Welcome to the labs!

Cryptography!

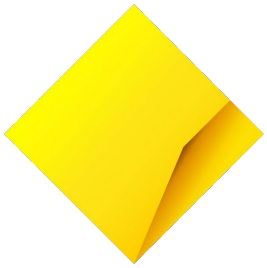
Thank you to our Sponsors!

Platinum Sponsor:

ATLASSIAN

amazon

Gold Sponsor:



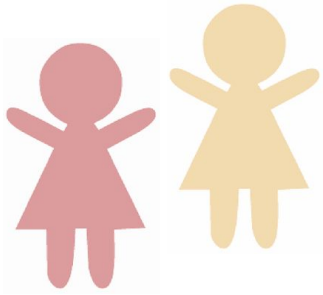
Commonwealth Bank

Who are the tutors?

Who are you?

Two Truths and a Lie

1. Get in a group of 3-5 people
2. Tell them three things about yourself:
   a. Two of these things should be true
   b. One of these things should be a lie!
3. The other group members have to guess which is the lie



Log on

Log on and jump on the GPN website

girlsprogramming.network/workshop

You can see:

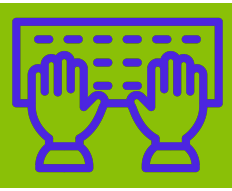
- These **slides** (to take a look back on or go on ahead).
- A link to the EdStem course
- Helpful bits of text you can **copy and paste**!

Tell us you're here!

Click on the
Start of Day Survey
and fill it in now!

Introduction to Edstem

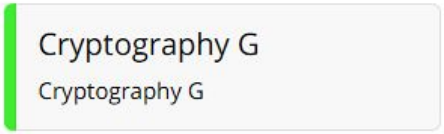
Signing up to Edstem



We are shifting all our courses to a new website called “Edstem” so here’s an overview of how to sign up and how to use it.

First let’s go through how to create an account.

1. Follow this link: https://edstem.org/au/join/qKyppB
2. Type in your name and your personal email address
3. Click Create Account
4. Go to your email to verify your account
5. Create a password
6. It should then take you to the courses home page.
7. Click on the one we will be using for this project: →



If you don’t have access to your email account, ask a tutor for a GPN edStem login

Getting to the lessons

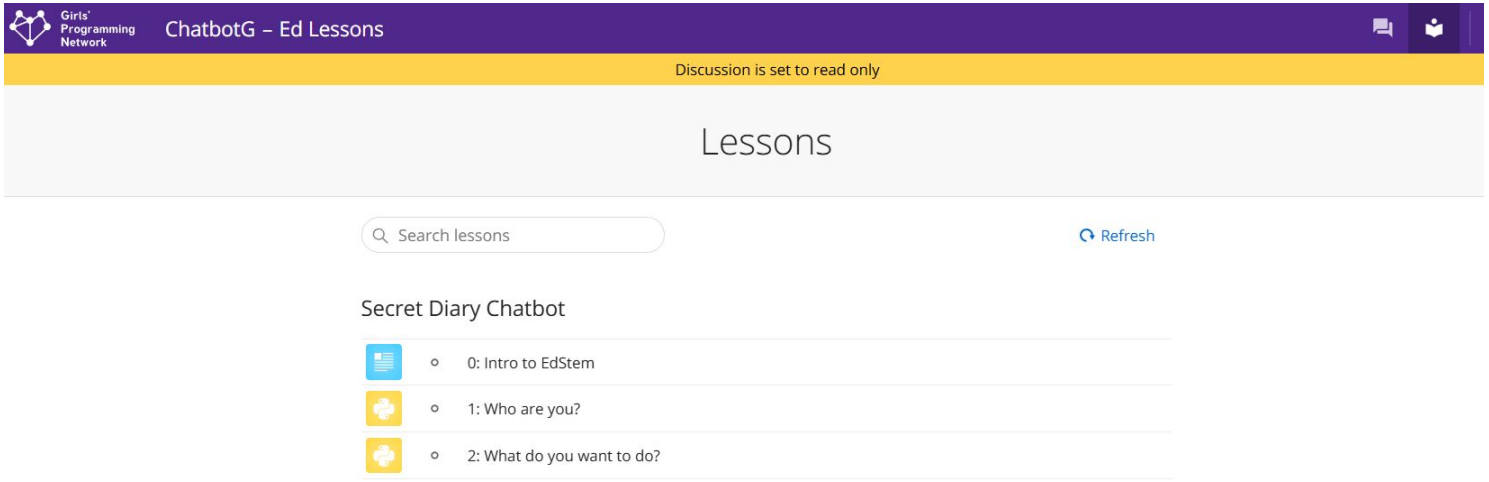
1. Once you are in the course, you'll be taken to a discussion page.
2. Click the button for the lessons page (top right - looks like a book)

The set up of the workbook

The main page:

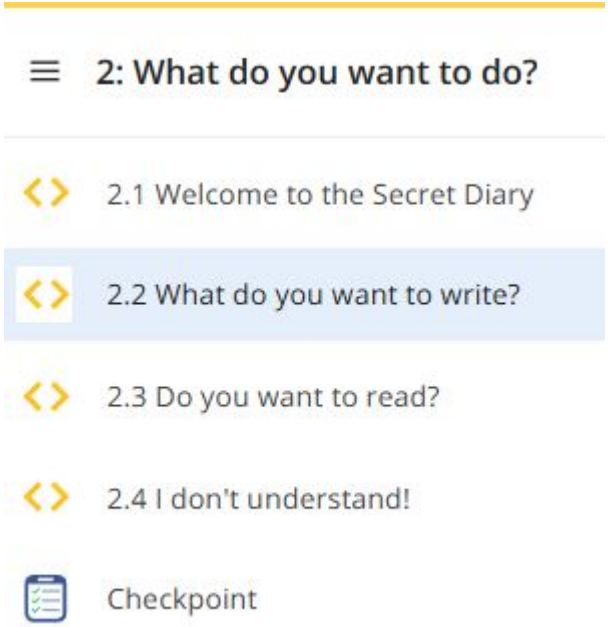
- Heading at the top that tells you the project you are in
- List of “Chapters” - They have an icon that looks like this:

- To complete your project, work through the chapters one at a time



Inside a Chapter

Inside a chapter there are two main types of pages:



1. **Lessons** - where you will do your coding.
   They have this icon: 

2. **Checkpoints**  Checkpoint

   Each chapter has a checkpoint to complete to move to the next chapter. Make sure you scroll down to see all the questions in a checkpoint.

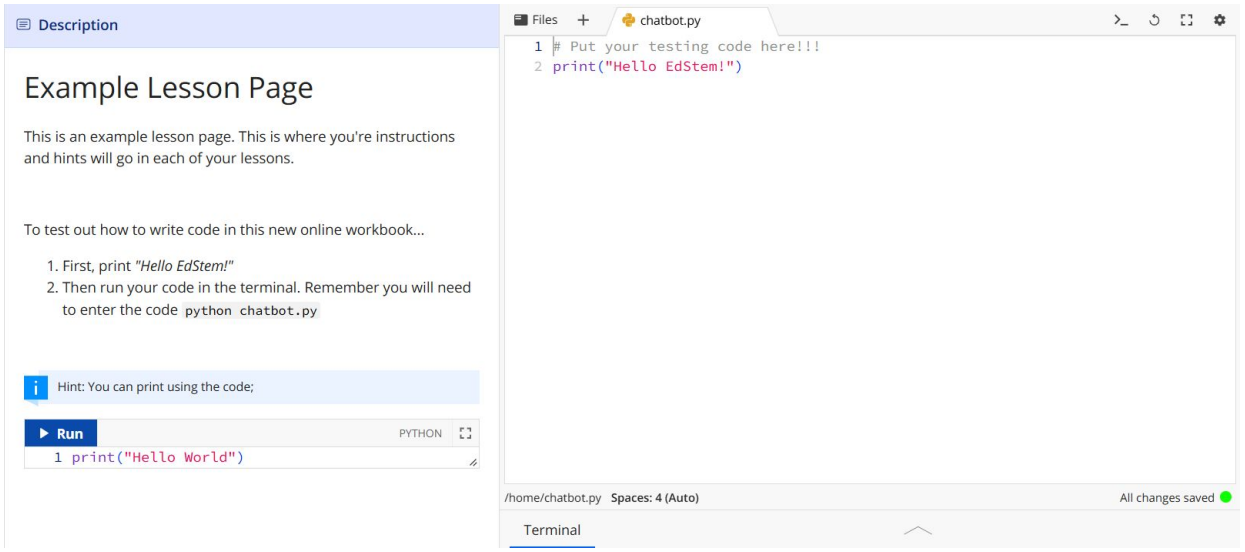
How to do the work

In each lesson there is:

1. A section on the left with instructions
2. A section on the right for your code

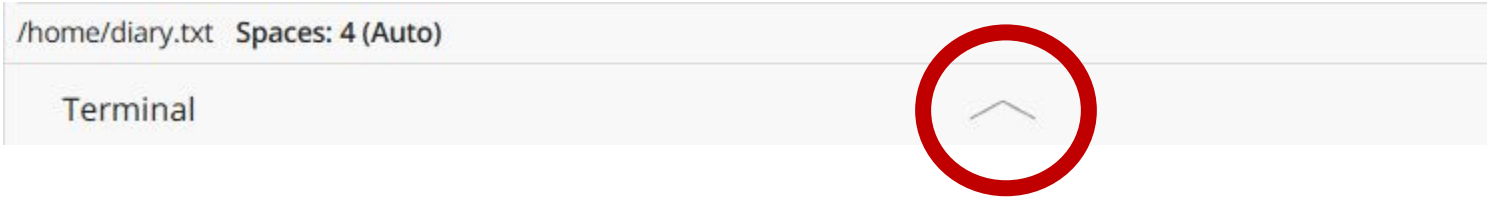
You will need to **copy your code from the last lesson**, then follow the instructions to change your code

There are also Hints and Code Blocks to help you



Running your code...

1. Open the Terminal window below your code



2. Click button that says “Click here to activate the terminal”.

3. Your code should run automatically.
4. Click the button again to rerun your code.
5. You can resize the Terminal window.

