

Welcome to the Labs

Scissors Paper Rock!



Thank you to our Sponsors!

Platinum Sponsor:



Who are the tutors?



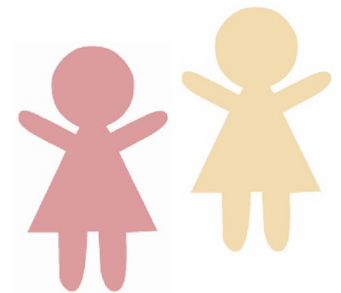
Who are you?



Ultimate Scissors Paper Rock

1. Start with a partner
2. play scissors paper rock!

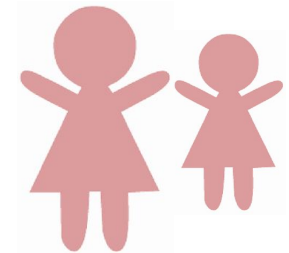
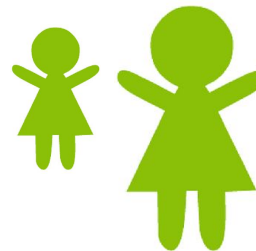
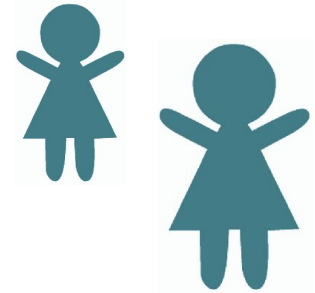
Who will be the champion?



Ultimate Scissors Paper Rock

1. Start with a partner
2. play scissors paper rock!
3. If you win they become your cheer squad!
And their squad becomes your squad!
4. Find a new partner!
5. Keep playing until there is only one person left!

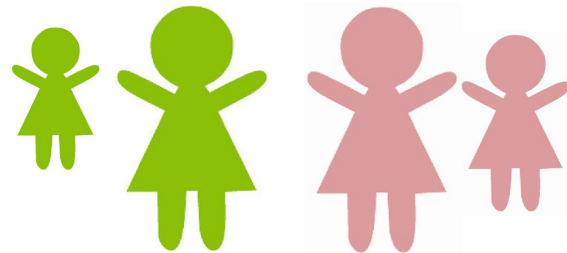
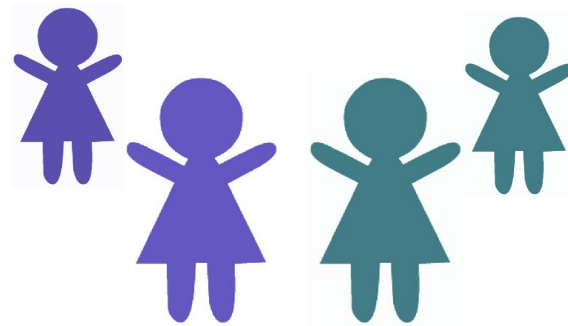
Who will be the champion?



Ultimate Scissors Paper Rock

1. Start with a partner
2. play scissors paper rock!
3. If you win they become your cheer squad!
And their squad becomes your squad!
4. Find a new partner!
5. Keep playing until there is only one person left!

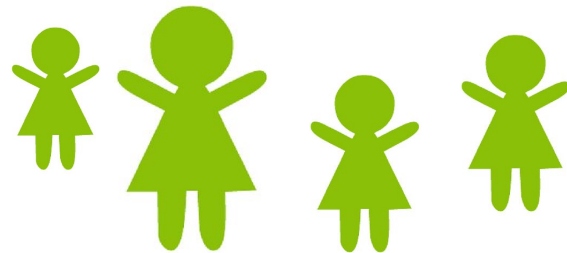
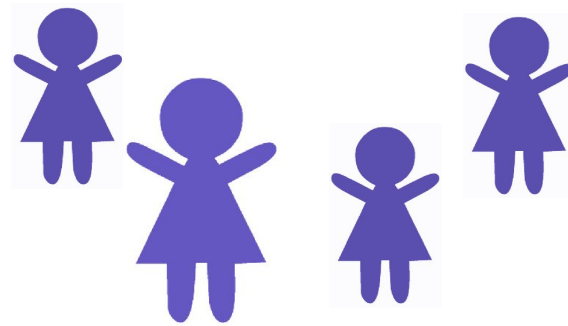
Who will be the champion?



Ultimate Scissors Paper Rock

1. Start with a partner
2. play scissors paper rock!
3. If you win they become your cheer squad!
And their squad becomes your squad!
4. Find a new partner!
5. Keep playing until there is only one person left!

