Introduction to Computer Programming

2.1 Control Flow

Announcement: Observation

- Helmut Hauser is observing my teaching in this class.
- You are not being observed.

Announcement: Change of IDE for home use

- Bug with curret version of Spyder
- Pycharm: IDE with more advanced features
- Versions: Community (free) or Professional (free when you create an account with university email https://www.jetbrains.com/pycharm/buy/#discounts (https://www.jetbrains.com/pycharm/buy/#discounts))
- Downloading installing and running:
  - Launches from Anaconda navigator (sign in with account created using university email to use Professional version)
  - Python download instructions for standalone installation: https://www.python.org/downloads/ (https://www.python.org/downloads/)
  - Standalone Pycharm installation instructions: https://www.jetbrains.com/pycharm/download/#section=mac (https://www.jetbrains.com/pycharm/download/#section=mac)
- 'First steps' instructions: https://www.jetbrains.com/help/pycharm/creating-and-running-your-first-python-project.html (https://www.jetbrains.com/help/pycharm/creating-and-running-your-first-python-project.html)
- IDLE (A very basic IDE) also downloads and installs with Python.

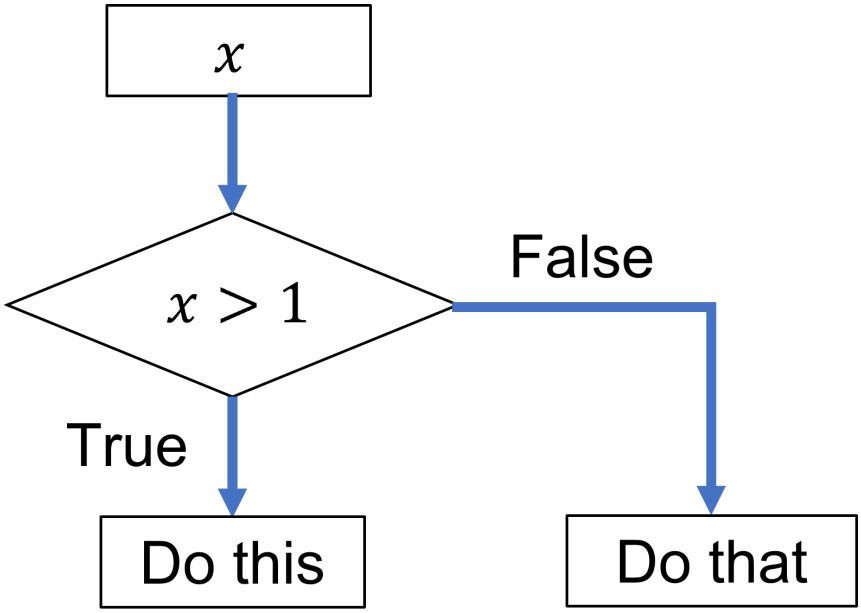
The goal of writing a computer program is to automate a process.

Throughout this course, we will study three fundamental topics that underpin automation:

- **Selection:** Decision-making
- **Repetition:** Repeatedly executing a process
- **Modularity:** Chunks of code that can be re-used

