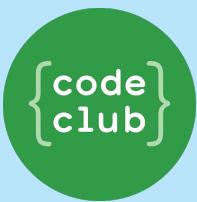


Python 1

ASCII Art



All Code Clubs must be registered. Registered clubs appear on the map at codeclub.org.uk - if your club is not on the map then visit jumpto.cc/18CpLPy to find out what to do.

Introduction:

Python allows you to turn a series of instructions into useful programs and fun games! In this project you'll learn how to run a Python program, and how to print text to the screen.



Activity Checklist

Follow these **INSTRUCTIONS** one by one



Test your Project

Click on the green flag to **TEST** your code



Save your Project

Make sure to **SAVE** your work now

Step 1: Saying hello

Activity Checklist

- Let's start by writing a very simple program, just so that you know how to get a Python program running. Open the IDLE program editor:

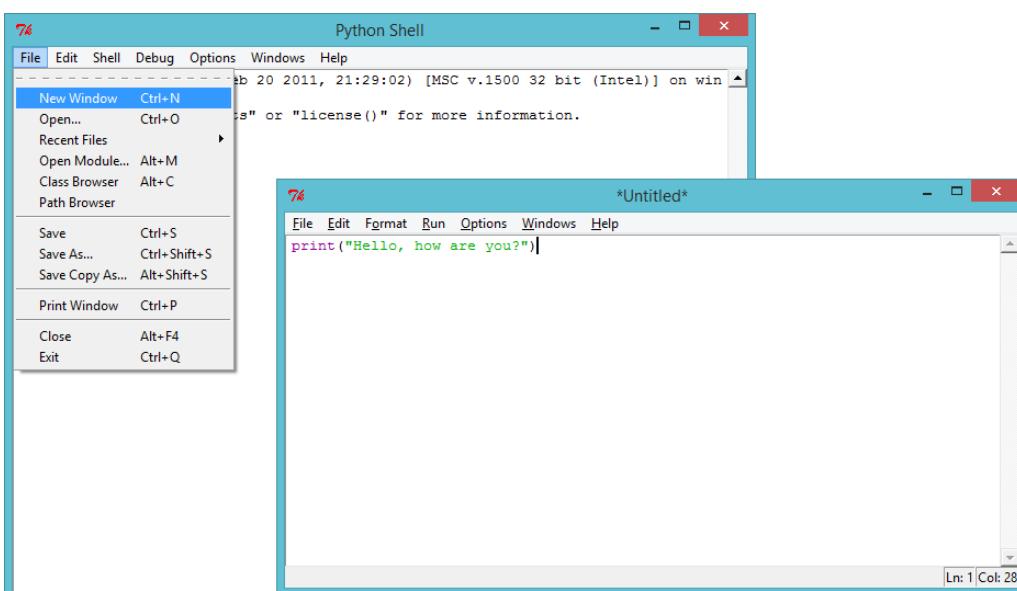
- On Windows, find IDLE in the start menu;
- On Mac, open up Terminal.app and type `idle` and press enter;
- On Linux, open up a Terminal, and type `idle` and press enter.

- Click `File → New Window`, and type the following into the window that appears:

```
print("Hello, how are you?")
```

This program will print some text to the screen. Notice that the text you want to print is surrounded by speech marks (`"`).