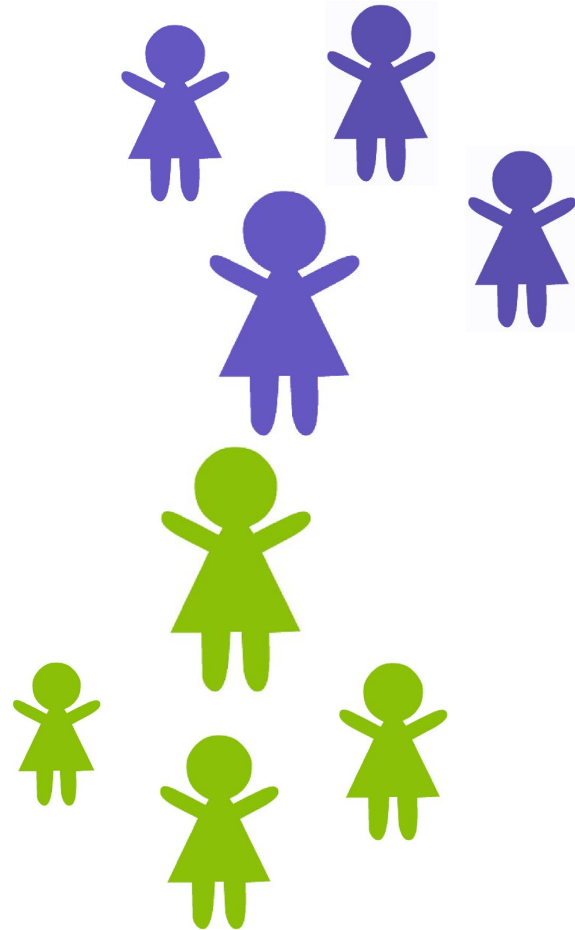
Who will be the champion?



Ultimate Scissors Paper Rock

1. Start with a partner
2. play scissors paper rock!
3. If you win they become your cheer squad!
And their squad becomes your squad!
4. Find a new partner!
5. Keep playing until there is only one person left!

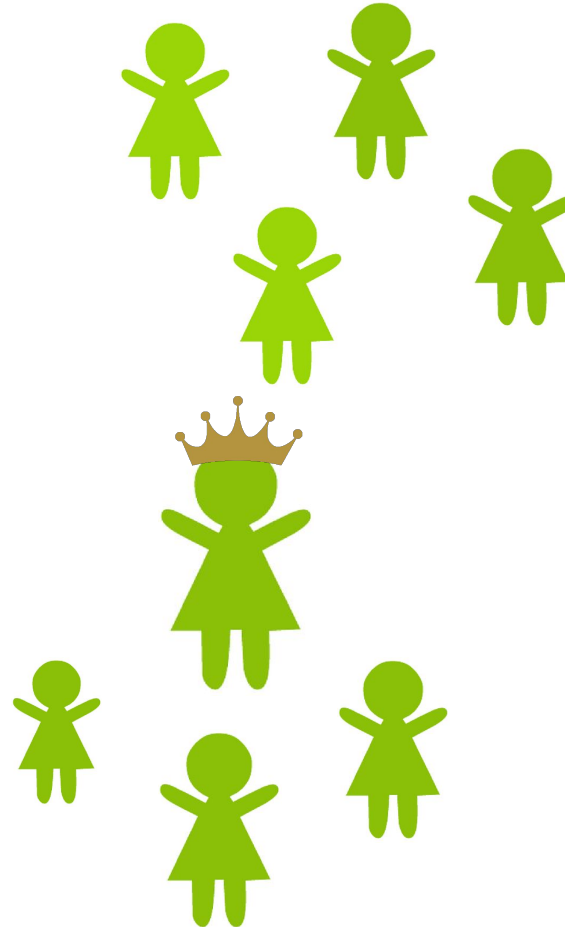
Who will be the champion?



Ultimate Scissors Paper Rock

1. Start with a partner
2. play scissors paper rock!
3. If you win they become your cheer squad!
And their squad becomes your squad!
4. Find a new partner!
5. Keep playing until there is only one person left!

Who will be the champion?



Log on

Log on and jump on the GPN website

girlsprogramming.network/workshop

You can see:

- These **slides** (to take a look back or go on ahead).
- A digital copy of your **workbook**.
- Help bits of text you can **copy and paste**!

There's also links to places where you can do more programming!



Tell us you're here!

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



Today's project!

Scissors Paper Rock



Using the workbook!

The workbooks will help you put your project together!

Each **Part** of the workbook is made of tasks!

Tasks - The parts of your project

Follow the tasks **in order** to make the project!

Hints - Helpers for your tasks!

Stuck on a task, we might have given you a hint to help you **figure it out**!

The hints have **unrelated** examples, or tips. **Don't copy and paste** in the code, you'll end up with something **CRAZY**!

Task 6.2: Add a blah to your code!

This has instructions on how to do a part of the project

1. **Start by doing this part**
2. **Then you can do this part**

Task 6.1: Make the thing do blah!

Make your project do blah

Hint

A clue, an example or some extra information to help you **figure out** the answer.

```
print('This example is not part of the project' )
```



Using the workbook!

The workbooks will help you put your project together!

Check off before you move on from a **Part!** Do some bonuses while you wait!

Checklist - Am I done yet?

Make sure you can tick off every box in this section before you go to the next Part.

Lecture Markers

This tells you you'll find out how to do things for this section during the names lecture.

Bonus Activities

Stuck waiting at a lecture marker? Try a purple bonus. They add extra functionality to your project along the way.

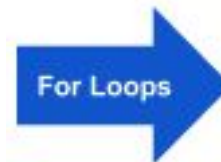


CHECKPOINT



If you can tick all of these off you're ready to move the next part!

- ☐ Your program does blah
- ☐ Your program does blob



★ BONUS 4.3: Do some extra!

Something to try if you have spare time before the next lecture!



Intro to programming



What is programming?



