



COMP110: Principles of Computing

3: Flowcharts and pseudocode

Learning outcomes

- ▶ Produce and explain basic flowcharts
- ▶ Produce and explain basic pseudocode

Worksheet B

- ▶ Flowcharts and pseudocode
- ▶ Due in class on **Monday 10th October** (next week)

Reading

B. Shneiderman, R. Mayer, D. McKay, and P. Heller, 1977.
Experimental Investigations of the Utility of Detailed
Flowcharts in Programming. *Communications of the ACM*,
20(6):373–381.

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** $0, 1, 2, \dots, n - 1$

For loops and ranges

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

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

More ranges

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

Algorithms



What is an algorithm?

What is an algorithm?

A **sequence of instructions** which can be followed **step by step** to perform a **computational task**.

Programs vs algorithms

Programs vs algorithms

- ▶ A program is **specific** to a particular programming language and/or machine

Programs vs algorithms

- ▶ A program is **specific** to a particular programming language and/or machine
- ▶ An algorithm is **general**

Programs vs algorithms

- ▶ A program is **specific** to a particular programming language and/or machine
- ▶ An algorithm is **general**
- ▶ An algorithm must be **implemented** as a program before a computer can run it

Programs vs algorithms

- ▶ A program is **specific** to a particular programming language and/or machine
- ▶ An algorithm is **general**
- ▶ An algorithm must be **implemented** as a program before a computer can run it
- ▶ An algorithm generally performs **one task**, whereas a program may perform **many**

Programs vs algorithms

- ▶ A program is **specific** to a particular programming language and/or machine
- ▶ An algorithm is **general**
- ▶ An algorithm must be **implemented** as a program before a computer can run it
- ▶ An algorithm generally performs **one task**, whereas a program may perform **many**
 - ▶ E.g. Microsoft Word is not an algorithm, but it implements many algorithms

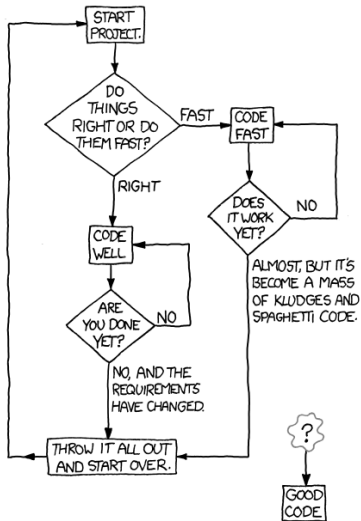
Programs vs algorithms

- ▶ A program is **specific** to a particular programming language and/or machine
- ▶ An algorithm is **general**
- ▶ An algorithm must be **implemented** as a program before a computer can run it
- ▶ An algorithm generally performs **one task**, whereas a program may perform **many**
 - ▶ E.g. Microsoft Word is not an algorithm, but it implements many algorithms
 - ▶ E.g. it implements an algorithm for determining where to break a line of text, how much space to add to centre a line, etc.

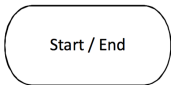
Flowcharts



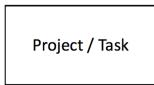
HOW TO WRITE GOOD CODE:



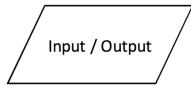
Flowchart symbols



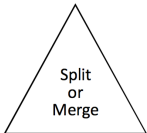
The start or end of a workflow.



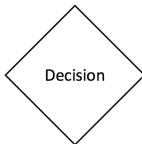
Process or action.



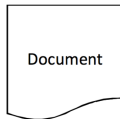
Data: Inputs to, and outputs from, a process.



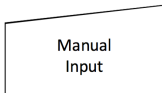
Upright indicates a process split,
inverted indicates a merge of processes.



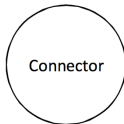
Decision point in a
process or workflow.



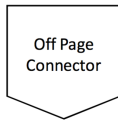
Document or report.



Prompt for information, manually
entered into a system.

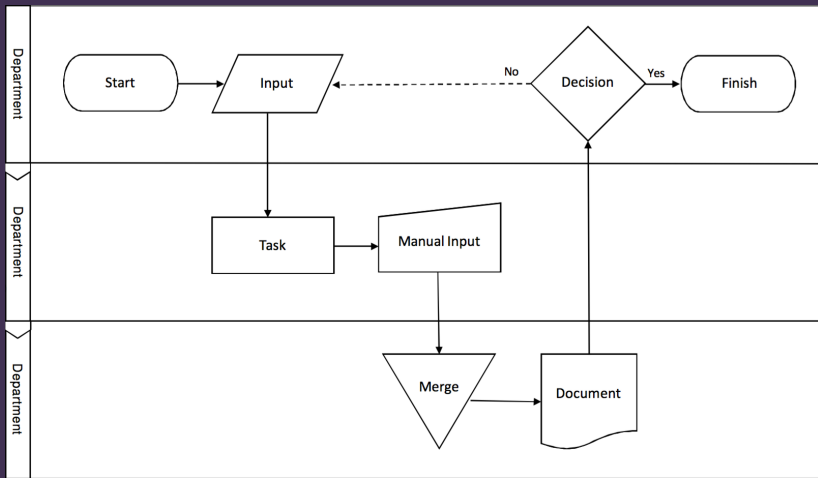


Used to connect one part of
a flowchart to another.