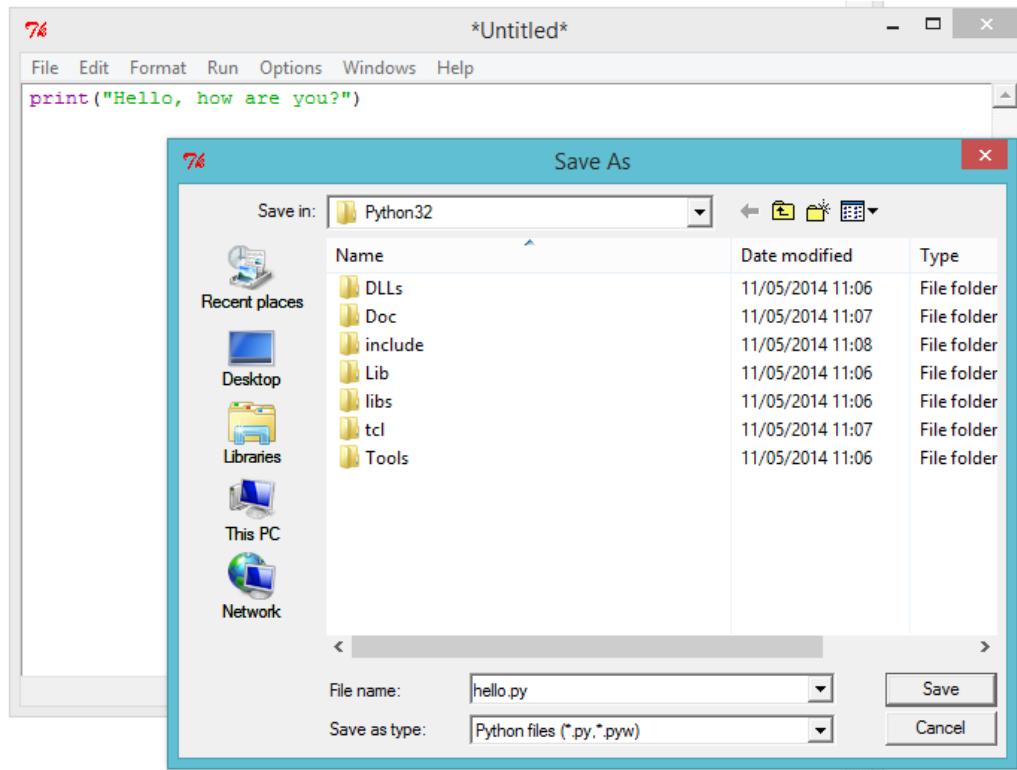
Here's an image showing what you need to do:



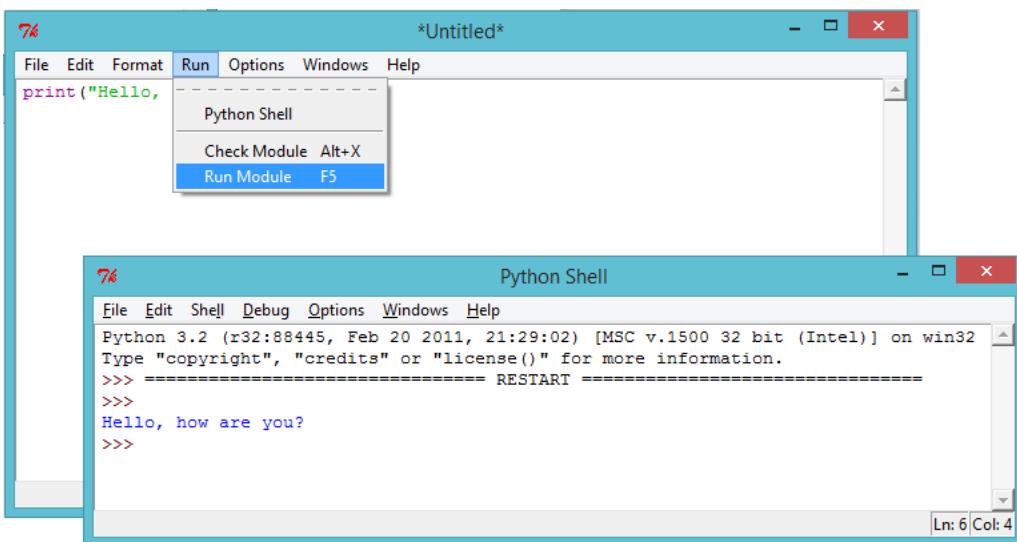
- Save the file, by clicking `File → Save`, and name the file



`hello.py` or something similar. Don't forget to type the `.py` bit at the end, which tells the computer that it's a Python file. Without it, your program won't be colour coded, which can be really helpful.



4. Run the file by clicking `Run → Run Module`. You should see another window appear, which is the Python shell. This is the place that your program will run. If everything has worked properly, you should see your text printed to the screen.



5. If you've made a mistake, for example missing out a speech



mark (""), then you'll get an error message instead, telling you what went wrong! Try it!

A screenshot of a Windows-style application window titled "hello.py - C:/Python32/hello.py". The menu bar includes File, Edit, Format, Run, Options, Windows, and Help. The main text area contains the Python code: `print("Hello, how are you?")`. A red horizontal bar highlights the closing parenthesis at the end of the string literal. A modal dialog box titled "SyntaxError" is displayed, showing the error message "EOL while scanning string literal" with a red X icon and an "OK" button. The status bar at the bottom right shows "Ln: 1 Col: 27".

6. Congratulations, you are now officially a Python programmer!
Give yourself a pat on the back (or if you're feeling lazy, get someone else to do it for you).