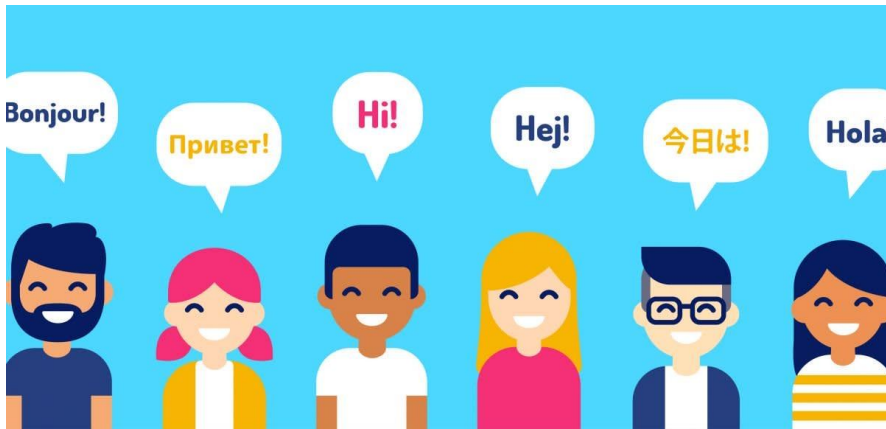
**Programming is not a
bunch of crazy numbers!**

**It's giving computers
a set of instructions!**



A special language

Humans have languages like English, French, Spanish, Mandarin



https://images.saymedia-content.com/.image/t_share/MTc0MTAyNzI3ODUxMjU1MjQx/how-to-easily-learn-a-language.jpg

And computers have languages like Python, Java, C and PHP



Problem solving

Programming is how we get computers to solve complicated problems for us, saving us both time and effort!

This might be solving maths problems or counting words in a paragraph!



People are smart, computers are dumb!

Computers do exactly what they're told. They follow instructions given to them in order, just like a cook following a recipe.



If the instructions are not in the correct order, we will end up with a mess!

Everyone/thing has strengths!



- Incomplete instructions are okay - we can fill in the blanks!
- Improves everyday



- Incomplete instructions are not okay
- Improves when you tell it how to

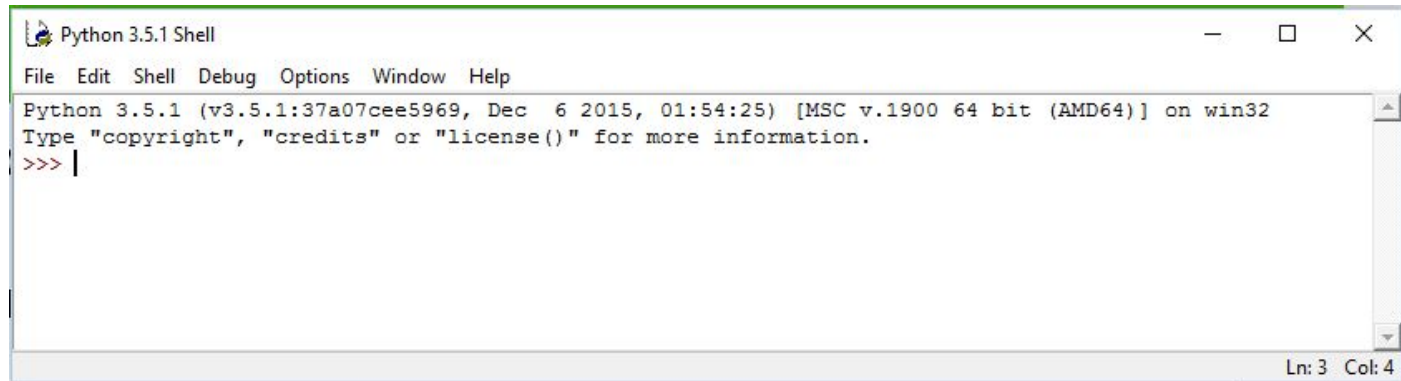
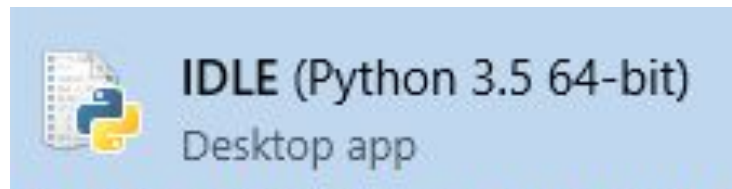
Intro to Python

Let's get coding!



Where do we program? In IDLE

Click the start button and type IDLE!



Make a mistake!

Type by **button mashing** the keyboard!

Then press enter!

asdf asdjlkj;pa j;k4uroei

Did you get a big red error message?



Mistakes are great!

*SyntaxError:
Invalid Syntax*

Good work you made an error!

*ImportError:
No module
named humour*

- Programmers make A LOT of errors!
- Errors give us hints to find mistakes
- Run your code often to get the hints!!
- Mistakes won't break computers!



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

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

*KeyError:
'Hairy Potter'*



Write some code!

Type this into the window
Then press enter!

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

Did it print:

hello world

???



Data types

In programming, we have special names for the following:

Number → Integer

Letter → Character

Word → String

Let's look at some examples



Characters - not always letters

What do all of these have in common?

