

Coursework key dates

Assignment: Single piece of coursework, completed individually.

Theme: Write a program to perform an encryption and decryption task + a short (2 page) report

Set: Friday 19th November 2021 (Week 8)

Deadline: Friday 10th December 2021 (Week 11)

Drop-in support classes

On-campus Group 1	On-campus Group 2	Online
Friday	Friday	Thursday
13:00-14:00	14:00-15:00	9:00-10:00
MVB 1.15	MVB 1.15	remo

Group 1 : EMAT, Innovation, Biorobotics, everyone not listed in Groups 1/2

Group 2: EENG, EDES, Digital Health MSc & CDT

Introduction to Computer Programming

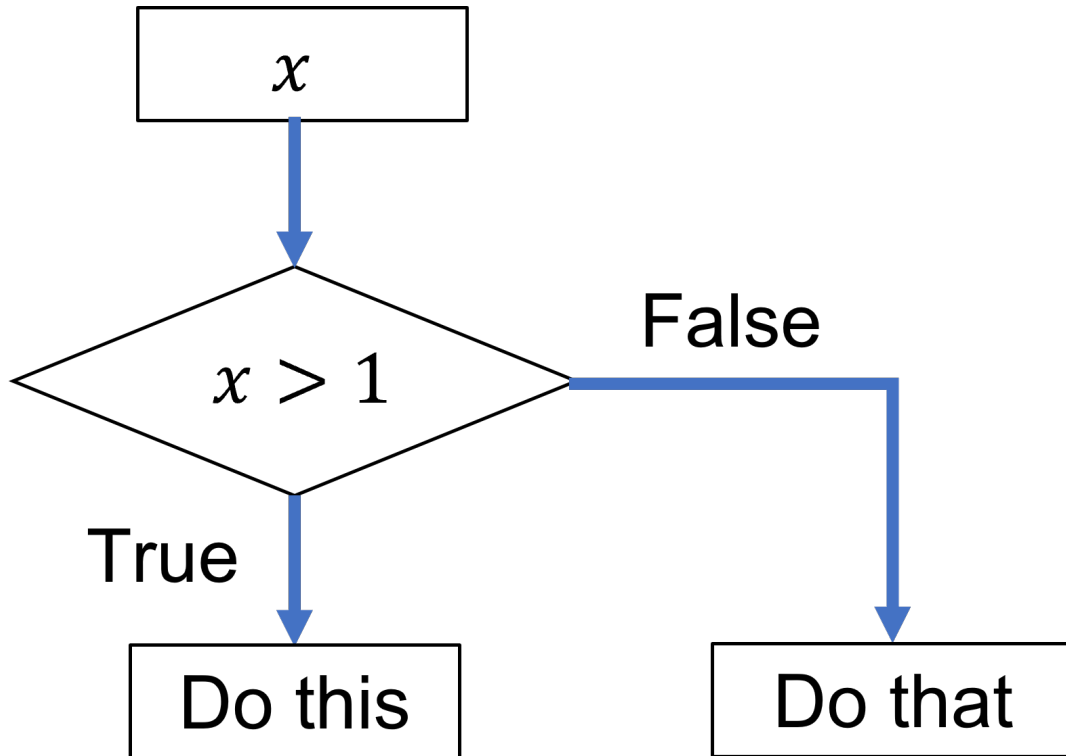
2.1 Control Flow



University of
BRISTOL

