For loops and ranges

```
for i in xrange(5):
    print i
```

- `xrange(n)` is the **sequence** $0, 1, 2, \ldots, n-1$
- So `xrange(5)` is the **sequence** $0, 1, 2, 3, 4$
- Note: `xrange(n)` **does not include** n
- The `for` loop iterates through the items in a sequence **in order**
- Can also use `range` instead of `xrange`, but `range` is less efficient
  - Homework (advanced): what is the difference between `range` and `xrange`?

For loops (1)

Socrative room code: `FALCOMPED`

```
a = 0
b = 0

for i in xrange(5):
    a = i
    b = b + i

print a
print b
```

For loops (1)

Socrative room code: FALCOMPED

```
a = 0
b = 0

for i in xrange(5):
    a = i
    b = b + i

print a
print b
```

Variable	Value
a	
b	
i	

For loops (2)

Socrative room code: FALCOMPED

```
a = 0
b = 0

for i in xrange(10):
    if i < 3 or i > 7:
        a += i
    else:
        b += i

print a
print b
```

For loops (2)

Socrative room code: FALCOMPED

```
a = 0
b = 0

for i in xrange(10):
    if i < 3 or i > 7:
        a += i
    else:
        b += i

print a
print b
```

Variable	Value
a	
b	
i	

More ranges

More ranges

- Can optionally specify **start point**

More ranges

- Can optionally specify **start point**
- `xrange(3, 10)` $\rightarrow [3, 4, 5, 6, 7, 8, 9]$

More ranges

- Can optionally specify **start point**
- `xrange(3, 10)` $\rightarrow [3, 4, 5, 6, 7, 8, 9]$
- If start point is specified, can optionally specify **step**

More ranges

- Can optionally specify **start point**
- `xrange(3, 10)` $\rightarrow [3, 4, 5, 6, 7, 8, 9]$
- If start point is specified, can optionally specify **step**
- `xrange(0, 20, 2)` $\rightarrow [0, 2, 4, 6, 8, 10, 12, 14, 16, 18]$

More ranges

- Can optionally specify **start point**
- `xrange(3, 10)` $\rightarrow [3, 4, 5, 6, 7, 8, 9]$
- If start point is specified, can optionally specify **step**
- `xrange(0, 20, 2)` $\rightarrow [0, 2, 4, 6, 8, 10, 12, 14, 16, 18]$
- Step can be negative:

More ranges

- Can optionally specify **start point**
- `xrange(3, 10)` $\rightarrow [3, 4, 5, 6, 7, 8, 9]$
- If start point is specified, can optionally specify **step**
- `xrange(0, 20, 2)` $\rightarrow [0, 2, 4, 6, 8, 10, 12, 14, 16, 18]$
- Step can be negative:
- `xrange(10, 0, -1)` $\rightarrow [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]$

While loops

Socrative room code: `FALCOMPED`

The `while` loop keeps executing while the condition is **true**

```
a = 1

while a < 100:
    a = a * 2

print a
```

While loops

Socrative room code: `FALCOMPED`

The `while` loop keeps executing while the condition is **true**

```
a = 1

while a < 100:
    a = a * 2

print a
```

Variable	Value
a	

Looping forever

```
a = 1

while True:
    a = a * 2
    print a
```

Summary

Summary

We have seen some basic code constructions in Python

Summary

We have seen some basic code constructions in Python

- `print` and `raw_input` for command-line input and output

Summary

We have seen some basic code constructions in Python

- `print` and `raw_input` for command-line input and output
- Variable assignment using =

Summary

We have seen some basic code constructions in Python

- `print` and `raw_input` for command-line input and output
- Variable assignment using =
- `if` statements for choosing whether or not to execute a block of code

Summary

We have seen some basic code constructions in Python

- `print` and `raw_input` for command-line input and output
- Variable assignment using =
- `if` statements for choosing whether or not to execute a block of code
- `for` loops to execute a block of code a specified number of times

Summary

We have seen some basic code constructions in Python

- `print` and `raw_input` for command-line input and output
- Variable assignment using =
- `if` statements for choosing whether or not to execute a block of code
- `for` loops to execute a block of code a specified number of times
- `while` loops to execute a block of code until a condition is no longer true

Summary

We have seen some basic code constructions in Python

- `print` and `raw_input` for command-line input and output
- Variable assignment using =
- `if` statements for choosing whether or not to execute a block of code
- `for` loops to execute a block of code a specified number of times
- `while` loops to execute a block of code until a condition is no longer true

These are enough to write some simple programs, but you will see several more in coming weeks...

FALMOUTH
UNIVERSITY

Flowcharts

FALMOUTH
UNIVERSITY

Pseudocode