"A"

'6'

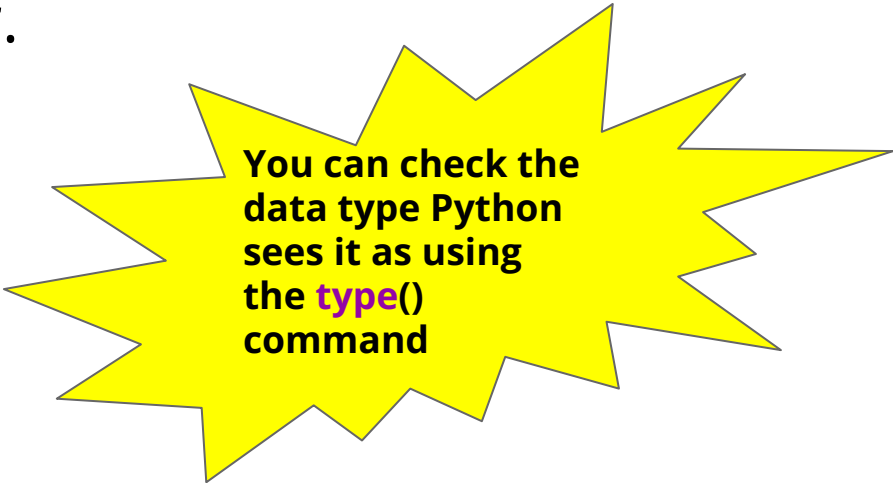
"f"

'\$'

Anything that only takes up only one space and is surrounded by 'single' or "double" quotes, is considered a **character** by the computer.

```
type("5") = char
```

```
type(5) = int
```



You can check the data type Python sees it as using the **type()** command

Strings

Strings are a group of more than one **character** put together and surrounded with "**quotes**"

All of these are strings:

"Dog"

"my name is"

"123 haha"

"\$%#^&@(){}[]"



A calculator for words!?

What do you think these bits of code do?

Try them and see!

```
>>> "cat" + "dog"
```

```
>>> "tortoise" * 3
```



Calculator for... words!?

What do you think these bits of code do?

Try them and see!

```
>>> "cat" + "dog"
```

```
catdog
```

```
>>> "tortoise" * 3
```



Calculator for... words!?

What do you think these bits of code do?

Try them and see!

```
>>> "cat" + "dog"
```

```
catdog
```

```
>>> "tortoise" * 3
```

```
tortoisetortoisetortoise
```



Calculator for words and number?

If we can do calculations with numbers, and calculations with words, can we do calculations with words *and* numbers?

Try writing this!

```
>>> 1 + "1"
```

```
>>> "100" * 2
```

How do we deal with this problem? See next slide!



Type casting

We tell the computer exactly what type we want to use!

We can turn a `string` into an `integer` using `int()`

```
>>> 5 + int("5")
```

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

```
>>> str(5) + "5"
```

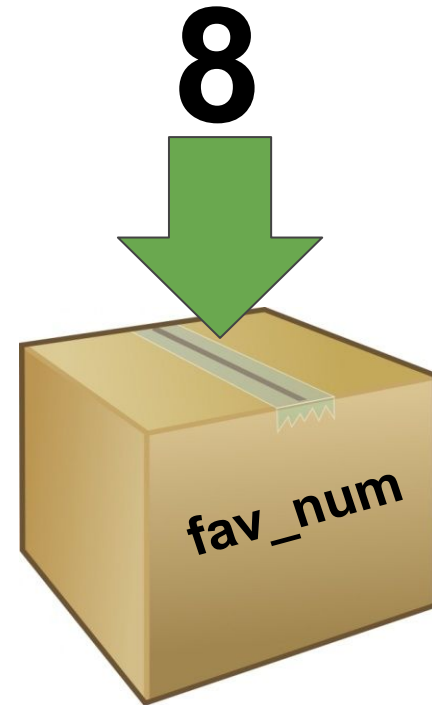
No Storing is Boring!

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.



$$\text{fav_num} - 6 \\ \Rightarrow \mathbf{2}$$

$$\text{fav_num} + 21 \\ \Rightarrow \mathbf{29}$$

$$\text{fav_num} * 2 \\ \Rightarrow \mathbf{16}$$

$$\text{fav_num} / 2 \\ \Rightarrow \mathbf{4}$$



Variables

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



$$\text{fav_num} - 6 \\ \Rightarrow 2$$

$$\text{fav_num} + 21 \\ \Rightarrow 29$$

$$\text{fav_num} * 2 \\ \Rightarrow 16$$

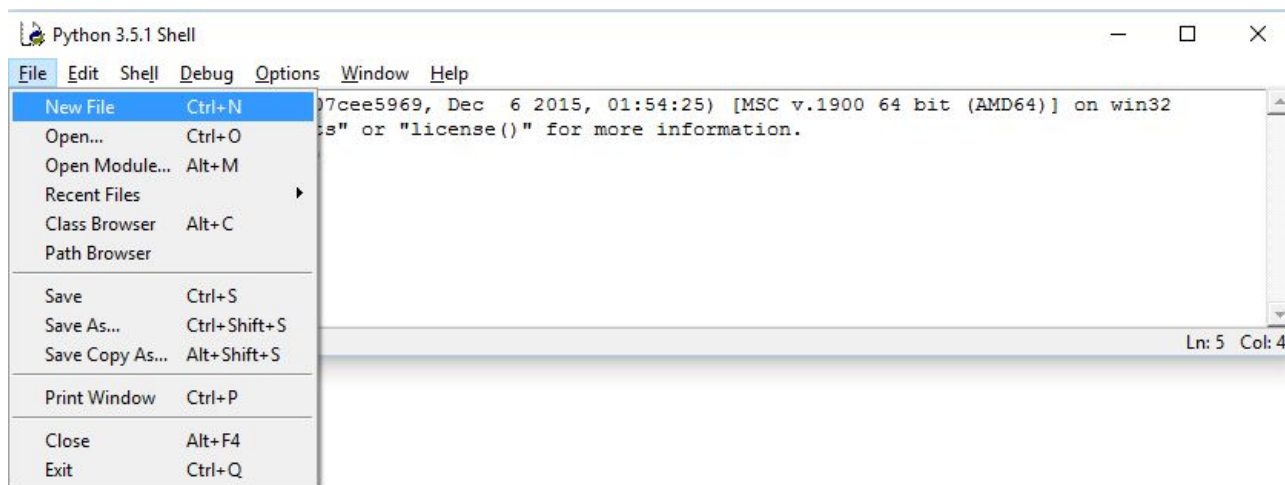
But writing 8 is
much shorter than
writing fav_num???