Don’t worry if you forget. Tutors will help!

Some shortcuts...

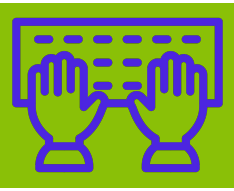
There are a couple things you can do to make copying your code from one page to another easier.

1) **Ctrl + A** Pressing these keys together will select all the text on a page

2) **Ctrl + C** Pressing these keys together will copy anything that's selected

3) **Ctrl + V** Pressing these keys together will paste anything you've copied

Need help with EdStem?

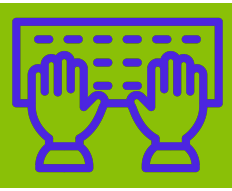


There is a section at the top of your workbook that explains how to use EdStem if you get stuck and need a reminder!

It's called 0: Intro to EdStem

Go to Part 0 and have a look!

Project time!



You now know all about EdStem!

You should now sign up and join our EdStem class. You should also have a look at part 0 of your workbook

Remember the tutors will be around to help!

Intro to Caesar Ciphers

Let’s get encrypting!

What is a cipher?

A cipher is a way to write a message so that no one else can read it!

Unless they know the secret key!

Examples of ciphers

If you've ever made up your own secret language or made notes to your friends so that other people can't read them, you've made a cipher!

For example:

gnidoc evol i

Can you figure out what this says?

Examples of ciphers

If you've ever made up your own secret language or made notes to your friends so that other people can't read them, you've made a cipher!

For example:

gnidoc evol i

Can you figure out what this says?

It says **I love coding** backwards!

Caesar Cipher

So what's a Caesar Cipher?

It's a cypher that Julius Caesar used in ancient Rome to send secret messages to his armies!

Let's learn how it works!

Make a Cipher Wheel

- Cut out green circle
- Cut out purple circle
- Put small circle on top of big circle matching centres
- Secure together with centre split pin
- Spin inside circle of letters around

Caesar Cipher Wheel template in Workshop Material folder

Shifting letters

A Caesar Cipher works by shifting letters in the alphabet so that they line up with new letters.

For example if we were to shift everything by 3 it would look like this:

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c

Line up the ‘a’ on both wheels and then turn the inside wheel 3 letters **anti-clockwise** so that you have your letters lining up like this!

Encrypting

Now, let's encrypt **I love coding** using the wheel

For our Caesar Cipher we take each letter and replace it with the 'shifted' letter

So, let's start with the letter 'i'

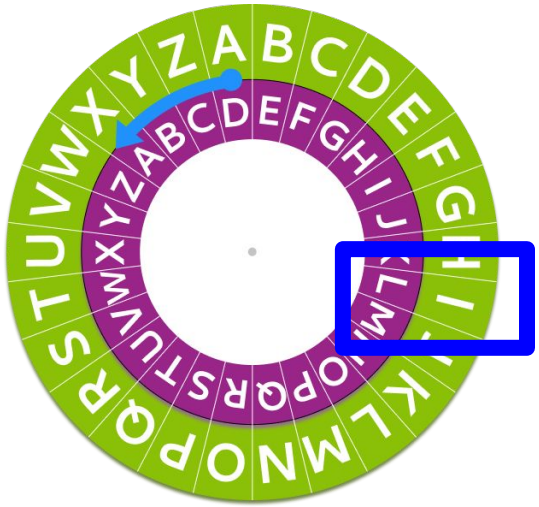
What new letter should we use to replace it?

>>> *Find letter i on the **outside** wheel and replace it with it's matching letter on the **inside** wheel* = the letter 'l'

Encrypting

Now, let's encrypt **I love coding** using the wheel

For our Caesar Cipher we take each letter and replace it with the 'shifted' letter



So, let's start with the letter 'i'

What new letter should we use to replace it?

>>> *Find letter i on the **outside** wheel and replace it with it's matching letter on the **inside** wheel* = the letter 'l'

Writing the whole message!

Let's do the rest of the message together

I love coding

I	Is replaced with	
o	Is replaced with	
v	Is replaced with	
e	Is replaced with	
c	Is replaced with	
o	Is replaced with	
d	Is replaced with	
i	Is replaced with	
n	Is replaced with	
g	Is replaced with	

Writing the whole message!

Let's do the rest of the message together

I love coding

l	Is replaced with	o
o	Is replaced with	
v	Is replaced with	
e	Is replaced with	
c	Is replaced with	
o	Is replaced with	
d	Is replaced with	
i	Is replaced with	
n	Is replaced with	
g	Is replaced with	

Writing the whole message!

Let's do the rest of the message together

I love coding

l	Is replaced with	o
o	Is replaced with	r
v	Is replaced with	
e	Is replaced with	
c	Is replaced with	
o	Is replaced with	
d	Is replaced with	
i	Is replaced with	
n	Is replaced with	
g	Is replaced with	

Writing the whole message!

Let's do the rest of the message together

I love coding

l	Is replaced with	o
o	Is replaced with	r
v	Is replaced with	y
e	Is replaced with	
c	Is replaced with	
o	Is replaced with	
d	Is replaced with	
i	Is replaced with	
n	Is replaced with	
g	Is replaced with	

Writing the whole message!

Let's do the rest of the message together

I love coding

l	Is replaced with	o
o	Is replaced with	r
v	Is replaced with	y
e	Is replaced with	h
c	Is replaced with	
o	Is replaced with	
d	Is replaced with	
i	Is replaced with	
n	Is replaced with	
g	Is replaced with	

Writing the whole message!

Let's do the rest of the message together

I love coding

l	Is replaced with	o
o	Is replaced with	r
v	Is replaced with	y
e	Is replaced with	h
c	Is replaced with	f
o	Is replaced with	
d	Is replaced with	
i	Is replaced with	
n	Is replaced with	
g	Is replaced with	

Writing the whole message!

Let's do the rest of the message together

I love coding

I	Is replaced with	o
o	Is replaced with	r
v	Is replaced with	y
e	Is replaced with	h
c	Is replaced with	f
o	Is replaced with	r
d	Is replaced with	
i	Is replaced with	
n	Is replaced with	
g	Is replaced with	

Writing the whole message!

Let's do the rest of the message together

I love coding

I	Is replaced with	o
o	Is replaced with	r
v	Is replaced with	y
e	Is replaced with	h
c	Is replaced with	f
o	Is replaced with	r
d	Is replaced with	g
i	Is replaced with	
n	Is replaced with	
g	Is replaced with	

Writing the whole message!

Let's do the rest of the message together

I love coding

I	Is replaced with	o
o	Is replaced with	r
v	Is replaced with	y
e	Is replaced with	h
c	Is replaced with	f
o	Is replaced with	r
d	Is replaced with	g
i	Is replaced with	l
n	Is replaced with	
g	Is replaced with	

Writing the whole message!

Let's do the rest of the message together

I love coding

l	Is replaced with	o
o	Is replaced with	r
v	Is replaced with	y
e	Is replaced with	h
c	Is replaced with	f
o	Is replaced with	r
d	Is replaced with	g
i	Is replaced with	l
n	Is replaced with	q
g	Is replaced with	

Writing the whole message!

Let's do the rest of the message together

I love coding

l	Is replaced with	o
o	Is replaced with	r
v	Is replaced with	y
e	Is replaced with	h
c	Is replaced with	f
o	Is replaced with	r
d	Is replaced with	g
i	Is replaced with	l
n	Is replaced with	q
g	Is replaced with	j

Secret Message

So our secret encrypted message is

L oryh frglqj

That's a lot harder to figure out than it just being backwards!

Encrypt your own name!

Using a key of minus 1 (so A=Z) (Jessica = Idrrhbz)

Write your name on the blank tag in name badge!

Decrypting

Writing secret messages isn't any fun if you can't figure out what they say!

Luckily you can also use your cipher wheel to *decrypt* a secret message.

How do you think we can do that?

What information do we need to know in order to decrypt a secret message?

It’s the key!

To decrypt a secret message **we need to know** the amount that we shifted the wheel when we encrypted it. That number is called **the key**!

Once we know the key we can just turn our wheel and read the wheel from the inside out!

*Find the letter on the **inside** wheel and replace it with it’s matching letter on the **outside** wheel*

Let’s check it works!

l	Is replaced with	
o	Is replaced with	
r	Is replaced with	
y	Is replaced with	
h	Is replaced with	
f	Is replaced with	
r	Is replaced with	
g	Is replaced with	
l	Is replaced with	
q	Is replaced with	
j	Is replaced with	

Let’s check it works!

l	Is replaced with	i
o	Is replaced with	
r	Is replaced with	
y	Is replaced with	
h	Is replaced with	
f	Is replaced with	
r	Is replaced with	
g	Is replaced with	
l	Is replaced with	
q	Is replaced with	
j	Is replaced with	

Let’s check it works!

l	Is replaced with	i
o	Is replaced with	l
r	Is replaced with	
y	Is replaced with	
h	Is replaced with	
f	Is replaced with	
r	Is replaced with	
g	Is replaced with	
l	Is replaced with	
q	Is replaced with	
j	Is replaced with	

Let’s check it works!

l	Is replaced with	i
o	Is replaced with	l
r	Is replaced with	o
y	Is replaced with	
h	Is replaced with	
f	Is replaced with	
r	Is replaced with	
g	Is replaced with	
l	Is replaced with	
q	Is replaced with	
j	Is replaced with	

Let’s check it works!

l	Is replaced with	i
o	Is replaced with	l
r	Is replaced with	o
y	Is replaced with	v
h	Is replaced with	
f	Is replaced with	
r	Is replaced with	
g	Is replaced with	
l	Is replaced with	
q	Is replaced with	
j	Is replaced with	

Let’s check it works!

l	Is replaced with	i
o	Is replaced with	l
r	Is replaced with	o
y	Is replaced with	v
h	Is replaced with	e
f	Is replaced with	
r	Is replaced with	
g	Is replaced with	
l	Is replaced with	
q	Is replaced with	
j	Is replaced with	

Let’s check it works!

l	Is replaced with	i
o	Is replaced with	l
r	Is replaced with	o
y	Is replaced with	v
h	Is replaced with	e
f	Is replaced with	c
r	Is replaced with	
g	Is replaced with	
l	Is replaced with	
q	Is replaced with	
j	Is replaced with	

Let's check it works!

l	Is replaced with	i
o	Is replaced with	l
r	Is replaced with	o
y	Is replaced with	v
h	Is replaced with	e
f	Is replaced with	c
r	Is replaced with	o
g	Is replaced with	
l	Is replaced with	
q	Is replaced with	
j	Is replaced with	

Let’s check it works!

l	Is replaced with	i
o	Is replaced with	l
r	Is replaced with	o
y	Is replaced with	v
h	Is replaced with	e
f	Is replaced with	c
r	Is replaced with	o
g	Is replaced with	d
l	Is replaced with	
q	Is replaced with	
j	Is replaced with	

Let’s check it works!

l	Is replaced with	i
o	Is replaced with	l
r	Is replaced with	o
y	Is replaced with	v
h	Is replaced with	e
f	Is replaced with	c
r	Is replaced with	o
g	Is replaced with	d
l	Is replaced with	i
q	Is replaced with	
j	Is replaced with	

Let’s check it works!