Save Your Project

Challenge: What's on your mind?

Change the program above to print something more interesting to the screen!

A screenshot of a Windows-style application window titled "Python Shell". The menu bar includes File, Edit, Shell, Debug, Options, Windows, and Help. The text area shows the Python shell prompt and output:
`Python 3.2 (r32:88445, Feb 20 2011, 21:29:02) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
I'm hungry!
>>> |`



Save Your Project

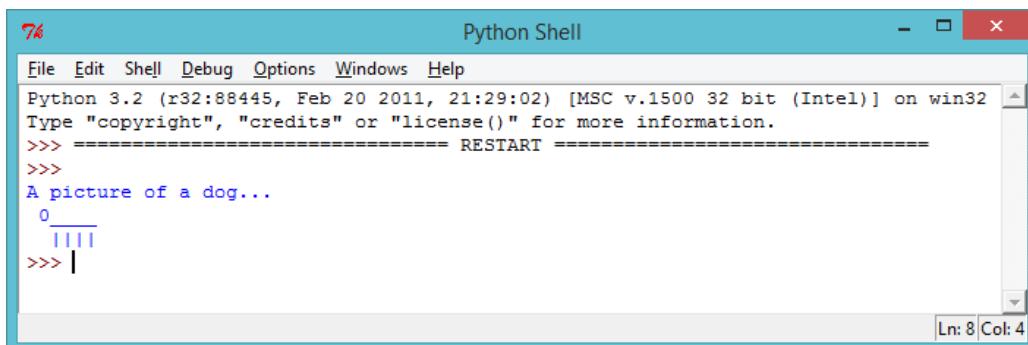
Step 2: About yourself

Activity Checklist

1. Let's print something much more fun than text... ASCII art!



ASCII art is creating pictures out of text. Here's an example - it's meant to be a dog!



A screenshot of the Python Shell window. The title bar says "Python Shell". The menu bar includes File, Edit, Shell, Debug, Options, Windows, and Help. The main window shows the following text:
Python 3.2 (r32:88445, Feb 20 2011, 21:29:02) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
A picture of a dog...
 0_____
 |||||
>>> |

Ln: 8 Col: 4

To make this masterpiece, you can type the following into the IDLE editor and run the program:

```
print("A picture of a dog...")  
print(" 0____ ")  
print(" ||||| ")
```

2. If you prefer, you can use 3 single quotes ('''') instead of speech marks, which allows you to print multiple lines of text with 1 print statement. Like this:



```
print('''  
A picture of a dog...  
 0_____  
 |||||  
''')
```

If you run this program, you'll see it prints the same dog as before.



Save Your Project

Challenge: About yourself

Write a Python program to tell others about yourself, by using text and ASCII art. You can create images of your hobbies, friends, family... anything you want! Here's an example:

The screenshot shows a Windows-style application window titled "Python Shell". The title bar includes standard window controls (minimize, maximize, close) and the title "Python Shell". The menu bar contains "File", "Edit", "Shell", "Debug", "Options", "Windows", and "Help". Below the menu is a status bar with "Ln: 9 Col: 0". The main window displays Python code and its output. The code includes a copyright notice, a restart message, and two ASCII art representations. The first ASCII art depicts a sheep with the text "My favourite animals are sheep:" above it. The second ASCII art depicts a house with the text "I live in Glasgow:" above it.

```
>>> Python 3.2.5 (default, May 15 2013, 23:06:03) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
My favourite animals are sheep:

o-###-
 | | #
I live in Glasgow:

 _\_
| # |
 \_ / |
 | # | #
 | # | #
>>>
```