We'll come back to this later!

Coding in a file!

Code in a file is code we can run multiple times! Make a reusable “hello world”!



1. Make a new file called hello.py, like the picture
2. Put your `print('hello world')` code in it
3. Run your file using the F5 key



Adding a comment!

Sometimes we want to write things in our file that the computer doesn't look at. 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 hello.py file
2. Run your code to make sure it doesn't do anything extra!



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? ')\nprint('Hello ' + my_name)
```

What do you think happens?



Asking a question!

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

What do you think happens?

```
What is your name? Maddie\nHello 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?

Give it a try on your own computer first!



```
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?

Give it a try on your own computer first!

```
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?

Give it a try on your own computer first!

```
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 printing and variables!

Let's put what we learnt into our project
Try to do Part 0 - Part 2

The tutors will be around to help!



If Statements



Conditions!

Conditions let us make decision.
First we test if the condition is met!
Then maybe we'll do the thing



If it's raining 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

Try typing these into IDLE!

`5 < 10`

`3 + 2 == 5`

`5 != 5`

`"Dog" == "dog"`

`"D" in "Dog"`

`"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
```



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

It's **False**!

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

Put in the
answer to
the question



Conditions

It's **False**!

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

What do you think happens?

```
>>>
```



Conditions

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

What do you think happens?

```
>>>
```



Nothing!

If statements

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

This line ...

... controls this line



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

What do you think happens?

```
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

What do you think happens?

```
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!
```



Else statements

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

What happens??

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

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

Else statements

```
word = "Chocolate"
```

```
"GPN":
```

```
    "GPN is awesome!")
```

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

```
word isn't GPN :(")
```

What happens??

Else statements

```
word = "Chocolate"
```

```
"GPN":
```

```
    "GPN is awesome!")
```

```
word isn't GPN :(")
```

else
Statements
means something
still happens if
the **if** statement
was **False**

What happens??

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

Elif statements

```
word = "Chocolate"

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

elif

Means we can
give specific
instructions for
other words

What happens??

Practice Time!



1. Create a new file, call it weather.py
2. Copy this code into your file

```
weather = input("What is the weather? ")  
if weather == "raining":
```

3. Add a third line to make it print a special message, but only if the user says "raining"
4. Run your code! Try typing in **raining**, try typing in **sunny**
5. BONUS! Add an else statement, to print a non-rainy message!



Practice Time!

1. Create a new file, call it weather.py
2. Copy this code into your file

```
weather = input("What is the weather? ")  
if weather == "raining":  
    print("Take an umbrella!")
```

3. Add a third line to make it print a special message, but only if the user says "raining"
4. Run your code! Try typing in **raining**, try typing in **sunny**
5. BONUS! Add an else statement, to print a non-rainy message!



Project Time!



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

See **if you can do Part 3**

The tutors will be around to help!

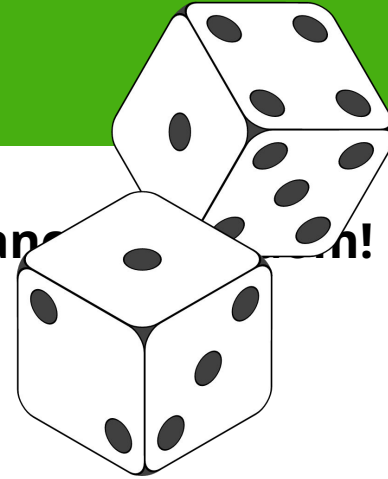


Random!



That's so random!

There's lots of things in life that are up to chance!



We want the computer to be random sometimes!



Python lets us **import** common bits of code people use! We're going to use the **random** module!



Using the random module



Let's choose something randomly from a list!

This is like drawing something out of a hat in a raffle!

Try this!

1. Import the random module!

```
>>> import random
```

2. Copy the shopping list into IDLE

```
>>> shopping_list = ["eggs", "bread", "apples", "milk"]
```

3. Choose randomly! Try it a few times!

```
>>> random.choice(shopping_list)
```



Using the random module



You can also assign your random choice to a variable

```
>>> import random
>>> shopping_list = ["eggs", "bread", "apples", "milk"]
>>> random_food = random.choice(shopping_list)
>>> print(random_food)
```



Project Time!



Raaaaaaaaaandom! Can you handle that?

Let's put what we learnt into our project

Try to do Part 4

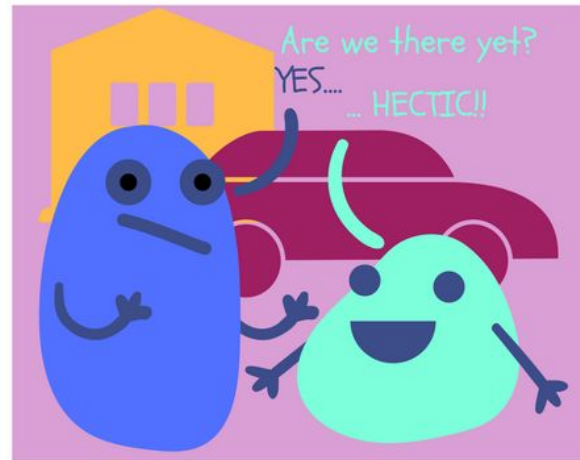
The tutors will be around to help!