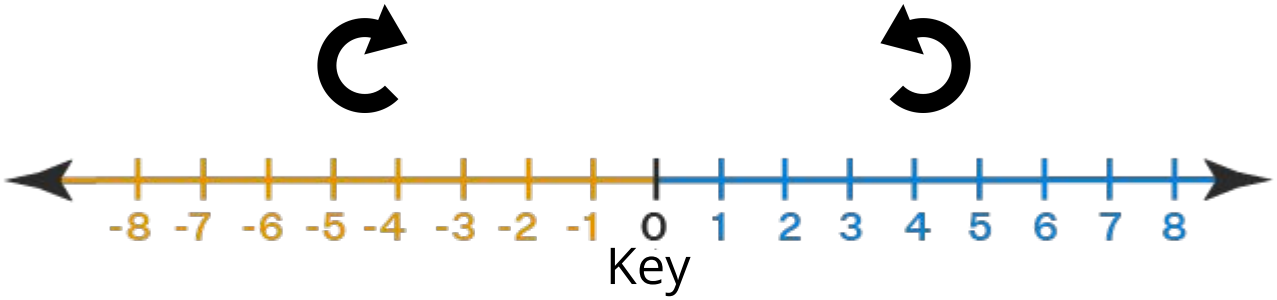
l	Is replaced with	i
o	Is replaced with	l
r	Is replaced with	o
y	Is replaced with	v
h	Is replaced with	e
f	Is replaced with	c
r	Is replaced with	o
g	Is replaced with	d
l	Is replaced with	i
q	Is replaced with	n
j	Is replaced with	

Let’s check it works!

l	Is replaced with	i
o	Is replaced with	l
r	Is replaced with	o
y	Is replaced with	v
h	Is replaced with	e
f	Is replaced with	c
r	Is replaced with	o
g	Is replaced with	d
l	Is replaced with	i
q	Is replaced with	n
j	Is replaced with	g

Another way to decrypt

- Another way to decrypt a message is to change the key value to become the negative of the encryption key value
- We will use this method in our code
- This is because to decrypt a message we need to shift the alphabet the opposite way.
- A negative key value means you turn your inner purple wheel to the right (clockwise)



Your Turn!

Try doing Lesson 1
using your Caesar Cipher wheels!

Your tutors are here to help you if you get stuck

Intro to Programming

What is programming?

Programming is not a bunch of crazy numbers!

It's giving computers a set of instructions!

A Special Language

A language to talk to dogs!

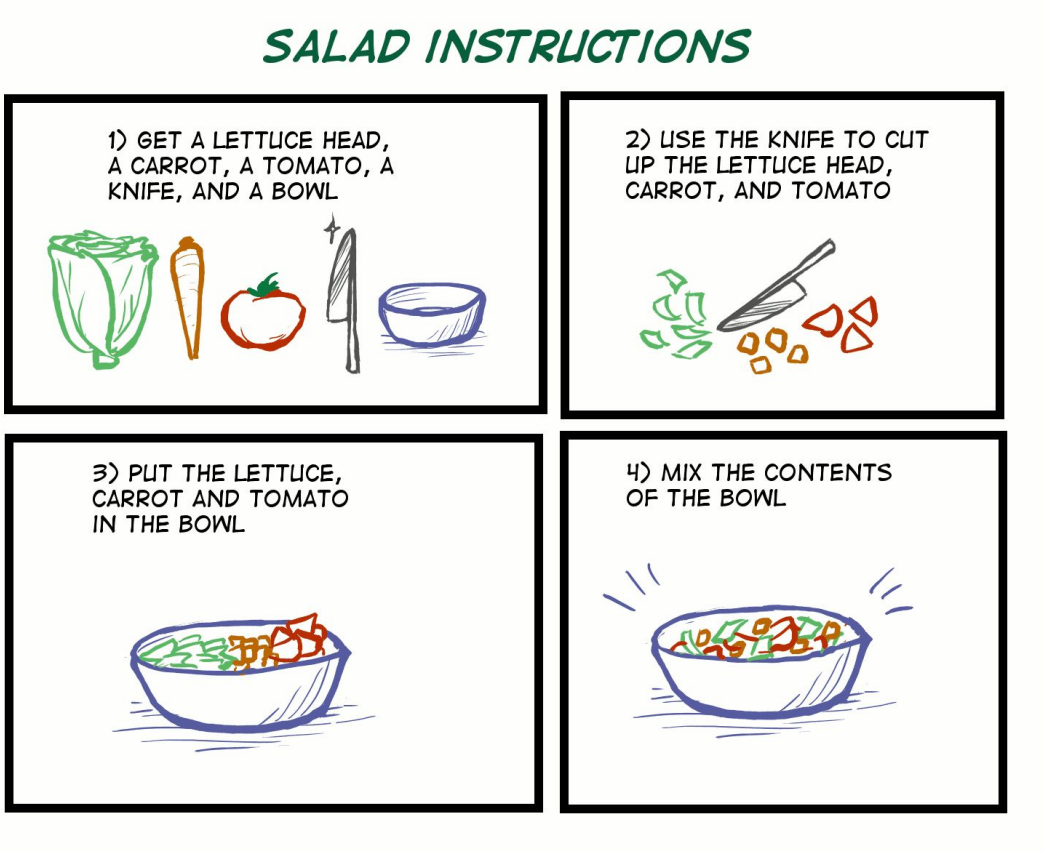
Programming is a language to talk to computers

People are smart! Computers are dumb!

Programming is creating a set of instructions, like a recipe.

Computers do EXACTLY what you say, every time.

Which is great if you give them a good recipe!

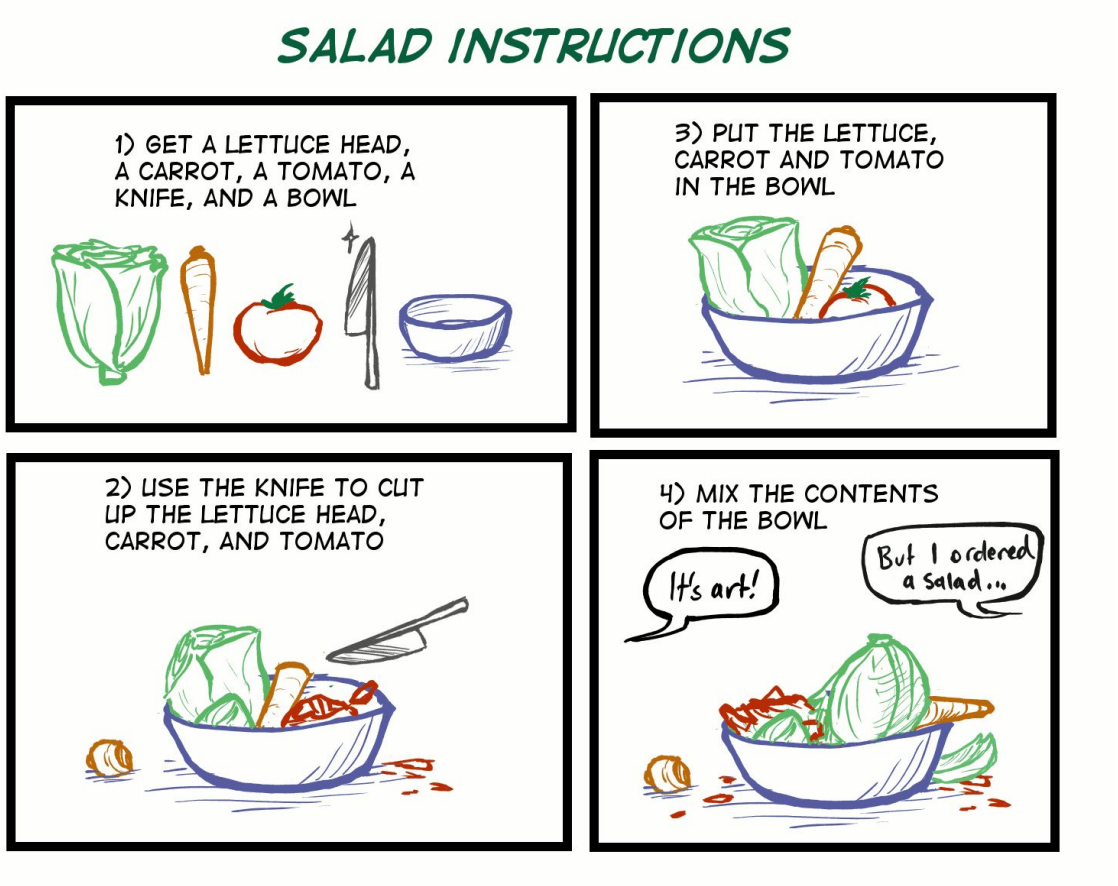


People are smart! Computers are dumb!

But if your recipe is wrong e.g. get it out of order….

A computer wouldn't know this recipe was wrong.