Save Your Project

Step 3: Calculating text



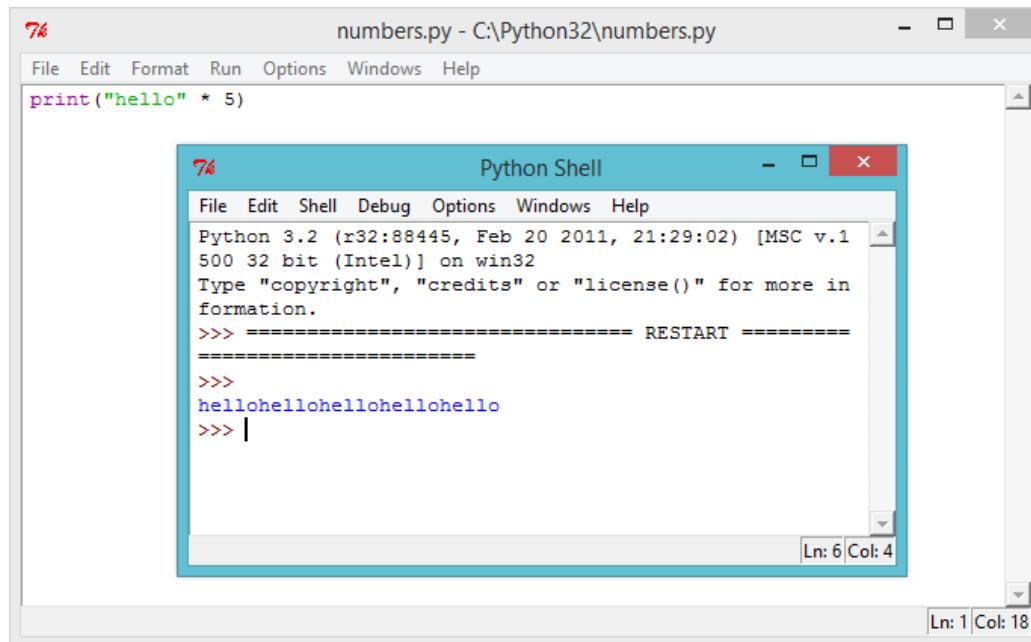
Activity Checklist

1. Python can also do calculations using text! What do you get if you multiply "hello" by 5? Let's ask Python, by running this program:

```
print("hello" * 5)
```



The star `*` in the program above is a multiply sign. Run the program above, and you should see the answer:



A screenshot of a Windows desktop environment. In the foreground, there is a code editor window titled "numbers.py - C:\Python32\numbers.py". The code inside is `print("hello" * 5)`. Behind it, a Python Shell window titled "Python Shell" is open, showing the output of the code: `Python 3.2 (r32:88445, Feb 20 2011, 21:29:02) [MSC v.1`, followed by a series of "hello"s, and then `>>> ===== RESTART =====`. The Python Shell window has status bars at the bottom indicating "Ln: 6 Col: 4" and "Ln: 1 Col: 18".

2. You can make the printed text above easier to read, by putting a space after the word `"hello"` in your program:

```
print("hello " * 5)
```

Run this program and you'll see that the output is a little easier to read than before.

3. If `"hello "` multiplied by 5 is `"hello hello hello hello hello "`, then what is `"hello" - 7`? Does this calculation even make sense?

A screenshot of the Python 3.2 IDE. The code editor window shows the file 'numbers.py' with the line `print("hello" - 7)`. Below it, the Python Shell window shows the output of running the program, which results in a `TypeError: unsupported operand type(s) for -: 'str' and 'int'` error. The status bar at the bottom indicates 'Ln: 1 Col: 18'.

Oops, you've broken it! Instead of an answer, we get an error message. It looks like that calculation doesn't make sense in Python!

4. How about addition? What answer do you think `"hello " + "world"` would give? Try it out, by running the following program:

```
print("hello " + "world")
```

A screenshot of the Python 3.2 IDE. The code editor window shows the file 'numbers.py' with the line `print("hello " + "world")`. Below it, the Python Shell window shows the output of running the program, which is `hello world`. The status bar at the bottom indicates 'Ln: 6 Col: 4'.

Does it give you the answer you expected?



Save Your Project

Challenge: Words and numbers

What does the following program print to the screen? See if you can guess correctly before running the program.

```
print("ha "*4)
print("ba" + "na"*2)
print("He" + "l"*2 + "o" + "!"*10)
```

Can you make up any words of your own?



Save Your Project

Step 4: ASCII patterns



Activity Checklist

- Now that you know how to do calculations on text, now what?