While Loops



Loops



We know how to do things on repeat!

Sometimes we want to do some code on repeat!

Introducing ... while loops!

What do you think this does?

```
i = 0
while i < 3:
    print("i is " + str(i))
    i = i + 1
```



Introducing ... while loops!

What do you think this does?

```
i = 0
while i < 3:
    print("i is " + str(i))
    i = i + 1
```

```
i is 0
i is 1
i is 2
>>>
```



Introducing ... while loops!

Stepping through a while loop...



Introducing ... while loops!

One step at a time!

```
◆ i = 0  
  while i < 3:  
    print("i is " + str(i))  
    i = i + 1
```

MY VARIABLES

i = 0

Set the
variable



Introducing ... while loops!

One step at a time!

0 is less
than 3!

```
i = 0
while i < 3:
    print("i is " + str(i))
    i = i + 1
```

MY VARIABLES

i = 0



Introducing ... while loops!

One step at a time!

Print!

```
i = 0
while i < 3:
    print("i is " + str(i))
    i = i + 1
```

i is 0

MY VARIABLES

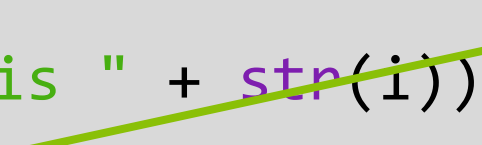
i = 0



Introducing ... while loops!

One step at a time!

```
i = 0
while i < 3:
    print("i is " + str(i))
    ◆ i = i + 1
```



MY VARIABLES

```
i = 0
i = 1
```

UPDATE
TIME!

```
i is 0
```



Introducing ... while loops!

One step at a time!

Take it
from the
top!

```
i = 0
while i < 3:
    print("i is " + str(i))
    i = i + 1
```

```
i is 0
```

MY VARIABLES

```
i = 0
i = 1
```



Introducing ... while loops!

One step at a time!

i is less
than 3!

```
i = 0
while i < 3:
    print("i is " + str(i))
    i = i + 1
```

MY VARIABLES

```
i = 0
i = 1
```

```
i is 0
```



Introducing ... while loops!

One step at a time!

Print!

```
i = 0
while i < 3:
    print("i is " + str(i))
    i = i + 1
```

```
i is 0
i is 1
```

MY VARIABLES


```
i = 0
i = 1
```



Introducing ... while loops!

One step at a time!

```
i = 0
while i < 3:
    print("i is " + str(i))
    ◆ i = i + 1
```



MY VARIABLES

```
i = 0
i = 1
i = 2
```

UPDATE
TIME!

```
i is 0
i is 1
```



Introducing ... while loops!

One step at a time!

Take it
from the
top!

```
i = 0
while i < 3:
    print("i is " + str(i))
    i = i + 1
```

```
i is 0
i is 1
```

MY VARIABLES

```
i = 0
i = 1
i = 2
```



Introducing ... while loops!

One step at a time!

2 is less
than 3!

```
◆ i = 0
  while i < 3:
    print("i is " + str(i))
    i = i + 1
```

MY VARIABLES

```
i = 0
i = 1
i = 2
```

```
i is 0
```

```
i is 1
```



Introducing ... while loops!

One step at a time!

Print!

```
i = 0
while i < 3:
    print("i is " + str(i))
    i = i + 1
```

```
i is 0
i is 1
i is 2
```

MY VARIABLES


```
i = 0
i = 1
i = 2
```



Introducing ... while loops!

One step at a time!

```
i = 0
while i < 3:
    print("i is " + str(i))
    ◆ i = i + 1
```



MY VARIABLES

```
i = 0
i = 1
i = 2
i = 3
```

UPDATE
TIME!

```
i is 0
i is 1
i is 2
```



Introducing ... while loops!

One step at a time!

Take it
from the
top!

```
i = 0
while i < 3:
    print("i is " + str(i))
    i = i + 1
```

```
i is 0
i is 1
i is 2
```

MY VARIABLES

```
i = 0
i = 1
i = 2
i = 3
```



Introducing ... while loops!

One step at a time!

3 IS NOT
less than
3!

```
i = 0
while i < 3:
    print("i is " + str(i))
    i = i + 1
```

We are
done
with this
loop!