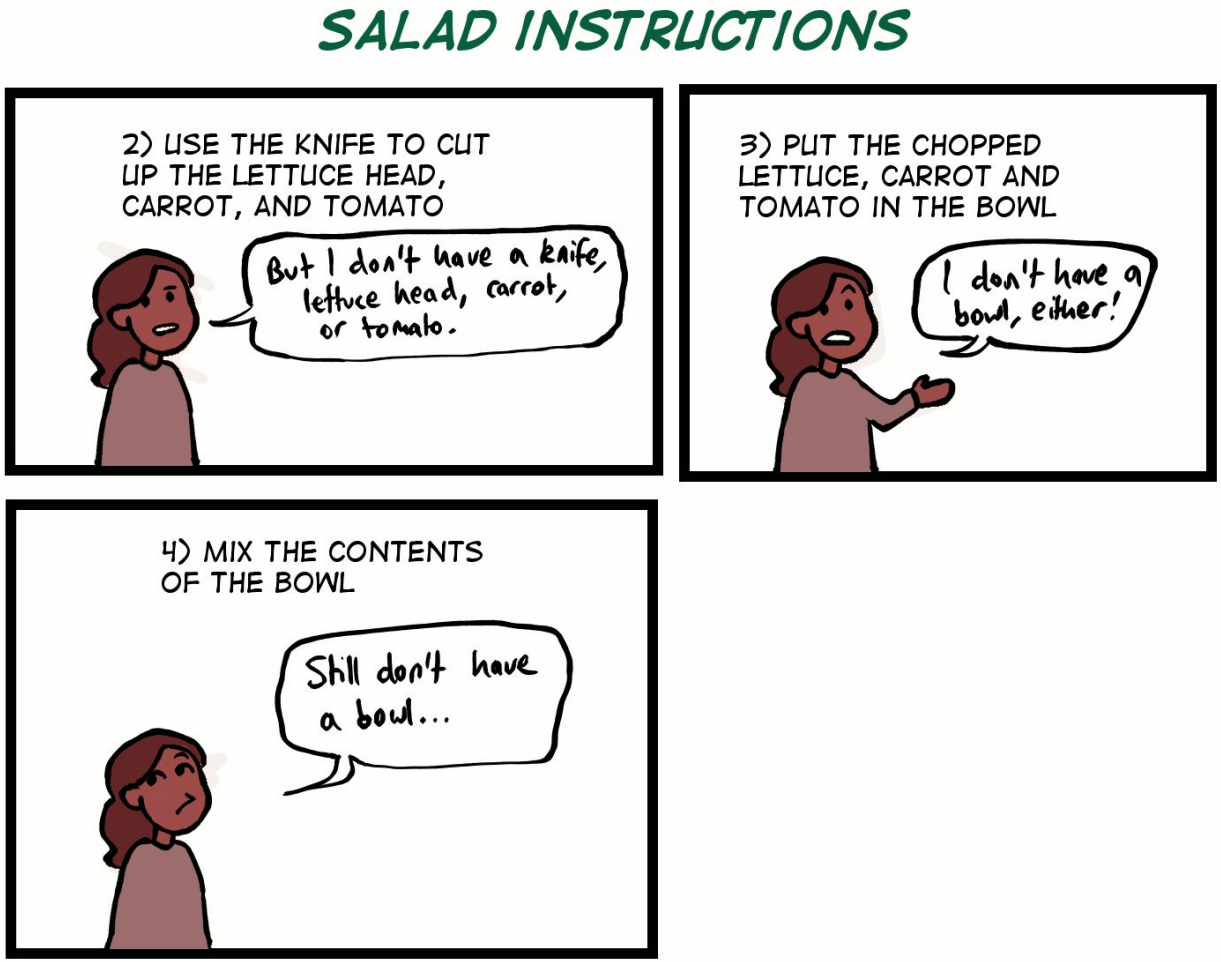
It would still try to make it anyway!



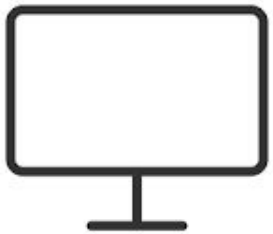
People are smart! Computers are dumb!

Computers are bad at filling in the gaps!

A computer wouldn't know something was missing, it would just freak out!



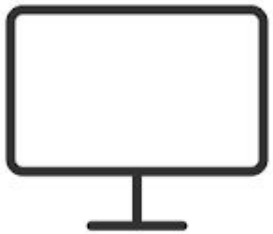
Everyone & Everything has strengths!



How is the human brain different from a computer's brain?

Everyone & Everything has strengths!

- Understand instructions very well despite spelling mistakes or typos
- Solve hard problems
- Invent computers and tell them what to do!
- Get smarter by learning

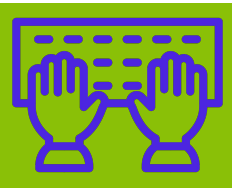


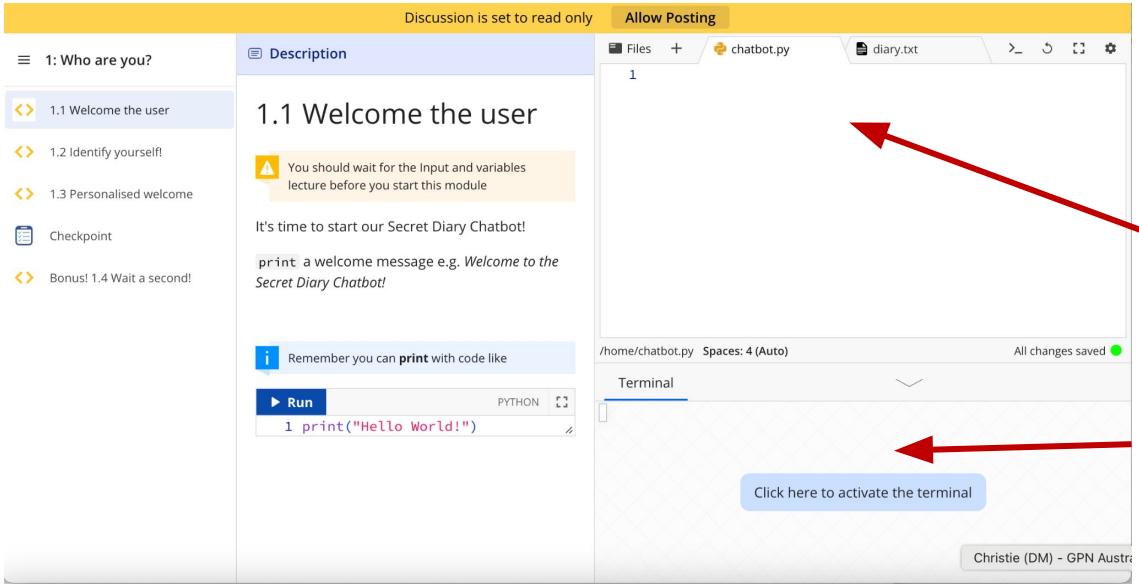
- Only does exactly what humans tell it
- Does it the same way every time
- Will work endlessly
- Really good at being repetitive
- REALLY fast
- Get smarter when humans tell it how

Intro to Python

Let’s get coding!

Let's make a mistake!





1. Type by **button mashing** the keyboard here e.g.

   `ks@674dbkjSDfkl`

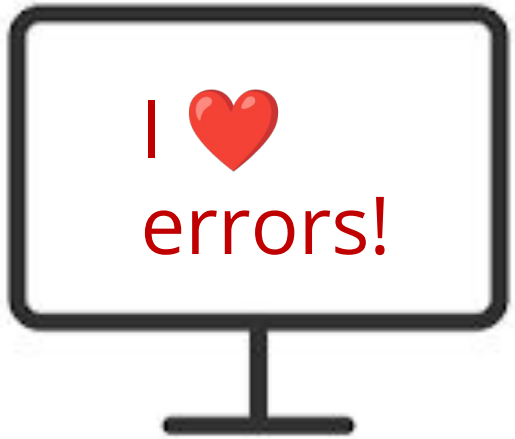
2. **Click** in the Terminal panel here to run your code!

Did you get a big ugly error message?

Mistakes are great!

Good work! You made an error!

- Programmers make A LOT of errors!
- Errors give us hints on how to fix.
- Run your code often to get the hints.
- Mistakes won't break computers.
- Some of the errors you may see...




```
KeyError: 'Hairy Potter'
```

```
ImportError: No module named humour
```

```
SyntaxError: Invalid Syntax
```

```
AttributeError: 'NoneType' object has no attribute 'foo'
```

```
TypeError: Can't convert 'int' object to str implicitly
```

We can learn from our mistakes!

```
Traceback (most recent call last):
  File "C:/Users/Madeleine/Desktop/tmp.py", line 9, in<module>
    print("I have " + 5 + " apples")
TypeError: can only concatenate str (not "int") to str
```

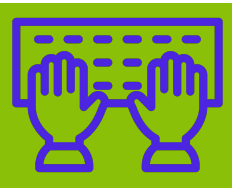
1. What went wrong

2. Which bit of code didn't work

3. Where that code is

We read error messages from bottom to top

Keeping organized with Comments!



Sometimes we want to write things in our file that the computer doesn't look at so we can write notes for later. We can use **comments** for that!

Sometimes we want to write a note for a people to read

```
# This code was written by Vivian
```

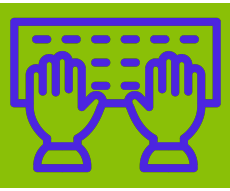
And sometimes we want to not run some code (but don't want to delete it!)

```
# print("Goodbye world!")
```

Try it!

1. Add a comment to your crypto.py file in Lesson 2.1
2. Run your code to make sure it doesn't do anything extra!

Write some code!!



1. Type the following into the “Playground” code window in chapter 2
2. Then run the code by clicking in the Terminal window

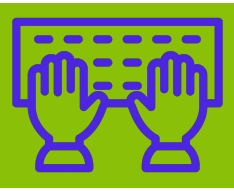
```
print('hello world')
```

Did it print:

```
hello world
```

???

Python the calculator!



Try writing some maths into python! After typing each line, test it out by clicking in the Terminal window.

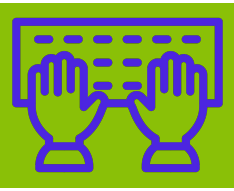
```
1. print(1 + 5)

2. print(2 - 7)

3. print(2 * 8)

4. print(12 / 3)
```

Python the calculator!