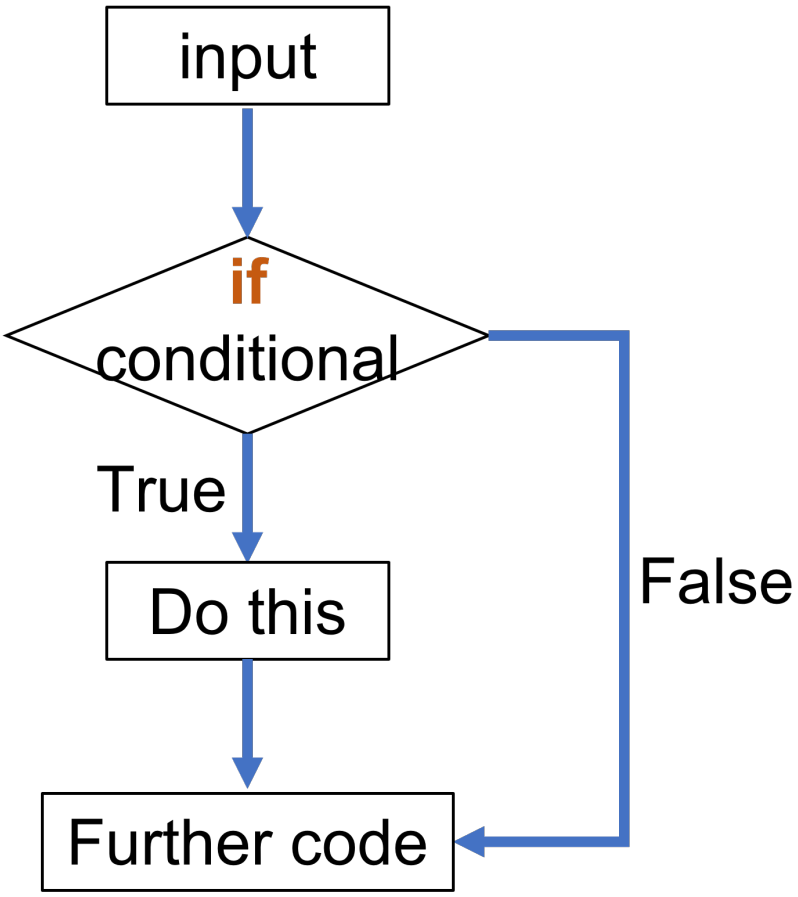
Conditional Statements

- The flow of a program is controlled by conditional statements.
- Conditional statements:
  - are used to make decisions within the program.
  - run different blocks of code depending on whether the Boolean value of an expression is `True` or `False`.
- This decision making is known as **Control Flow**



if

Runs a block of code only `if` the Boolean value of a condition is `True`



In [29]:

```
x = 2

if x > 1:

    print("Do this")

print("Further code")

```

```
Do this
Further code
```

The key ingredients are:

1. **The `if` keyword**
2. **The condition**: often includes *comparison*, *logical* or *identity* operators.
3. **The colon `:`** follows the condition to be evaluated.
4. **The *indented* block of code**: run if the Boolean value of the condition is `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 by 4 spaces after you type `if:`.

Example

Print two variables, `a` and `b` only if they are *both* greater than 10.

Hint: Use comparison and logical operators we studied last week.

In [ ]:
```
1
```

Best Practise for Code Layout - Blank lines and space

Using blank space between lines can improve the readability of your code.

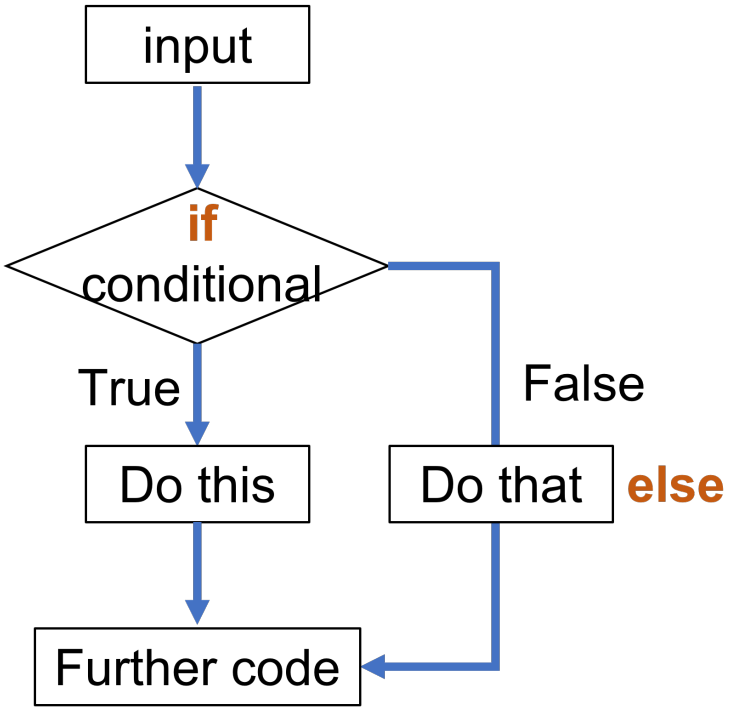
Code that's tightly packed together can be hard to read.