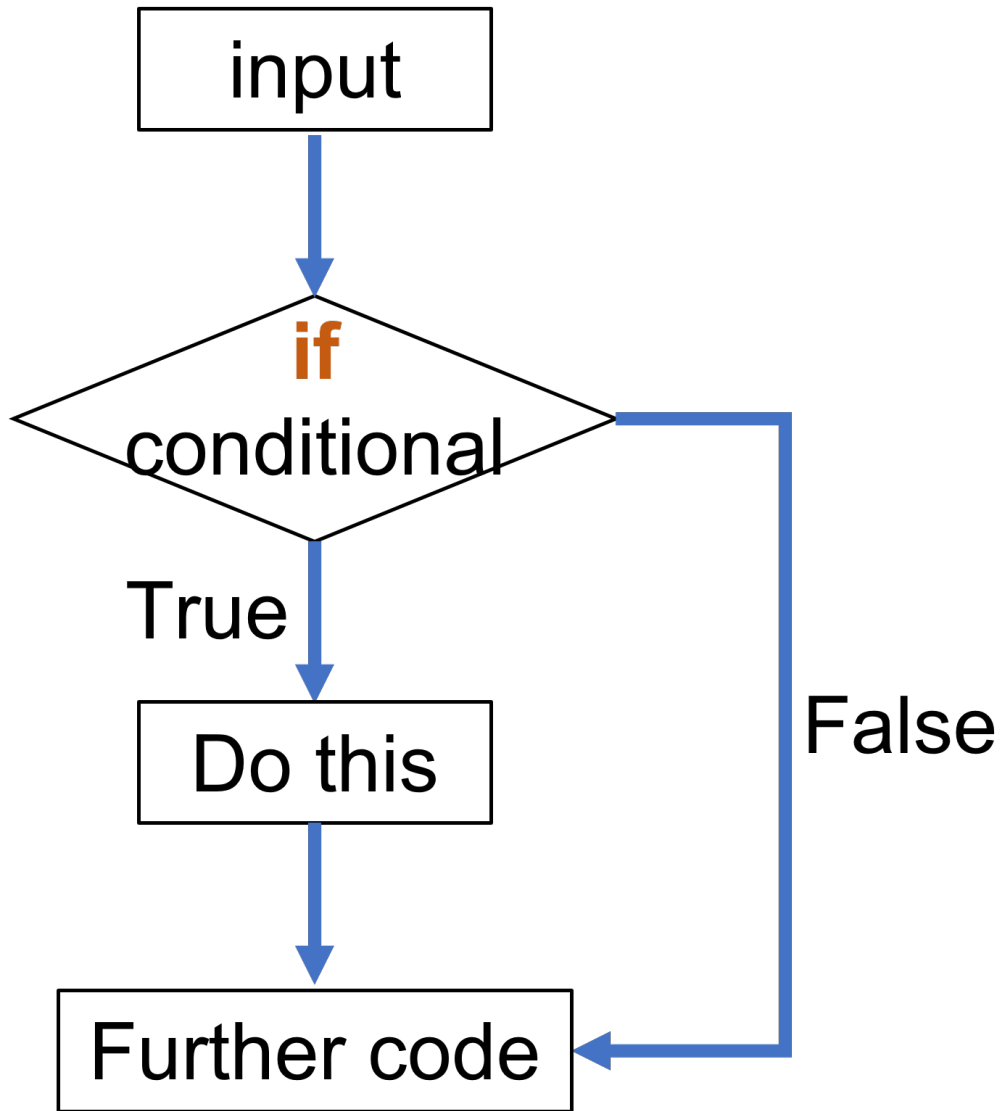
Conditional Statements

- Make decisions within a program and direct the flow.
- Run different blocks of code depending on whether a Boolean expression evaluates to True or False .
- This decision making is known as **Control Flow**



if

Runs a block of code only if a condition is True



```
In [1]:
```

```
x = 2

if x > 10:
    print("Do this") # block of code to run only if condition is True
print("Further code")
```

Further code

The colon `:` follows the condition to be evaluated.

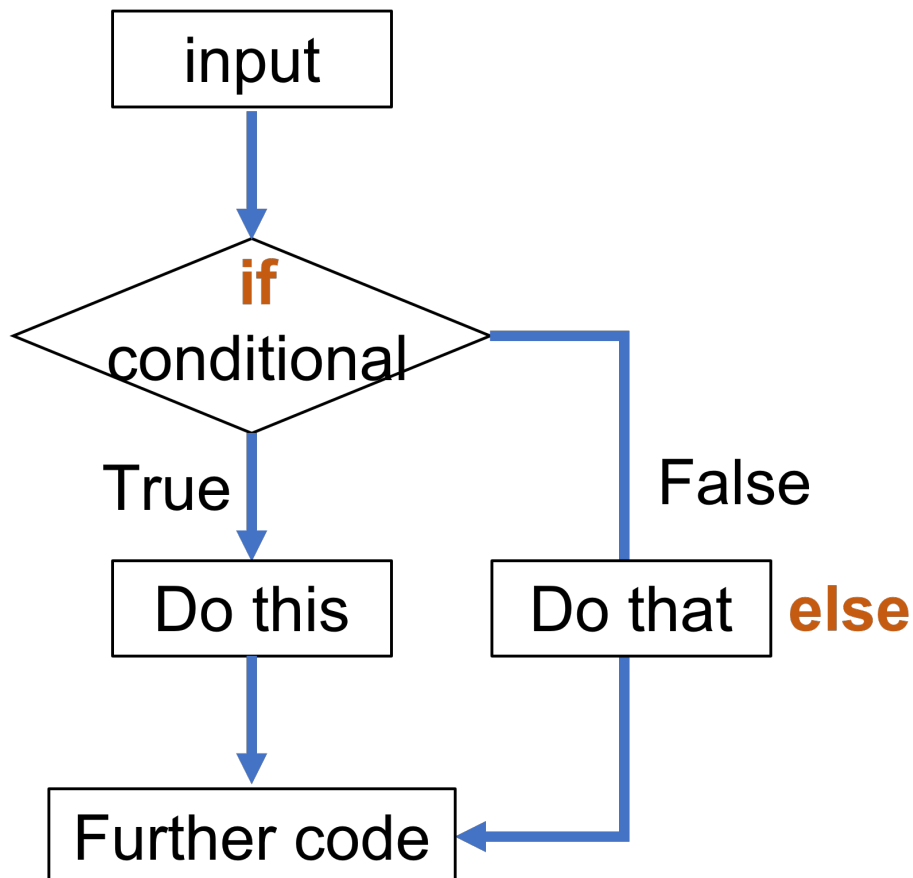
The indent is used to determine which pieces of code are executed in the case that the condition evaluates to `True` .