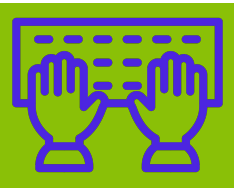
Try writing some maths into python! After typing each line, test it out by clicking in the Terminal window.

```
1. print(1 + 5)
   6
2. print(2 - 7)


3. print(2 * 8)


4. print(12 / 3)
```

Python the calculator!

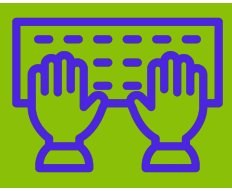


Try writing some maths into python! After typing each line, test it out by clicking in the Terminal window.

```
1. print(1 + 5)
   6
2. print(2 - 7)
   -5
3. print(2 * 8)

4. print(12 / 3)
```

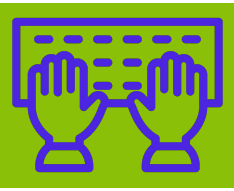
Python the calculator!



Try writing some maths into python! After typing each line, test it out by clicking in the Terminal window.

```
1. print(1 + 5)
   6
2. print(2 - 7)
   -5
3. print(2 * 8)
   16
4. print(12 / 3)
```

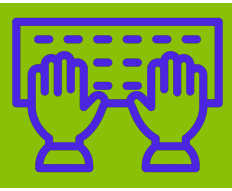
Python the calculator!



Try writing some maths into python! After typing each example, run by clicking in the Terminal window.

```
1. print(1 + 5)
   6
2. print(2 - 7)
   -5
3. print(2 * 8)
   16
4. print(12 / 3)
   4
```

A calculator for words!

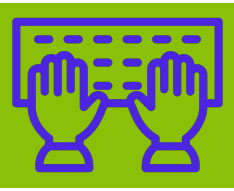


What do you think these bits of code do?
Try them! Run the code after typing in each example.

```
1. print("cat" + "dog")
```

```
2. print("tortoise" * 3)
```

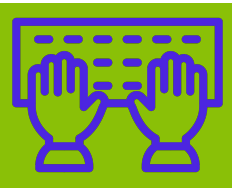
A calculator for words!



What do you think these bits of code do?
Try them! Run the code after typing in each example.

1.
```
print("cat" + "dog")
```
```
catdog
```
2.
```
print("tortoise" * 3)
```

A calculator for words!



What do you think these bits of code do?
Try them! Run the code after typing in each example.

1.
```
print("cat" + "dog")
```
```
catdog
```
2.
```
print("tortoise" * 3)
```
```
tortoisetortoisetortoise
```

Strings!

`Strings` are things with `"quotes"`

1. Strings can be added!

2. Strings can be multiplied!

Strings!

Strings can have any letters in them, even just spaces!

```
"Hello, world!"
"bla bla bla"
":)"      "  "
'I can use single quotes too!'
"¯\_(ツ)_/¯"
"asdfghjklqwertyuiopzxcvbnm"
"DOGS ARE AWESOME!"
"!@#$%^&*()_+-=[]|\:;'<>,./?"
```

Strings and Ints!

Integers are numbers in python.

We can do maths with integers but not strings.

Predict what will happen if we write and run the following code.

```
1.  print(5 + "5")
```

Strings and Ints!

Integers are numbers in python.

We can do maths with integers but not strings.

Predict what will happen if we write and run the following code.

```
1.  print(5 + "5")
```

TypeError: can only concatenate str (“not int”) to str

Strings and Ints!

Integers are numbers in python.

We can do maths with integers but not strings.

Predict what will happen if we write and run the following code.

```
1.  print(5 + "5")
    TypeError: can only concatenate str ("not int") to str
```

2. But, we can turn a string into an integer using int()

```
print(5 + int("5"))
```

Strings and Ints!

Integers are numbers in python.

We can do maths with integers but not strings.

Predict what will happen if we write and run the following code.

```
1.  print(5 + "5")
    TypeError: can only concatenate str ("not int") to str
```

2. But, we can turn a string into an integer using int()

```
print(5 + int("5"))
10
```

Strings and Ints!

Integers are numbers in python.

We can do maths with integers but not strings.

Predict what will happen if we write and run the following code.

```
1.  print(5 + "5")
    TypeError: can only concatenate str ("not int") to str
2.  But, we can turn a string into an integer using int()
    print(5 + int("5"))
    10
```

3. Similarly, we turn an integer into a string using str()

```
print(str(5) + "5")
```

Strings and Ints!

Integers are numbers in python.

We can do maths with integers but not strings.

Predict what will happen if we write and run the following code.

```
1.  print(5 + "5")
    TypeError: can only concatenate str ("not int") to str
2.  But, we can turn a string into an integer using int()
    print(5 + int("5"))
    10
```

3. Similarly, we turn an integer into a string using str()

```
print(str(5) + "5")
55
```

A special string...

There are some combinations of characters in python that do some interesting things...

One of these character combinations is `"\n"`

Let's see what it does?

```
>> print("\n")
```

A special string...

There are some combinations of characters in python that do some interesting things...

One of these character combinations is `"\n"`

Let's see what it does?

```
>> print("\n")



>>
```

A special string...

There are some combinations of characters in python that do some interesting things...

One of these character combinations is `"\n"`

Let's see what it does?

```
>> print("\n")

>>
```

But nothing happened?

A special string...

Let's try something that will be easier to notice...

Let's see what it does?

```
>> print("Hello\nWorld")
```

A special string...

Let's try something that will be easier to notice...

Let's see what it does?

```
>> print("Hello\nWorld")
Hello
World
```

Typing \n in a string results in a new line!

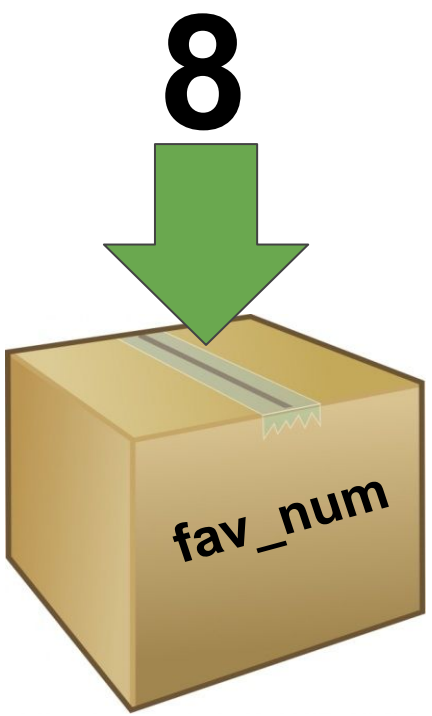
No Storing is Boring!

It's useful to be able to remember things for later!

Computers remember things in **"variables"**

Variables are like putting things into a **labeled cardboard box**.

Let's make our favourite number 8 today!



Variables

Instead of writing the number 8, we can write fav_num.


```
fav_num - 6
```

=> 2

```
fav_num + 21
```

=> 29

```
fav_num * 2
```

=> 16

```
fav_num / 2
```

=> 4

Variables

Instead of writing the number 8, we can write fav_num.


```
fav_num - 6
   => 2
```

```
fav_num + 21
   => 29
```

```
fav_num * 2
   => 16
```

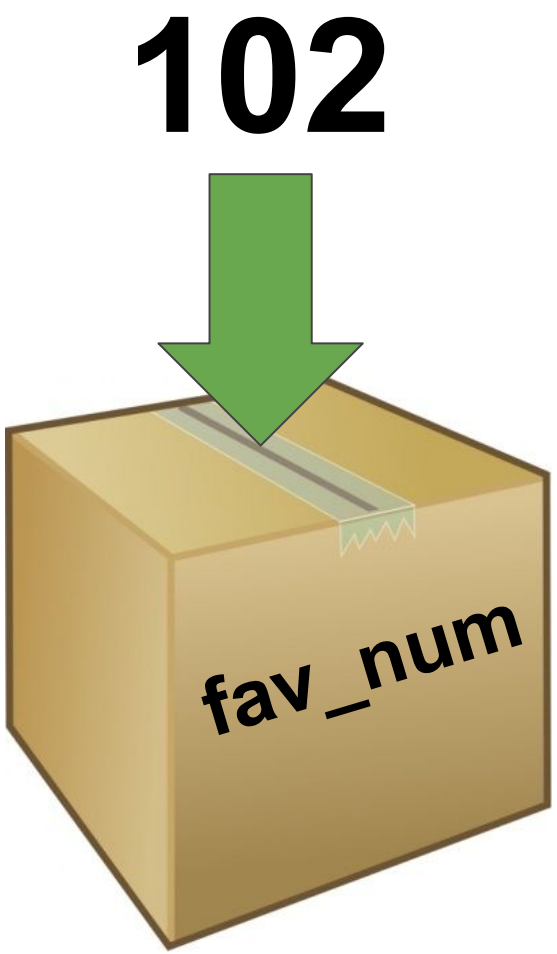
But writing 8 is much shorter than writing `fav_num`???

Variables

Variables are useful for storing things that change

(i.e. things that "vary" - hence the word "variable")

Try changing `fav_num` to **102**.



Variables

We're able to use our code for a new purpose, without rewriting everything:


```
fav_num - 6
```

=> 96

```
fav_num + 21
```

=> 123

```
fav_num * 2?
```

=> 204

```
fav_num / 2?
```

=> 51

No variables VS using variables

4
Changes

1
Change

8 - 6
8 * 2
8 + 21
8 / 2

102 - 6
102 * 2
102 + 21
102 / 2

fav_num = **8**
fav_num - 6
fav_num * 2
fav_num + 21
fav_num / 2

fav_num = **102**
fav_num - 6
fav_num * 2
fav_num + 21
fav_num / 2

Reusing variables

We can replace values in variables:

```
animal = "dog"
print("My favourite animal is a " + animal)
animal = "cat"
print("My favourite animal is a " + animal)
animal = animal + "dog"
print("My favourite animal is a " + animal)
```

What will this output?

Reusing variables

We can replace values in variables:

```
animal = "dog"
print("My favourite animal is a " + animal)
animal = "cat"
print("My favourite animal is a " + animal)
animal = animal + "dog"
print("My favourite animal is a " + animal)
```

```
My favourite animal is a dog
My favourite animal is a cat
My favourite animal is a catdog
```

What can we store?