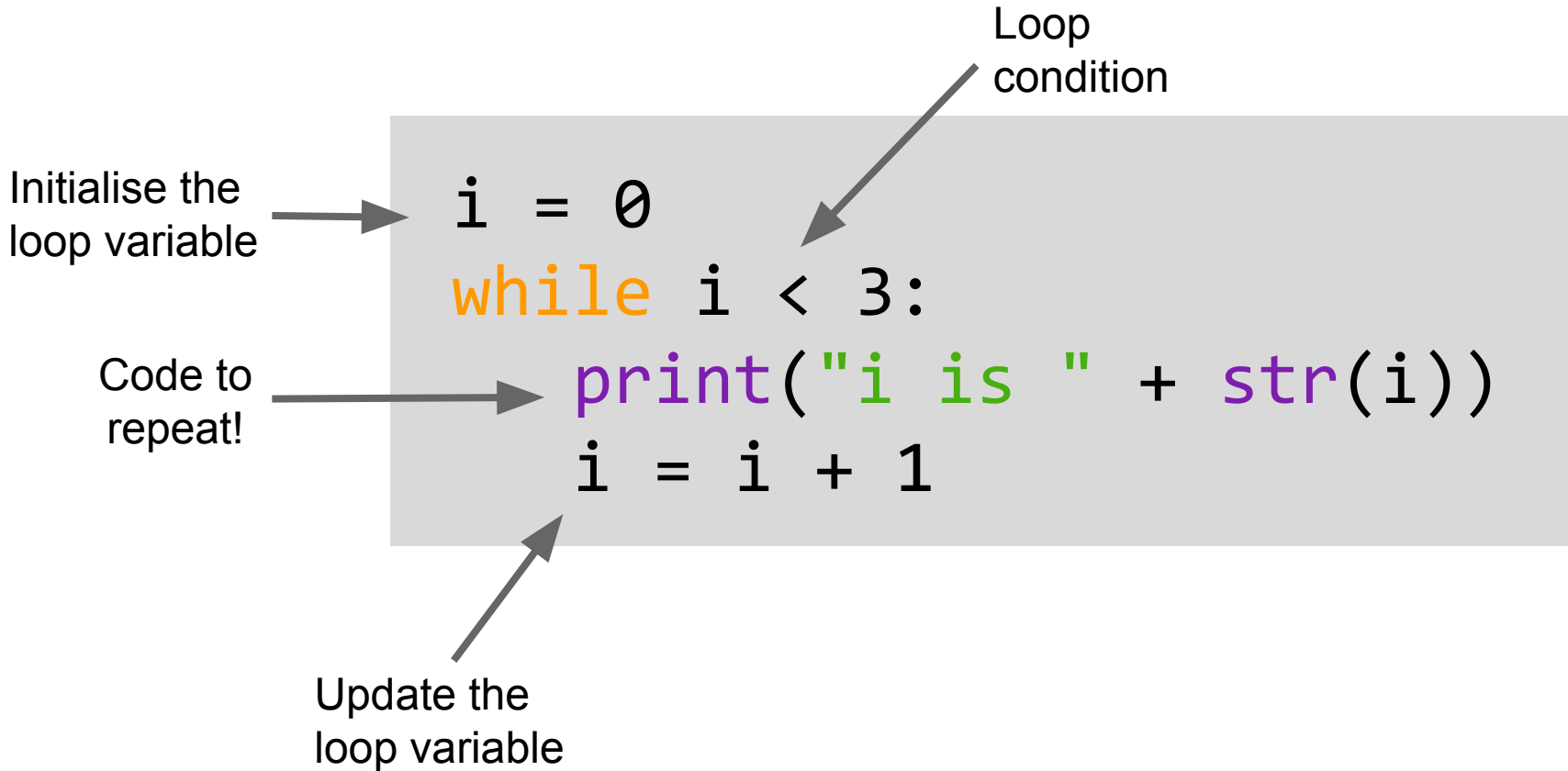
```
i is 0
i is 1
i is 2
```

MY VARIABLES

```
i = 0
i = 1
i = 2
i = 3
```



Introducing ... while loops!



What happens when.....

What happens if we forget to update the loop variable?

```
i = 0
while i < 3:
    print("i is " + str(i))
```



What happens when.....

What happens if we forget to update the loop variable?

```
i = 0
while i < 3:
    print("i is " + str(i))
```

i is 0

i is 0

i is 0

i is 0

i is 0

i is 0

i is 0

i is 0

i is 0

i is 0

i is 0

i is 0

i is 0

i is 0



Infinite loop!

Sometimes we want our loop to go forever!

So we set a condition that is always True!

We can even just write True!

```
while True:  
    print("Are we there yet?")
```



Project Time!



while we're here:

Try to do Part 5!

And extension Parts 8 & 9

The tutors will be around to help!



For Loops



Looping through lists!

What would we do if we wanted to print out this list, one word at a time?

```
words = ['This', 'is', 'a', 'sentence']
```

```
print(words[0])
```

```
print(words[1])
```

```
print(words[2])
```

```
print(words[3])
```

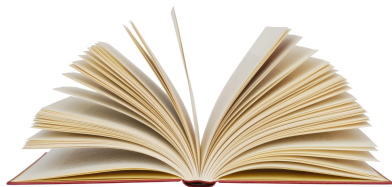
What if it had a 100 items??? That would be BORING!



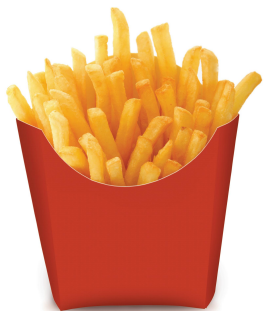
For Loops

For loops allow you to do something for **each** item in a **group** of things

There are many real world examples, like:



**For each page in this book:
Read**



**For each chip in this bag of chips:
Eat**



Looping over a list of ints

We can loop through a list:

```
numbers = [1, 2, 3, 4]
for i in numbers:
    print(i)
```

What's going to happen?



Looping over a list of ints

We can loop through a list:

```
numbers = [1, 2, 3, 4]
for i in numbers:
    print(i)
```

What's going to happen?

```
>>> 1
>>> 2
>>> 3
>>> 4
```

- Each item of the list takes a turn at being the variable `i`
- Do the body once for each item
- We're done when we run out of items!