The indent can be any number of spaces.

- must be the same for all lines in a block of code.
- 4 spaces is considered best practise.
- Many IDEs (e.g. Spyder) automatically indent after you type `if:` .

if ... else

Runs a block of code only if a condition is `True`
Otherwise runs a different block of code.



In [3]:

```
x = 9

if x > 10:
    print("Do this") # if condition is True
else:
    print("Do that") # if condition is False

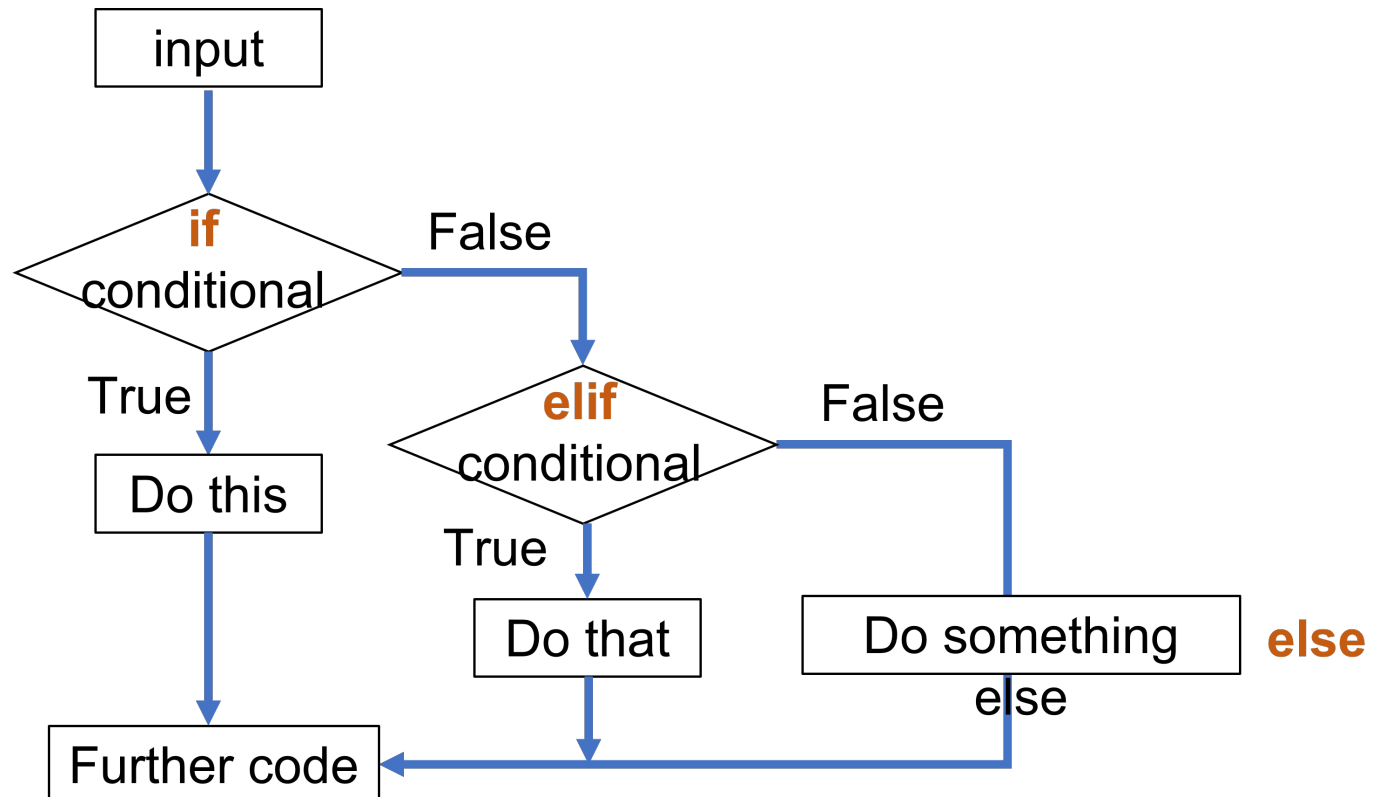
print("Further code")
```

Do that
Further code

if...elif...(else)

Runs the indented block of code after `if` if the `if` condition is `True`
... or runs the indented block of code after `elif` if the `elif` condition is `True`
... otherwise runs the indented block of code after `else` .