We can put any value in a variable:

```
apples = 5 + 5
print(apples)
apples = apples - 1
print(apples)
apples = "Delicious"
print(apples)
```

What will this output?

What can we store?

We can put any value in a variable:

```
apples = 5 + 5
print(apples)
apples = apples - 1
print(apples)
apples = "Delicious"
print(apples)
```

```
10
9
Delicious
```

Variables

Your turn!

Can you guess what each `print` will do?

```
>>> x = 3
>>> print(x)

>>> print(x + x)

>>> y = x
>>> print(y)

>>> y = y + 1
>>> print(y)
```

Variables

Your turn!

Can you guess what each `print` will do?

```
>>> x = 3
>>> print(x)
3
>>> print(x + x)

>>> y = x
>>> print(y)

>>> y = y + 1
>>> print(y)
```

Variables

Your turn!

Can you guess what each `print` will do?

```
>>> x = 3
>>> print(x)
3
>>> print(x + x)
6
>>> y = x
>>> print(y)

>>> y = y + 1
>>> print(y)
```

Variables

Your turn!

Can you guess what each `print` will do?

```
>>> x = 3
>>> print(x)
3
>>> print(x + x)
6
>>> y = x
>>> print(y)
3
>>> y = y + 1
>>> print(y)
```

Variables

Your turn!

Can you guess what each `print` will do?

```
>>> x = 3
>>> print(x)
3
>>> print(x + x)
6
>>> y = x
>>> print(y)
3
>>> y = y + 1
>>> print(y)
4
```

Switcharoo - Making copies!

Set some variables!

```
>>> x = 3
>>> y = x
>>> x = 5
```

What do x and y contain now?

Let's find out together!

Switcharoo - Making copies!

Set some variables!

```
>>> x = 3
>>> y = x
>>> x = 5
```

What do x and y contain now?

```
>>> x
5
>>> y
3
```

y hasn't changed because it has a copy of x in it!

Different data!

There are lots of types of data! Our main 4 ones are these:

Strings

Things in quotes used for storing text

```
“This is a string”
```

Ints

Whole numbers we can do maths with

```
a = 1
b = 2
print(a + b)
```

Floats

Decimal numbers for maths

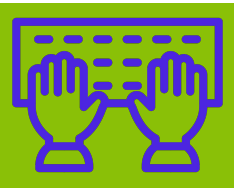
```
a = 1.5
b = 2.0
print(a / b)
```

Booleans

For `True` and `False`

```
a = 5 > 3
boring = False
```

Printing the data types



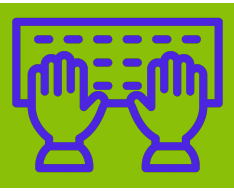
All of the four data types can be printed by putting them in a print statement like...

```
>> print(True)
```

or

```
>> print(5.5)
```

Printing the data types



All of the four data types can be printed by putting them in a print statement like...

```
>> print(True)
True
```

or

```
>> print(5.5)
```

Printing the data types

All of the four data types can be printed by putting them in a print statement like...

```
>> print(True)
True
```

or

```
>> print(5.5)
5.5
```

Printing the data types

All of the four data types can be printed by putting them in a print statement like...

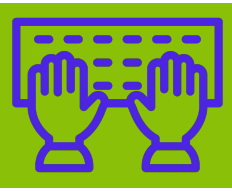
```
>> print(True)
True
```

or

```
>> print(5.5)
5.5
```

Remember earlier when we added two strings in a print? We can do a similar thing with variables!

Printing the data types



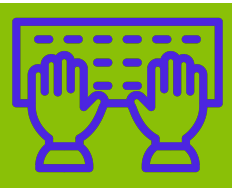
Let's try to print these things so we can see what will happen...

```
>> fav_num = "53"
>> print("My favourite number is "+fav_num)
```

or

```
>> fav_num = 53
>> print("My favourite number is "+fav_num)
```

Printing the data types



Let's try to print these things so we can see what will happen...

```
>> fav_num = "53"
>> print("My favourite number is "+fav_num)
My favourite number is 53
```

or

```
>> fav_num = 53
>> print("My favourite number is "+fav_num)
```

Printing the data types

Let's try to print these things so we can see what will happen...

```
>> fav_num = "53"
>> print("My favourite number is "+fav_num)
My favourite number is 53
```

or

```
>> fav_num = 53
>> print("My favourite number is "+fav_num)
Traceback (most recent call last): ...
```

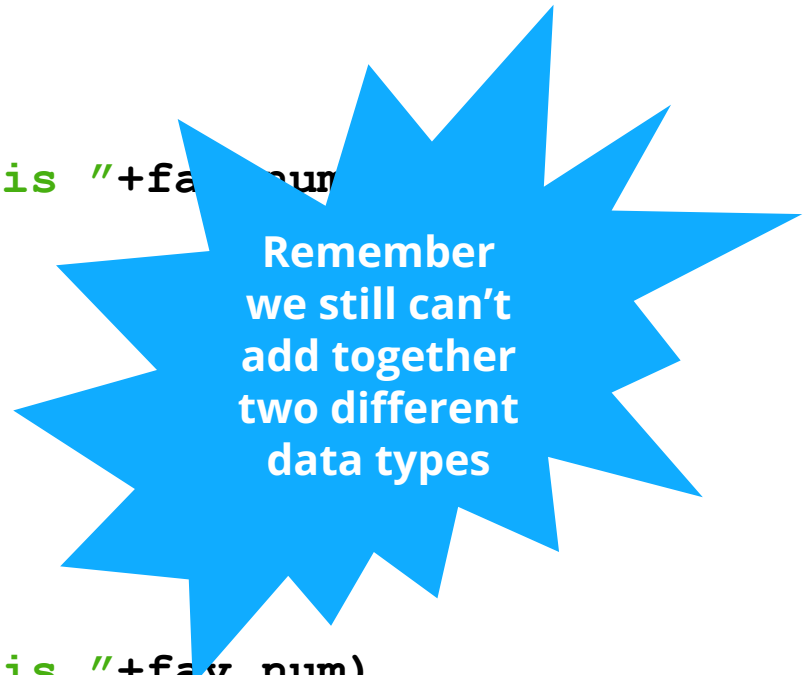
Printing the data types

Let's try to print these things so we can see what will happen...

```
>> fav_num = "53"
>> print("My favourite number is "+fa
My favourite number is 53
```

or

```
>> fav_num = 53
>> print("My favourite number is "+fav_num)
Traceback (most recent call last): ...
```



Asking a question!

It’s more fun when we get to interact with the computer!

Try out this code to get the computer to ask you a question!

```
my_name = input('What is your name? ')
print('Hello ' + my_name)
```

What do you think happens?

Asking a question!

It's more fun when we get to interact with the computer!

Try out this code to get the computer to ask you a question!

```
my_name = input('What is your name? ')
print('Hello ' + my_name)
```

What do you think happens?

```
What is your name? Maddie
Hello Maddie
```

Asking a question!

Store the answer in the variable my_name

Writing input tells the computer to wait for a response

This is the question you want printed to the screen

```
my_name = input('What is your name? ')
print('Hello ' + my_name)
```

What do you think happens?

```
What is your name? Maddie
Hello Maddie
```

We can use the answer the user wrote that we then stored later!

Asking a question!

How would we ask somebody for their favourite type of cake?

How would we print their answer?

```
What cake do you like? chocolate
chocolate cake for you!
```

Asking a question!

How would we ask somebody for their favourite type of cake?

How would we print their answer?

```
flavour = input('What cake do you like? ')
```

```
What cake do you like? chocolate
chocolate cake for you!
```

Asking a question!

How would we ask somebody for their favourite type of cake?

How would we print their answer?

```
flavour = input('What cake do you like? ')
print(flavour + ' cake for you!')
```

```
What cake do you like? chocolate
chocolate cake for you!
```

Project time!

You now know all about variables & input!

Let’s put what we learnt into our project

Try Lesson 2

The tutors will be around to help!

Strings, Ints & Modulo

Strings!

Strings are a sequence of characters in python.

Strings are created by enclosing characters inside `"quotes"`

`>>> alphabet = 'abcdefghijklmnopqrstuvwxyz'` creates a string variable that contains the letters of the alphabet

We can add strings together

```
>>> "abc" + “def” = “abcdef”
```

Strings

We can get individual letters from a string using indexes.

```
>>> yum = "chocolate"
>>> yum[0]


>>> yum[5]


>>> yum[-1]


>>> yum[500]
```

Strings

We can get individual letters from a string using indexes.

```
>>> yum = "chocolate"
>>> yum[0]
'c'
>>> yum[5]


>>> yum[-1]


>>> yum[500]
```

Computers start counting from 0, not 1!

Strings

We can get individual letters from a string using indexes.

```
>>> yum = "chocolate"
>>> yum[0]
'c'
>>> yum[5]
'l'
>>> yum[-1]

>>> yum[500]
```

Computers start counting from 0, not 1!

Strings

We can get individual letters from a string using indexes.

```
>>> yum = "chocolate"
>>> yum[0]
'c'
>>> yum[5]
'l'
>>> yum[-1]
'e'
>>> yum[500]
```

Computers start counting from 0, not 1!

Strings

We can get individual letters from a string using indexes.

```
>>> yum = "chocolate"
>>> yum[0]
'c'
>>> yum[5]
'l'
>>> yum[-1]
'e'
>>> yum[500]
IndexError: string index out of range
```

Computers start counting from 0, not 1!

Searching Strings

If we want to find where a letter is in a string, we look it up using `index()`

```
>>> yum = "chocolate"
>>> yum.index('h')


>>> yum.index('o')


>>> yum.index('z')
```

Searching Strings

If we want to find where a letter is in a string, we look it up using `index()`

```
>>> yum = "chocolate"
>>> yum.index('h')
1
>>> yum.index('o')

>>> yum.index('z')
```

Searching Strings

If we want to find where a letter is in a string, we look it up using `index()`

```
>>> yum = "chocolate"
>>> yum.index('h')
1
>>> yum.index('o')
2
>>> yum.index('z')
```

Only the index of the first 'o' is returned!