However too many blank lines make code sparse and slower to scroll through.

As a general rule, use blank lines to break your program into **clear steps**.
e.g. successive blocks of code beginning `if`, `elif` and `else`

`if… else`

- Runs a block of code only `if` the Boolean value of a condition is `True`
- Otherwise runs a *different* block of code



In [34]:
```
x = 12

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

else:
    print("Do that") # if condition is False

print("Further code")

```

```
Do this
Further code
```

Note:
Only one of the indented blocks of code (after `if` **or** after `else`) is executed!

Example

A digital thermostat checks the current temperature read by a sensor and compares it to a preset temperature.

The heating is switched:

- **ON** if temperature lower than preset temperature
- **OFF** if temperature higher than, or same as, preset temperature

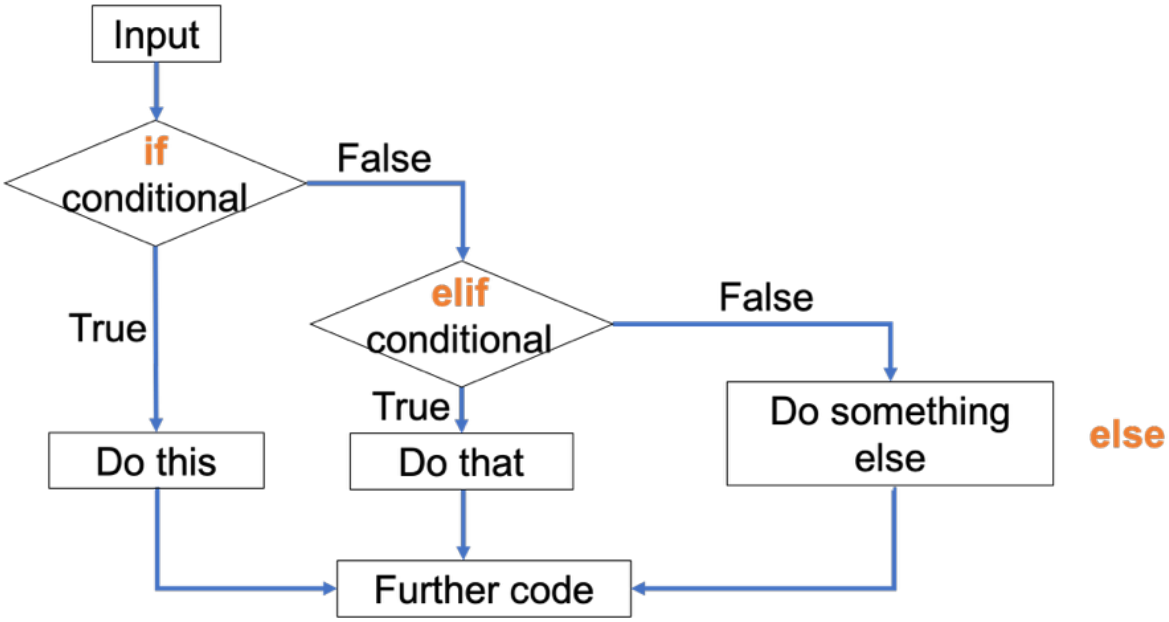
Write a program to simulate the behaviour of the digital thermostat, by using a variable `temp` to represent the current temperature.

In [ ]:
```
1
```

`if` … `elif` …(`else`)

- Runs indented code after `if` , if the Boolean value of a condition is `True` .
- Otherwise runs indented code after `elif` ('else if') if the Boolean value of a *different* condition is `True` .
- Otherwise runs indented code after `else` if *all* preceding `if` and `elif` statements output `False` .

Only one of the three blocks (after `if` **or** after `elif` **or** after `else`) is executed.



In [40]:
```
x = 12

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 ar

print("Further code")

```

```
Do this
Further code
```

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

The `else` statement is optional.