Only one of the three blocks is executed.



In [4]:

```
x = 4

if x > 10:
    print("Do this")           # if condition is True
elif x > 5:
    print("Do that")          # if another condition is True
else:
    print("Do something else") # if all preceding conditions are False
print("Further code")
```

Do something else
Further code

An unlimited number of `elif` statements can be used after an `if` statement

The `else` statement is optional.

In [8]:

```
x = -2

if x > 10:
    print("x is greater than 10") # if condition is True
elif x > 5:
    print("x is greater than 5") # if another condition is True
elif x > 0:
    print("x is greater than 0") # if another condition is True

print("Further code")
```

Further code

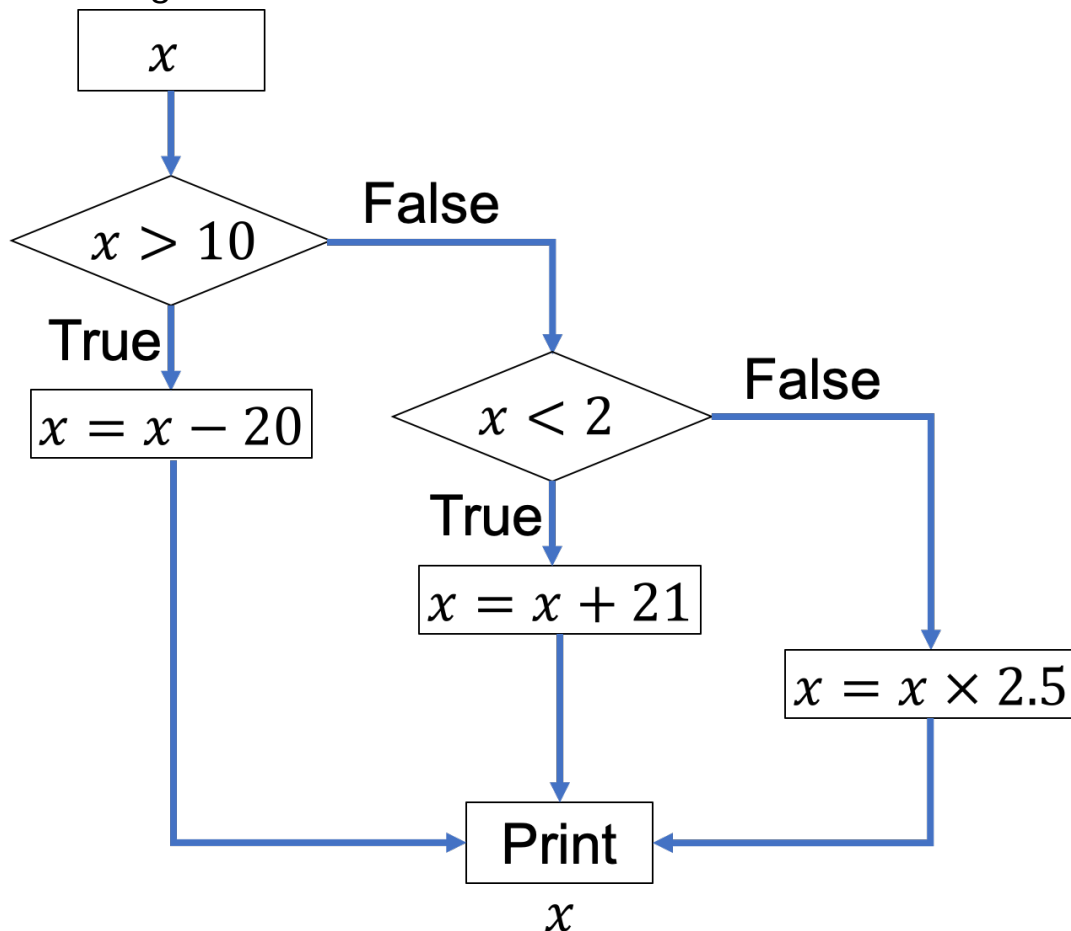
Summary

- Conditional statements (`if` , `elif` and `else`) perform a test on an expression with a Boolean (`True` or `False`) value.
- Execute/skip blocks of code based on the `True` / `False` value of the expression.

In-class Demos

Example 1:

Write a program to modify the initial value of the variable `x` and print the new value, as shown in the flow diagram.



In [13]:

```
# Initial value of x  
  
# x is greater than 10  
  
# x is less than 2  
  
# x is in range 2 to 10, inclusive  
  
# Final value of x
```

Let's remind ourselves of an example from last week.

Is it lunchtime?

True if time between lunch start and end times.
False if not.

Is it time for work?