Searching Strings

If we want to find where a letter is in a string, we look it up using `index()`

```
>>> yum = "chocolate"
>>> yum.index('h')
1
>>> yum.index('o')
2
>>> yum.index('z')
ValueError: substring not found
```

Only the index of the first 'o' is returned!

Test if character in string

We can test if a character is in a string!

```
>>> yum = "chocolate"
>>> if ‘a’ in yum:
```

Maths on Indexes!

We can use any sort of **int** as an index, including the result of an expression or maths equation!

```
>>> yum = "chocolate"
>>> len(yum)


>>> yum[9 - 1]
```

Maths on Indexes!

We can use any sort of **int** as an index, including the result of an expression or maths equation!

```
>>> yum = "chocolate"
>>> len(yum)
9
>>> yum[9 - 1]
```

Maths on Indexes!

We can use any sort of **int** as an index, including the result of an expression or maths equation!

```
>>> yum = "chocolate"
>>> len(yum)
9
>>> yum[9 - 1]
'e'
```

Modulo %

Modulo **%** is a maths operation

% gives the **remainder** of a division

You'll need to use it in your code!

- 10 % 8 = 2 (10 divided by 8 is 1 with remainder 2)
- 20 % 7 = 6 (20 divided by 7 is 2 with remainder 6)
- 5 % 6 = 5 (5 divided by 6 is 0 with remainder 5)

Project time!

You now know all about strings, ints and modulo!

Let's put what we learnt into our project

Try Lesson 3

The tutors will be around to help!

For Loops

For Loops

For loops allow you to do something a certain number of times.

We use them when we know exactly how many times we want to do something!

For Loops

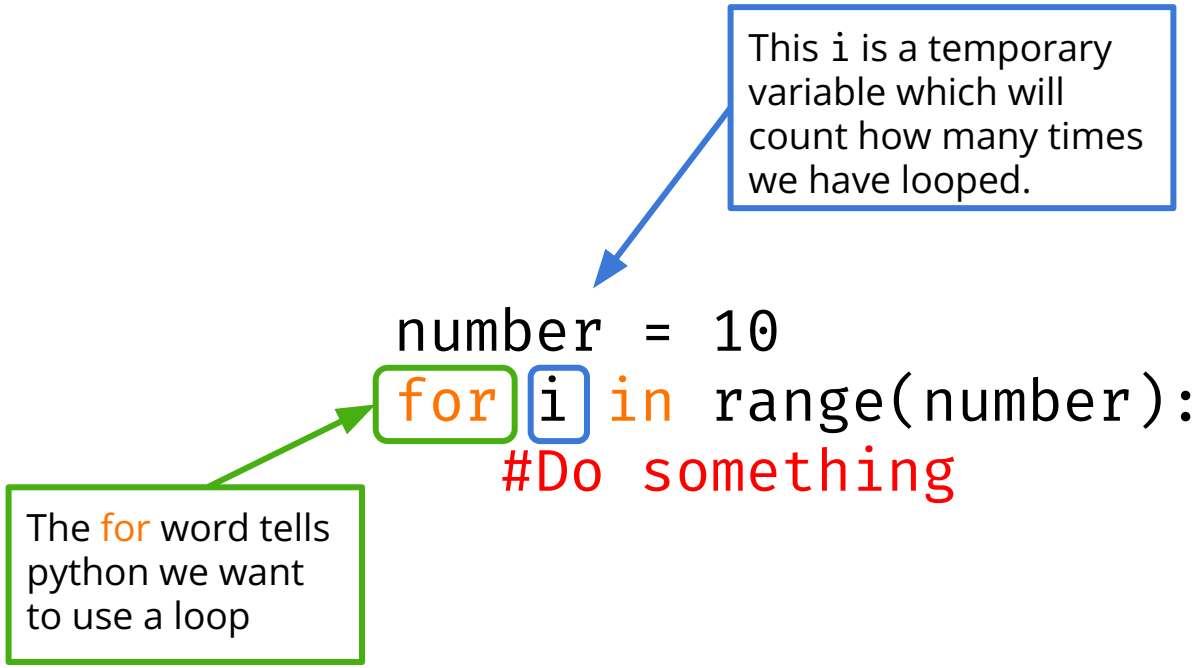
```
number = 10
for i in range(number):
    #Do something
```

For Loops

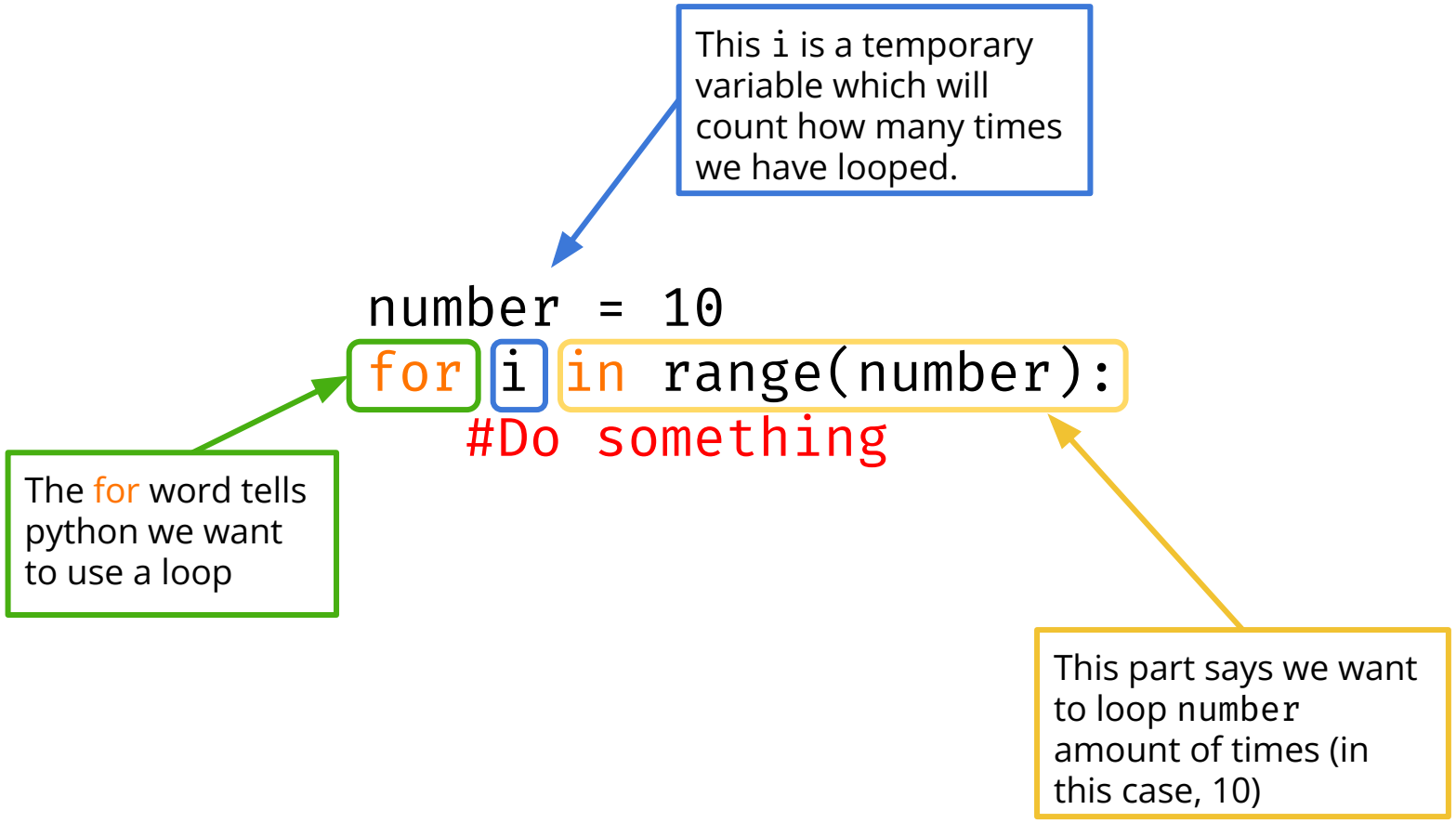
```
number = 10
for i in range(number):
    #Do something
```