Practice Time!



1. Make a new file called yummy.py

2. Copy in this list

```
>>> fruits = ['apple', 'banana', 'mango']
```

3. Add **2 lines of code** that makes your program print out this.
Use a for loop!

```
>>>Yummy apple
```

```
>>>Yummy banana
```

```
>>>Yummy mango
```

HINT!

```
numbers = [1, 2, 3, 4]
for i in numbers:
    print(i)
```




How does it work??

Somehow it knows how to get one fruit out at a time!!


It's like it knows english!

```
fruits = ['apple', 'banana', 'mango']  
for fruit in fruits:  
    print('yummy ' + fruit)
```



But fruit is just a variable! We could call it anything! Like dog!

```
fruits = ['apple', 'banana', 'mango']  
for dog in fruits:  
    print('yummy ' + dog)
```




```
>>>Yummy apple  
>>>Yummy banana  
>>>Yummy mango
```



How does it work??

Everything in the list gets to have a turn at being the dog variable




```
fruits = ['apple', 'banana', 'mango']  
▶ for dog in fruits:  
    print('yummy ' + dog)
```

Let's set dog to to the **first** thing in the list!
dog is now 'apple'!



How does it work??

Everything in the list gets to have a turn at being the dog variable



```
fruits = ['apple', 'banana', 'mango']  
▶ for dog in fruits:  
    print('yummy ' + dog)
```

Let's set dog to to the **first** thing in the list!

dog is now 'apple'!

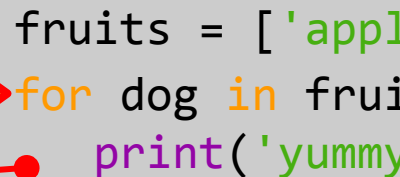
```
print('yummy ' + dog)
```

>>>Yummy apple



How does it work??

Everything in the list gets to have a turn at being the dog variable



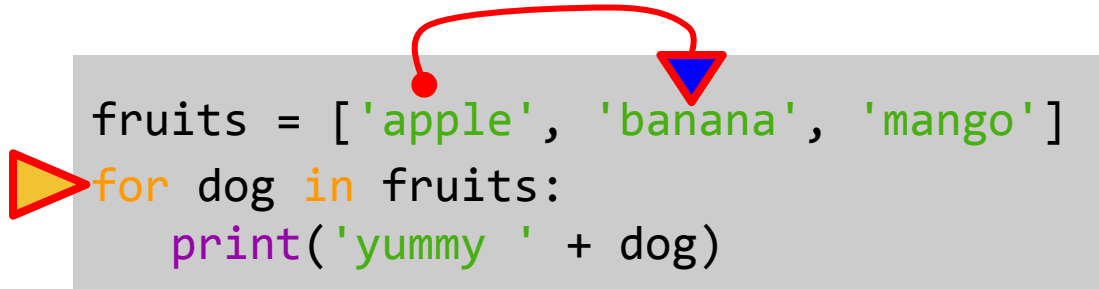
```
fruits = ['apple', 'banana', 'mango']  
for dog in fruits:  
    print('yummy ' + dog)
```

```
>>>Yummy apple
```

Let's set dog to to the **first** thing in the list!
dog is now 'apple'!
print('yummy ' + dog)
We're at the end of the loop body, back to the top!

How does it work??

Everything in the list gets to have a turn at being the dog variable



```
fruits = ['apple', 'banana', 'mango']  
▶ for dog in fruits:  
    print('yummy ' + dog)
```

```
>>>Yummy apple
```

Let's set dog to to the **first** thing in the list!
dog is now 'apple'!
print('yummy ' + dog)
We're at the end of the loop body, back to the top!

Let's set dog to to the **next** thing in the list!
dog is now 'banana'!



How does it work??

Everything in the list gets to have a turn at being the dog variable

```
fruits = ['apple', 'banana', 'mango']  
for dog in fruits:  
    print('yummy ' + dog)
```

```
>>>Yummy apple  
>>>Yummy banana
```

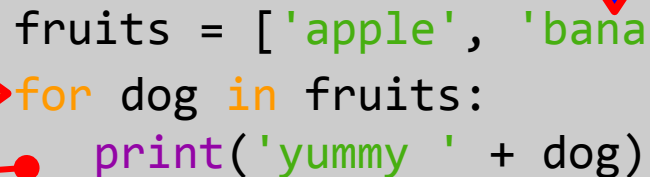
Let's set dog to to the **first** thing in the list!
dog is now 'apple'!
print('yummy ' + dog)
We're at the end of the loop body, back to the top!

Let's set dog to to the **next** thing in the list!
dog is now 'banana'!
print('yummy ' + dog)



How does it work??

Everything in the list gets to have a turn at being the dog variable



```
fruits = ['apple', 'banana', 'mango']  
for dog in fruits:  
    print('yummy ' + dog)
```

```
>>>Yummy apple
```

```
>>>Yummy banana
```

Let's set dog to to the **first** thing in the list!

dog is now 'apple'!

```
print('yummy ' + dog)
```

We're at the end of the loop body, back to the top!

Let's set dog to to the **next** thing in the list!

dog is now 'banana'!

```
print('yummy ' + dog)
```

Out of body, back to the top!



Project Time!



Now you know how to use a for loop!

Try to do Extension Part 6-7!

The tutors will be around to help!



Tell us what you think!

Click on the
End of Day Form
and fill it in now!