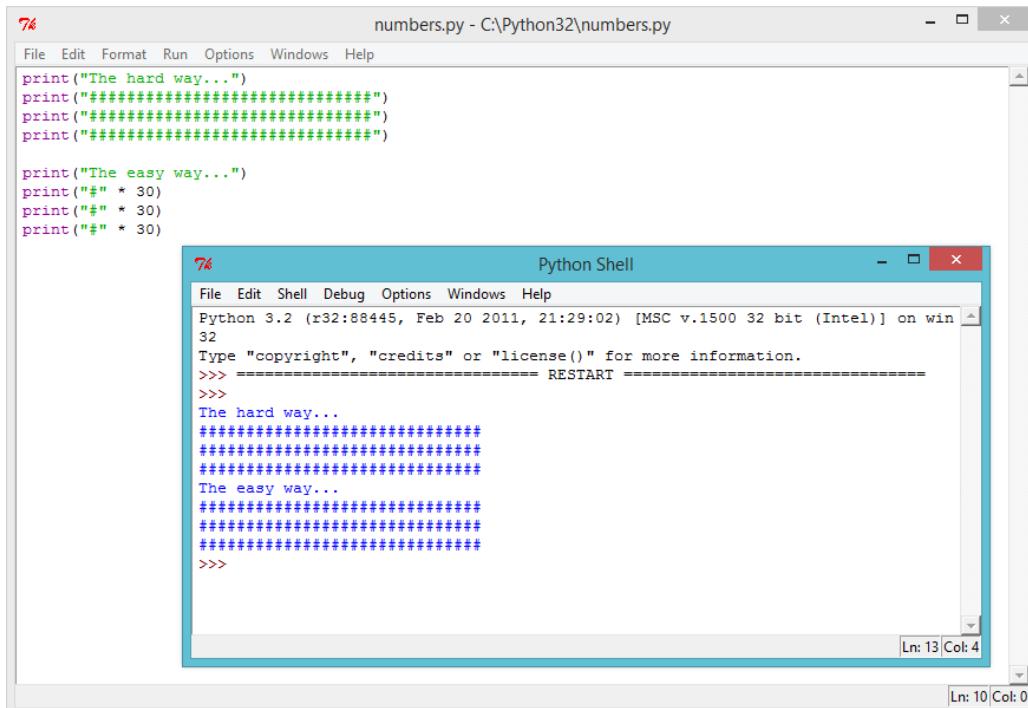
Why is it useful? Well, let's say you wanted to draw an ASCII art rectangle that is 30 characters long and 3 characters high. You could either draw it the hard way, like this:

```
print("#####")
print("#####")
print("#####")
```

Or you could save time and draw it the easy way, like this:

```
print("#" * 30)
print("#" * 30)
print("#" * 30)
```

Both give you exactly the same rectangle printed to the screen:



The screenshot shows two windows side-by-side. The left window is titled "numbers.py - C:\Python32\numbers.py" and contains the following code:

```
print("The hard way...")
print("#####")
print("#####")
print("#####")

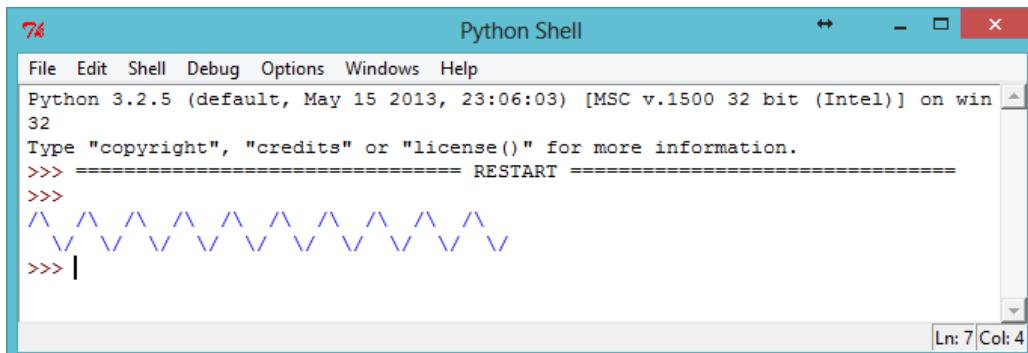
print("The easy way...")
print("#" * 30)
print("#" * 30)
print("#" * 30)
```

The right window is titled "Python Shell" and shows the output of running this script. It prints two identical rectangles to the screen:

```
The hard way...
#####
#####
#####
The easy way...
#####
#####
#####
>>>
```

2. You could even use calculations to make interesting patterns, like this wave:

```
print("/\" *10)
print(" \\"*10)
```



The screenshot shows a single "Python Shell" window with the following code and its output:

```
print("/\" *10)
print(" \\"*10)
```

The output displays a wave pattern made of backslash (\) and forward slash (/) characters:

```
>>> |
```



Challenge: Code a scarf

Your best friend is having an 11th birthday party, and as a gift you've decided to code them a scarf! Use calculations wherever possible to make your own scarf pattern.

If you're feeling generous, you could even code them a cake (including 11 candles) to go with it!