The for word tells python we want to use a loop

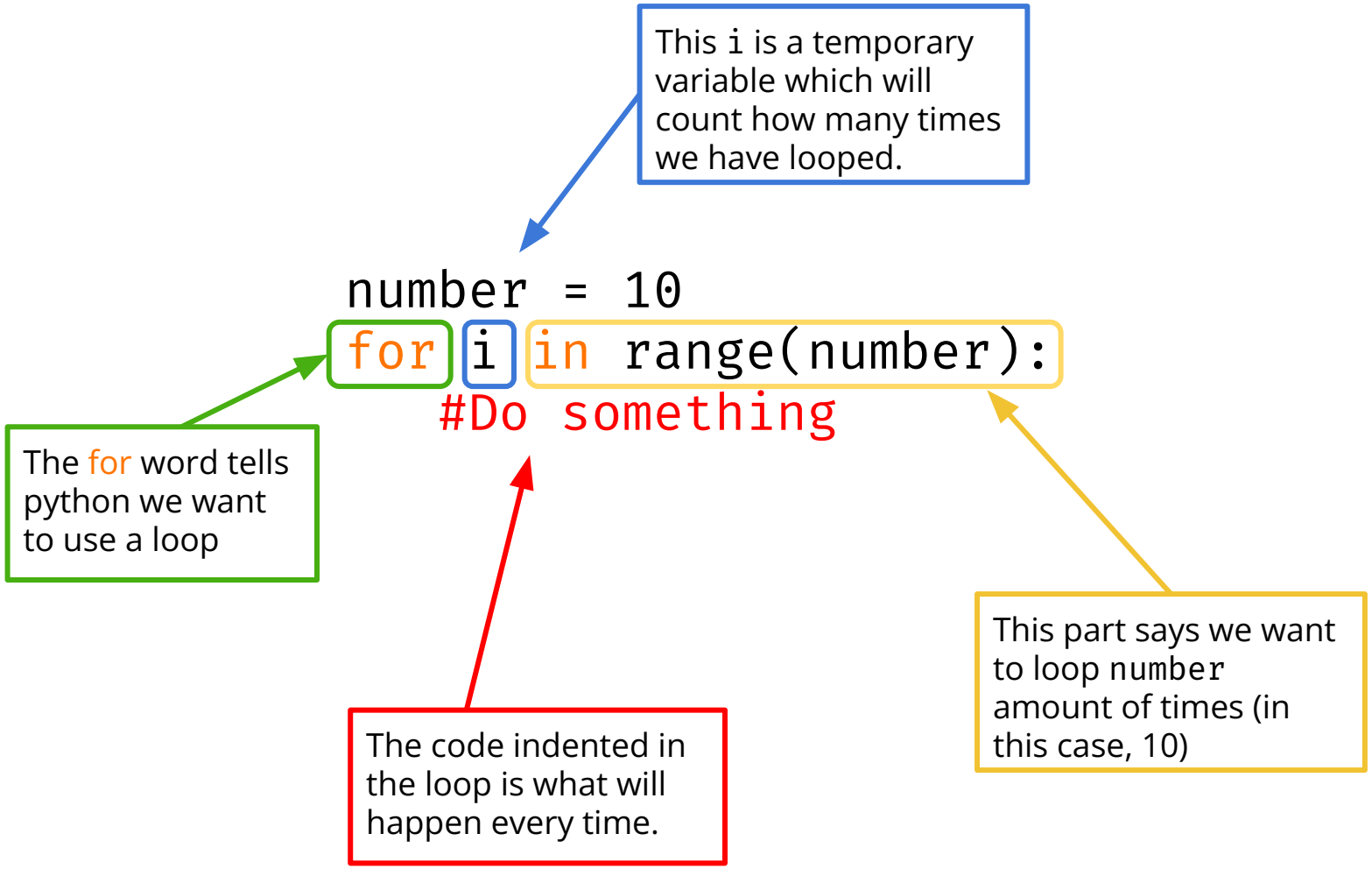
For Loops


This i is a temporary variable which will count how many times we have looped.
number = 10
for i in range(number):
#Do something
The for word tells python we want to use a loop

For Loops


This i is a temporary variable which will count how many times we have looped.
number = 10
for i in range(number):
#Do something
The for word tells python we want to use a loop
This part says we want to loop number amount of times (in this case, 10)

For Loops


This i is a temporary variable which will count how many times we have looped.
number = 10
for i in range(number):
#Do something
The for word tells python we want to use a loop
The code indented in the loop is what will happen every time.
This part says we want to loop number amount of times (in this case, 10)

Looping how many times?

We can loop through a list:

```
friends = 4
for i in range(friends):
    print(“Hello friend!”)
```

What’s going to happen?

We do what’s in the for loop as many times as what is in the “range”

Looping how many times?

We can loop through a list:

```
friends = 4
for i in range(friends):
    print(“Hello friend!”)
```

What’s going to happen?

```
>>> Hello friend!
>>> Hello friend!
>>> Hello friend!
>>> Hello friend!
```

We do what’s in the for loop as many times as what is in the “range”

Project Time!

Now you know how to use a for loop!

Try to do Lesson 4

...if you are up for it!

The tutors will be around to help!

If Statements

Conditions!

Conditions let us make decisions.

First we test if the condition is met!

Then maybe we'll do the thing

<u>If it's raining</u> take an umbrella

Yep it's raining

..... take an umbrella

Booleans (True and False)

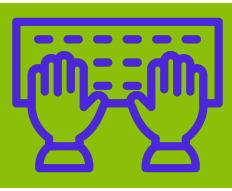
Computers store whether a condition is met in the form of

`True` and `False`

To figure out if something is `True` or `False` we do a comparison

Can you guess what these are?	
5 < 10	"Dog" == "dog"
3 + 2 == 5	"D" in "Dog"
5 != 5	"Q" not in "Cat"

Booleans (True and False)



Python has some special comparisons for checking if something is **`in`** something else. **Try these!**

```
>>> "A" in "AEIOU"
>>> "Z" in "AEIOU"
>>> "a" in "AEIOU"
```

```
>>> animals = ["cat", "dog", "goat"]
>>> "banana" in animals
>>> "cat" in animals
```

Booleans (True and False)

Python has some special comparisons for checking if something is **`in`** something else. **Try these!**

```
True  "A" in "AEIOU"
False "Z" in "AEIOU"
False "a" in "AEIOU"
```

```
>>> animals = ["cat", "dog", "goat"]
False "banana" in animals
True  "cat" in animals
```

Conditions

So to know whether to do something, they find out if it's `True`!

```
fave_num = 5
if fave_num < 10:
    print("that's a small number")
```

Conditions

So to know whether to do something, they find out if it's `True`!

```
fave_num = 5
if fave_num < 10:
    print("that's a small number")
```

That's the condition!

Conditions

So to know whether to do something, they find out if it's `True`!

```
fave_num = 5
if fave_num < 10:
    print("that's a small number")
```

That's the condition!

Is it `True` that fave_num is less than 10?

- Well, fave_num is 5
- And it's `True` that 5 is less than 10
- So it is `True`!

Conditions

So to know whether to do something, they find out if it's `True`!

```
fave_num = 5
if True:
    print("that's a small number")
```

Put in the answer to the question

Is it `True` that fave_num is less than 10?

- Well, fave_num is 5
- And it's `True` that 5 is less than 10
- So it is `True`!

Conditions

So to know whether to do something, they find out if it's `True`!

```
fave_num = 5
if True:
    print("that's a small number")
```

What do you think happens?

```
>>>
```

Conditions

So to know whether to do something, they find out if it's `True`!

```
fave_num = 5
if True:
    print("that's a small number")
```

What do you think happens?

```
>>> that's a small number
```

Conditions

How about a different number???

```
fave_num = 9000
if fave_num < 10:
    print("that's a small number")
```

Conditions

Find out if it's `True`!

```
fave_num = 9000
if False:
    print("that's a small number")
```

Put in the answer to the question

Is it `True` that fave_num is less than 10?

- Well, fave_num is 9000
- And it's not `True` that 9000 is less than 10
- So it is `False`!

Conditions

How about a different number???

```
fave_num = 9000
if fave_num < 10:
    print("that's a small number")
```

What do you think happens?

```
>>>
```

Conditions

How about a different number???

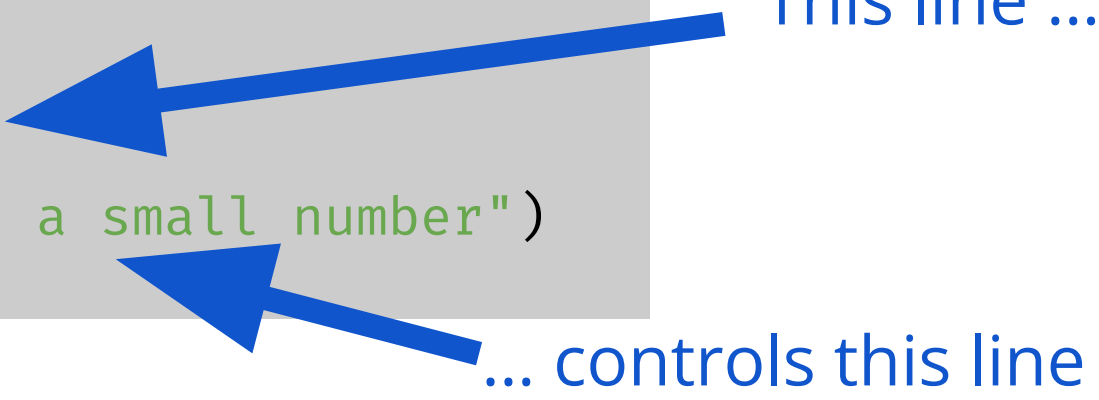
```
fave_num = 9000
if fave_num < 10:
    print("that's a small number")
```

What do you think happens?

```
>>>
```

If statements

```
fave_num = 5
if fave_num < 10:
    print("that's a small number")
```



If statements

Actually .....

```
fave_num = 5
if fave_num < 10:
    print("that's a small number")
    print("and I like that")
    print("A LOT!!")
```

This line ...

... controls anything below it that is indented like this!

If statements

```
fave_num = 5
if fave_num < 10:
    print("that's a small number")
    print("and I like that")
    print("A LOT!!")
```

What do you think happens?

>>>

If statements

```
fave_num = 5
if fave_num < 10:
    print("that's a small number")
    print("and I like that")
    print("A LOT!!")
```

```
>>> that's a small number
>>> and I like that
>>> A LOT!!
```

If statements

```
word = "GPN"
if word == "GPN":
  print("GPN is awesome!")
```

What happens?

If statements

```
word = "GPN"
if word == "GPN":
  print("GPN is awesome!")
```

What happens?

```
>>> GPN is awesome!
```

If statements

```
word = "GPN"
if word == "GPN":
  print("GPN is awesome!")
```

What happens?

```
>>> GPN is awesom
```

But what if we want something different to happen if the word isn't "GPN"

Else statements

`else` statements means something still happens if the **`if`** statement was **`False`**

```
word = "Chocolate"
if word == "GPN":
  print("GPN is awesome!")
else:
  print("The word isn't GPN :(")
```

What happens?

Else statements

`else` statements means something still happens if the **`if`** statement was **`False`**

```
word = "Chocolate"
if word == "GPN":
  print("GPN is awesome!")
else:
  print("The word isn't GPN :(")
```

What happens?

```
>>> The word isn't GPN :(
```

Elif statements

elif
Means we can give specific instructions for other words

```
word = "Chocolate"
if word == "GPN":
  print("GPN is awesome!")
elif word == "Chocolate":
  print("YUMMM Chocolate!")
else:
  print("The word isn't GPN :(")
```

What happens?

Elif statements

elif
Means we can give specific instructions for other words

```
word = "Chocolate"
if word == "GPN":
  print("GPN is awesome!")
elif word == "Chocolate":
  print("YUMMM Chocolate!")
else:
  print("The word isn't GPN :(")
```

What happens?

```
>>> YUMMM Chocolate!
```

Project Time!

You now know all about `if` and `else`!

See `if` you can do Lesson 5

The tutors will be around to help!