Connector used to connect one
page of a flowchart to another.

Swimlanes



Activity

- ▶ In **groups of 2-3**
- ▶ **Draw** a flowchart for **logging into Facebook**
- ▶ Draw your flowchart using **pen and paper**
- ▶ Include at least two swimlanes: **the user's browser/device** and **the Facebook server**

Software for drawing flowcharts

Software for drawing flowcharts

Intended for drawing flowcharts:

- ▶ Gliffy <https://www.gliffy.com>
- ▶ Microsoft Visio

Software for drawing flowcharts

Intended for drawing flowcharts:

- ▶ Gliffy <https://www.gliffy.com>
- ▶ Microsoft Visio

Can draw flowcharts:

- ▶ Microsoft PowerPoint
- ▶ Google Docs

Software for drawing flowcharts

Intended for drawing flowcharts:

- ▶ Gliffy <https://www.gliffy.com>
- ▶ Microsoft Visio

Can draw flowcharts:

- ▶ Microsoft PowerPoint
- ▶ Google Docs

If you're desperate:

- ▶ Any drawing package (Inkscape, Adobe Illustrator, Apple Keynote, ...)
- ▶ MS Paint

Pseudocode



Pseudocode

Pseudocode

Flowcharts are useful, but...

Pseudocode

Flowcharts are useful, but...

- ▶ Can be time-consuming to draw

Pseudocode

Flowcharts are useful, but...

- ▶ Can be time-consuming to draw
- ▶ Do not reflect structured programming concepts well

Pseudocode

Flowcharts are useful, but...

- ▶ Can be time-consuming to draw
- ▶ Do not reflect structured programming concepts well

Pseudocode expresses an algorithm in a way that looks more like a structured program

Pseudocode example

```
print "How old are you?"  
read age  
if age < 13 then  
    print "You are a child"  
else if age < 18 then  
    print "You are a teenager"  
else  
    print "You are an adult"  
end if
```

Pseudocode example

```
sum  $\leftarrow$  0                                ▷ initialisation  
for i in 1, ..., 9 do  
    sum  $\leftarrow$  sum + i  
end for  
print sum                                ▷ print the result
```

Pseudocode example

```
 $a \leftarrow 1$                                 ▷ initialisation  
while  $a < 100$  do  
     $a \leftarrow a \times 2$   
end while  
print  $a$                                 ▷ print the result
```

Formatting pseudocode

Formatting pseudocode

- ▶ Pseudocode is a **communication tool**, not a **programming language**

Formatting pseudocode

- ▶ Pseudocode is a **communication tool**, not a **programming language**
- ▶ Important: **clear, concise, unambiguous, consistent**

Formatting pseudocode

- ▶ Pseudocode is a **communication tool**, not a **programming language**
- ▶ Important: **clear, concise, unambiguous, consistent**
- ▶ **Not** important: adhering to a strict set of style guidelines, ensuring direct translatability to your chosen programming language

Level of abstraction

Level of abstraction

Whether working with flowcharts or pseudocode, choose your **level of abstraction** carefully

Level of abstraction: Good

Fill kettle

Turn kettle on

Put instant coffee in mug

if sugar wanted **then**

 Add sugar

end if

Wait for kettle to boil

if milk wanted **then**

 Pour water to $\frac{4}{5}$ full

 Add milk

else

 Fill mug with water

end if

Stir

Level of abstraction: Not so good

Position kettle beneath tap

Turn tap on

while water is below halfway point **do**

 Wait

end while

Turn tap off

Place kettle on base

Press power button

...

Level of abstraction: Silly

Place right palm on kettle handle

Bend fingers on right hand

Lift arm upwards

while tap spout is not directly above kettle **do**

 Move arm to the right

end while

Place left palm on tap handle

Bend fingers on left hand

Rotate left hand

...

Level of abstraction: also silly

Make a cup of coffee

Activity

A number guessing game: The computer chooses a number between 1 and 20 at random. The player guesses a number. The computer says whether the guessed number is “too high”, “too low” or “correct”. The game ends when the correct number is guessed, or after 5 incorrect guesses.

- ▶ In **groups of 2-3**
- ▶ **Write** pseudocode for the number guessing game
- ▶ Write your pseudocode with **pen and paper** or using your favourite **text editor or word processor**