True if time between work start and end times **and not** lunchtime.
False if not.

Let's build on the example from last week by including control statements.

Example 2:

Write a program that tells the user what activity to do based on the time of day.

- eat lunch if it is lunchtime
- do work if it is time for work
- go home if it is before or after work

In [12]:

```
# ----- Program from last week -----  
# Variables  
t = 18.00          # current time  
Ls = 13.00         # lunch starts  
Le = 14.00         # lunch ends  
Ws = 8.00          # work starts  
We = 17.00         # work ends  
  
lunchtime = Ls <= t < Le          # lunchtime  
work_time = Ws <= t < We and not lunchtime # work_time  
#-----
```

Go home

Coursework key dates

Assignment: Single piece of coursework, completed individually.

Theme: Write a program to perform an encryption and decryption task + a short (2 page) report

Set: Friday 19th November 2021 (Week 8)

Deadline: Friday 10th December 2021 (Week 11)

Drop-in support classes

On-campus Group 1	On-campus Group 2	Online
Friday	Friday	Thursday
13:00-14:00	14:00-15:00	9:00-10:00
MVB 1.15	MVB 1.15	remo

Group 1 : EMAT, Innovation, Biorobotics, everyone not listed in Groups 1/2

Group 2: EENG, EDES, Digital Health MSc & CDT

Introduction to Computer Programming

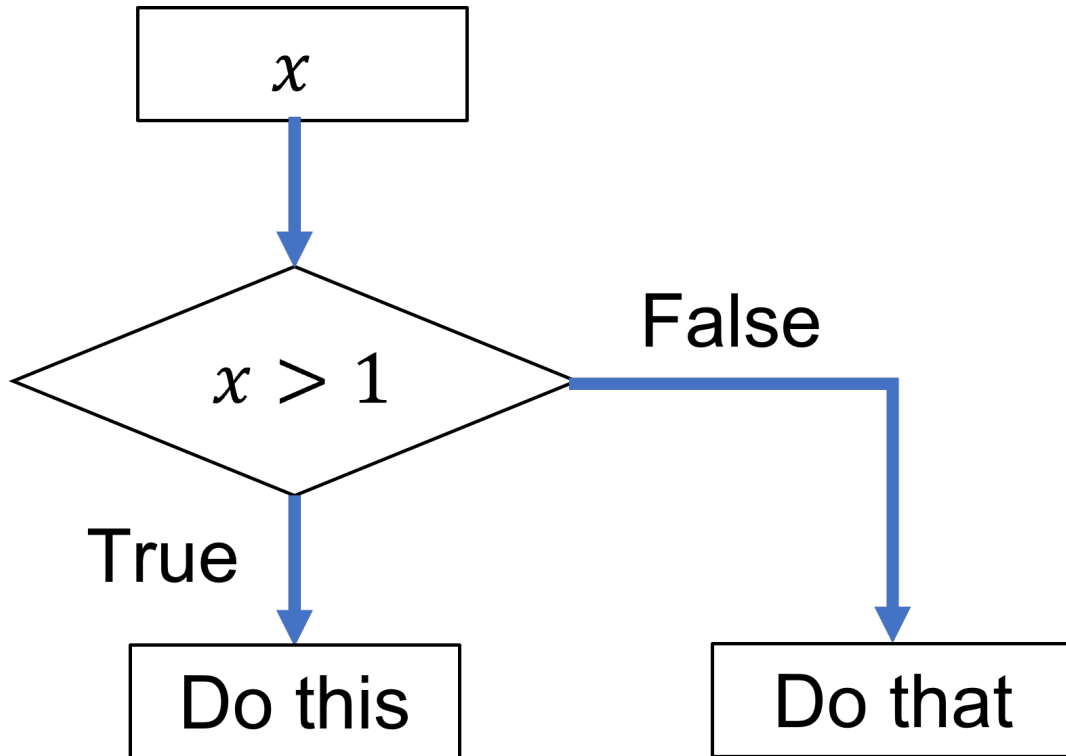
2.1 Control Flow



University of
BRISTOL

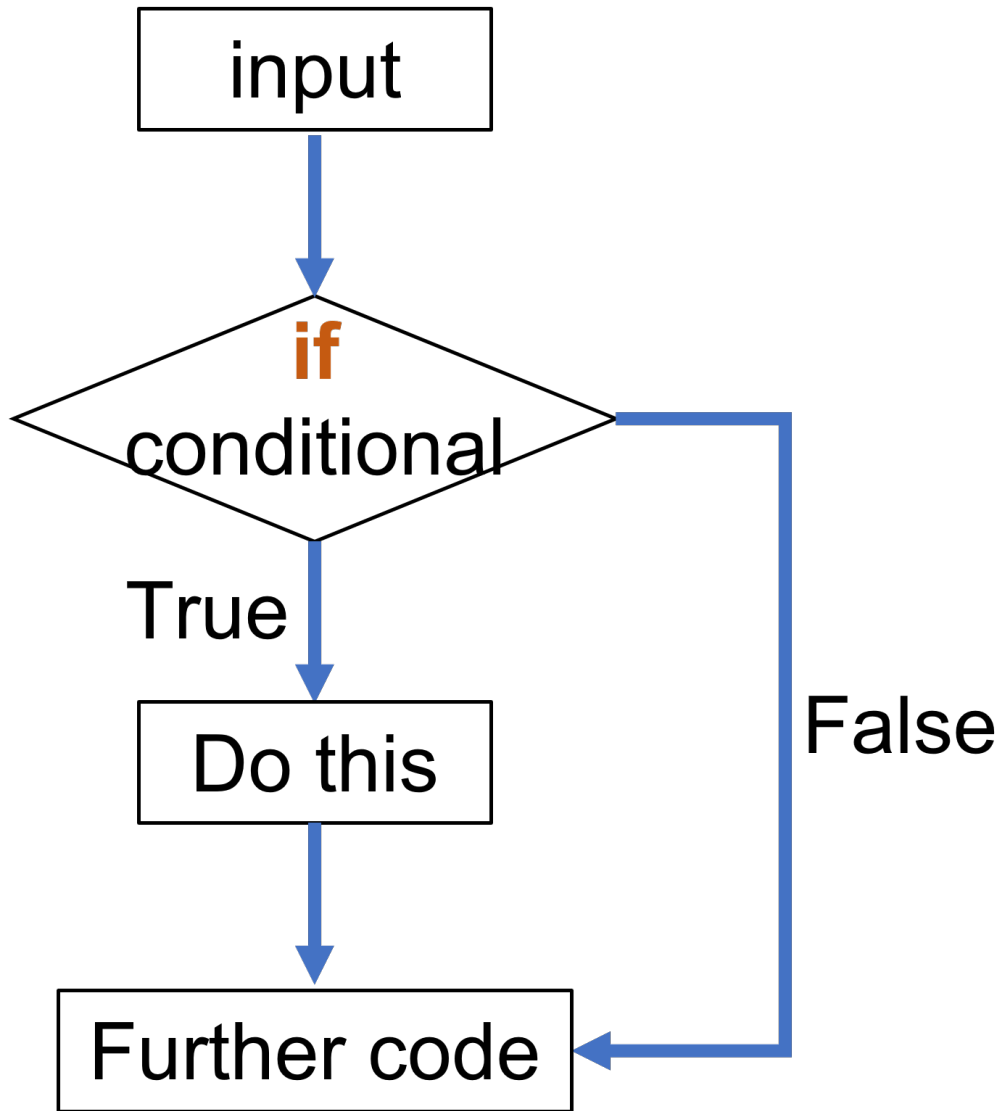
Conditional Statements

- Make decisions within a program and direct the flow.
- Run different blocks of code depending on whether a Boolean expression evaluates to True or False .
- This decision making is known as **Control Flow**



if

Runs a block of code only if a condition is True



In [1]:

Further code

The colon `:` follows the condition to be evaluated.

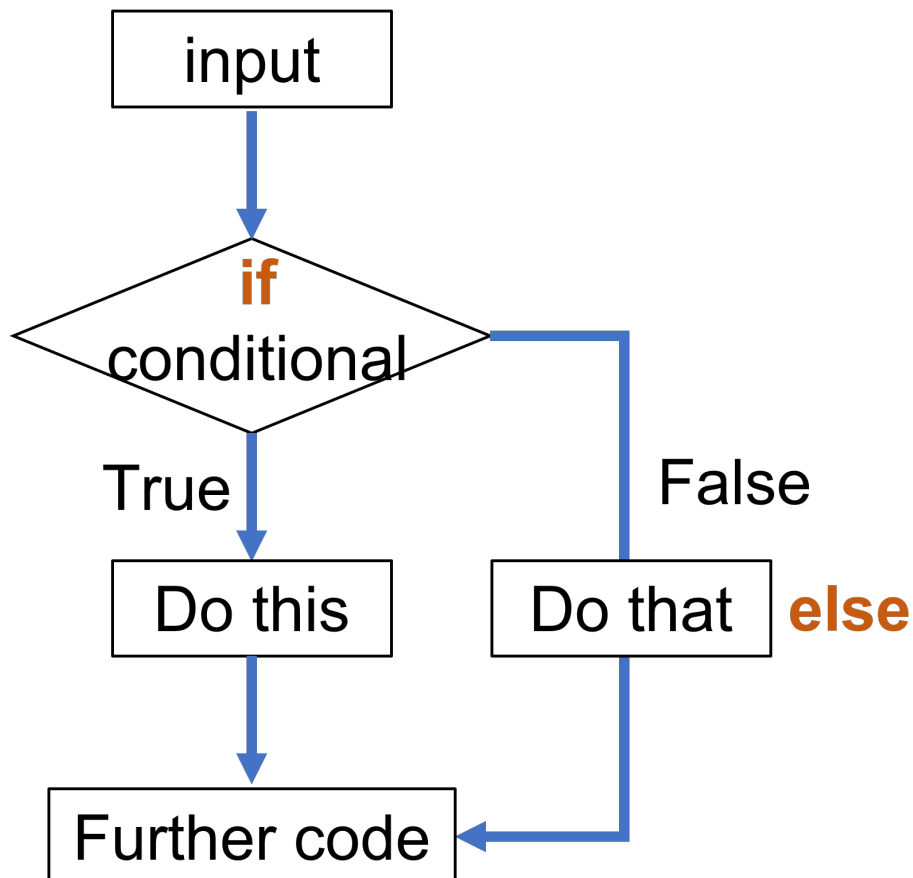
The indent is used to determine which pieces of code are executed in the case that the condition evaluates to `True` .