In [44]:
```
x = 12

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")

```

```
x is greater than 10
Further code
```

Example

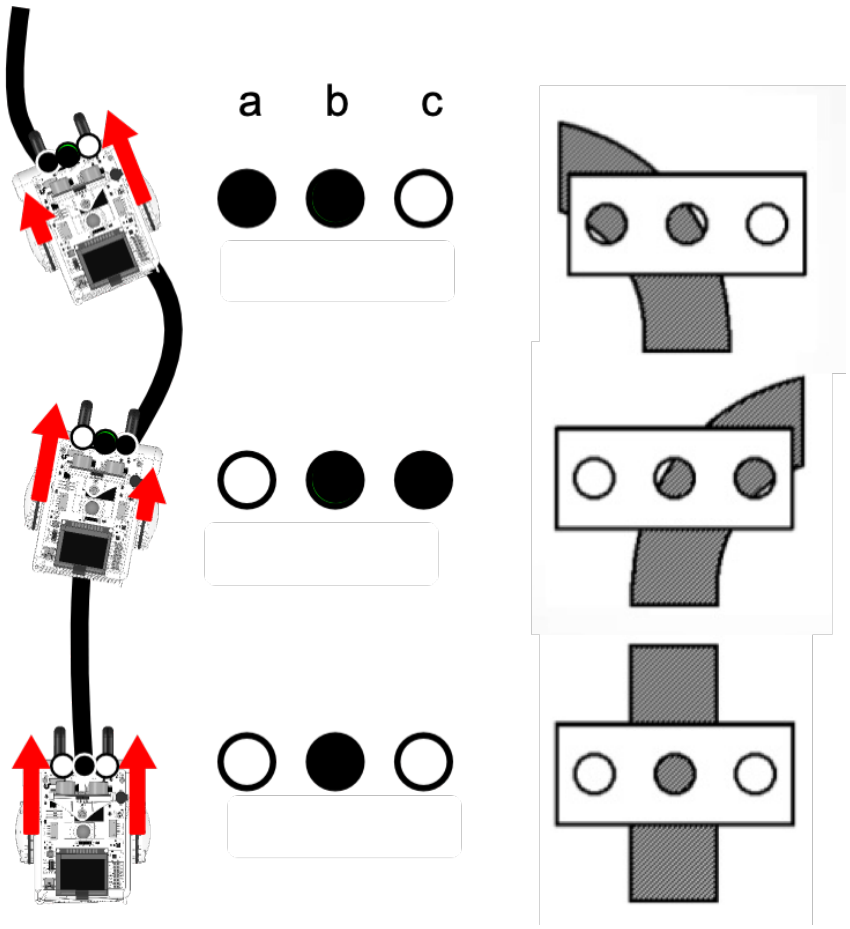
A two-wheeled robot uses three sensors (a, b and c) to follow a black line on a white surface.

The sensors output:

- 0 if over a white surface
- 1 if over a black surface

The arrows show what the relative speed of the two wheels should be, depending on the output of the sensors.

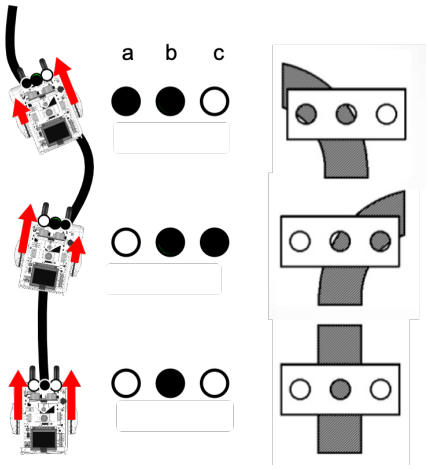
Write a program to simulate the robot's control system by representing the outputs of each sensors as a variable.



Hint: Remember `1` is `True` and `0` is `False`

In [ ]:
```

```



Summary

- Conditional statements (`if` , `elif` and `else`) perform a test on an expression with a Boolean (`True` or `False`) value.
- The program then executes or skips blocks of code based on the `True` / `False` output of the conditional statement.
- This is known as **Control Flow**.

Questions?