The indent can be any number of spaces.

- must be the same for all lines in a block of code.
- 4 spaces is considered best practise.
- Many IDEs (e.g. Spyder) automatically indent after you type `if:` .

if ... else

Runs a block of code only if a condition is `True`
Otherwise runs a different block of code.



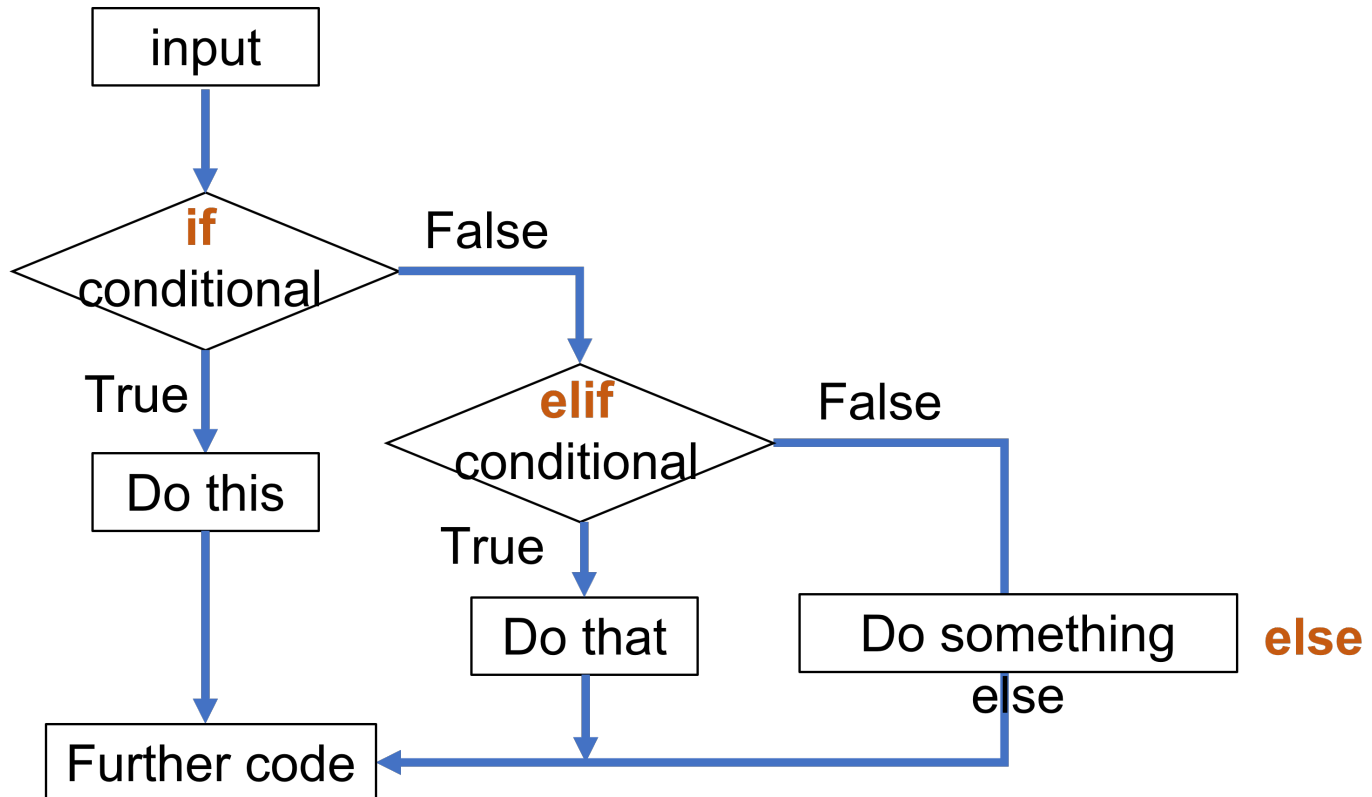
In [3]:

```
Do that
Further code
```

if...elif...(else)

Runs the indented block of code after `if` if the `if` condition is `True`
... or runs the indented block of code after `elif` if the `elif` condition is `True`
... otherwise runs the indented block of code after `else`.

Only one of the three blocks is executed.



In [4]:

```
Do something else
Further code
```

An unlimited number of `elif` statements can be used after an `if` statement

The `else` statement is optional.

In [8]:

Further code

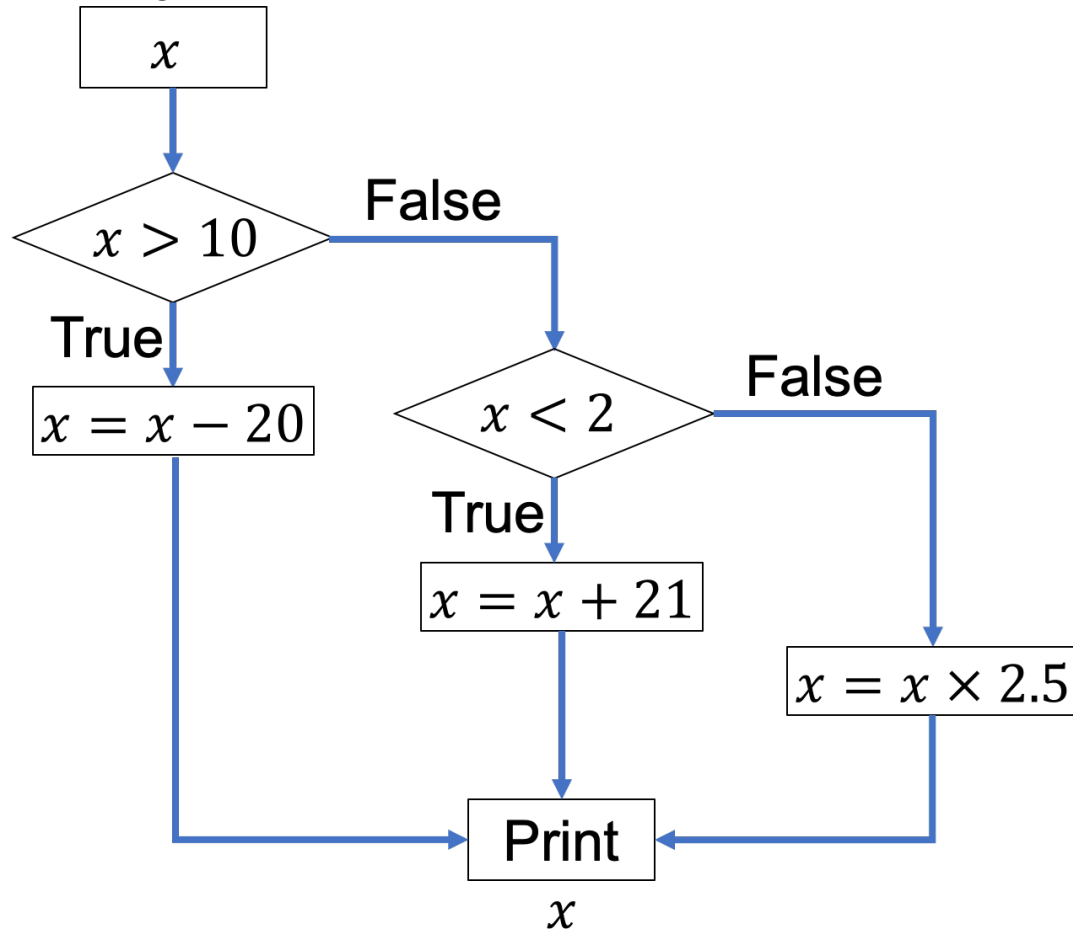
Summary

- Conditional statements (`if` , `elif` and `else`) perform a test on an expression with a Boolean (`True` or `False`) value.
- Execute/skip blocks of code based on the `True / False` value of the expression.

In-class Demos

Example 1:

Write a program to modify the initial value of the variable x and print the new value, as shown in the flow diagram.



In [13]:

Let's remind ourselves of an example from last week.

Is it lunchtime?

True if time between lunch start and end times.

False if not.

Is it time for work?

True if time between work start and end times **and not** lunchtime.

False if not.

Let's build on the example from last week by including control statements.

Example 2:

Write a program that tells the user what activity to do based on the time of day.

- eat lunch if it is lunchtime
- do work if it is time for work
- go home if it is before or after work

In [12]:

Go home