ENGG1811 Computing for Engineers

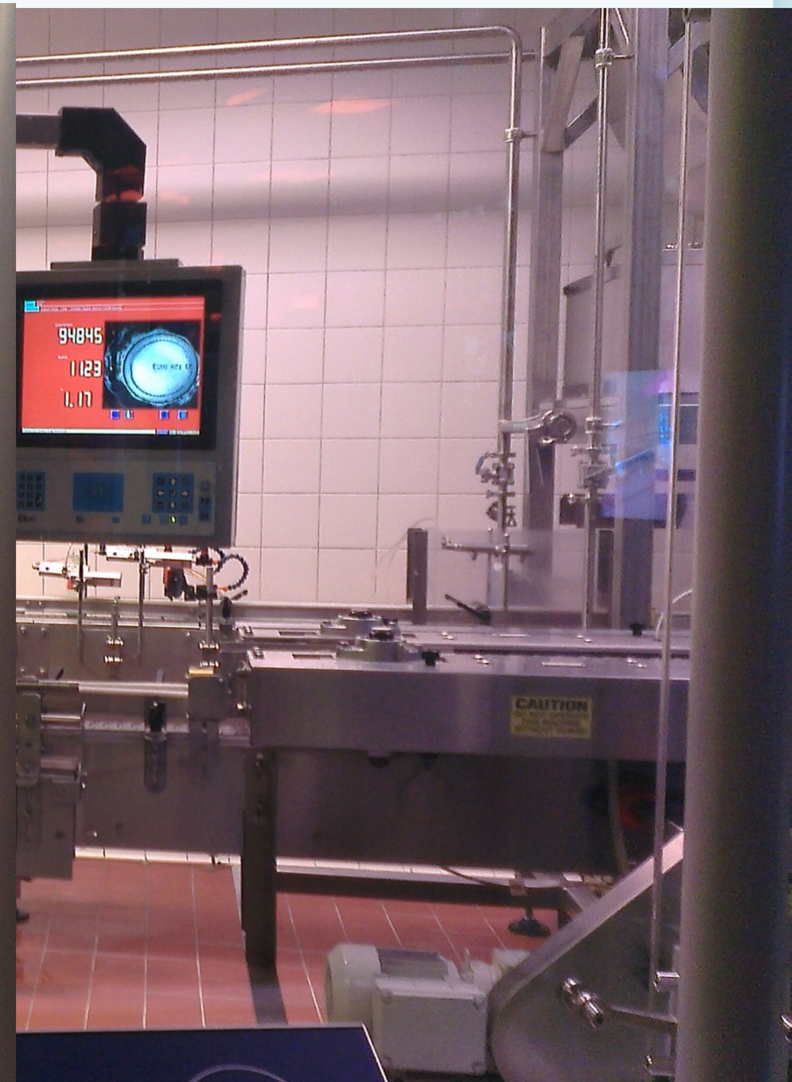
Week 1

Introduction to Programming and Python

Computers have changed engineering …

http:/ … 48/410.html

Computers have changed engineering …



How computing is used in engineering?

- *Automation* is a major application of computing in engineering
  - There are many other applications of computing in engineering. More to come.
  - Message: *Computing will play a key role in addressing grand challenges in engineering, e.g. aging infrastructure, etc.*
  - http://www.engineeringchallenges.org

- Automation: Computers/machines repeatedly performing the same procedure
  - Procedure: a sequence of *instructions*

Problem solving

- Engineering: invention, problem solving, …
- Problem solving requires you to understand how things work, test out ideas etc.
- How can you use computers to solve problems for you?
  - How can you use computers to understand, investigate, test and design?
  - A key idea is abstraction. You can use the same computing method to count the number of heart beats, the breathing rate, number of walking steps

Programming

- If you come out with a method for the computer to solve a problem, you need to be able to tell the computer how to do it.
  - You need to give instructions to computers

- Programming skill: The ability to give instructions to computers to perform the intended tasks

Python

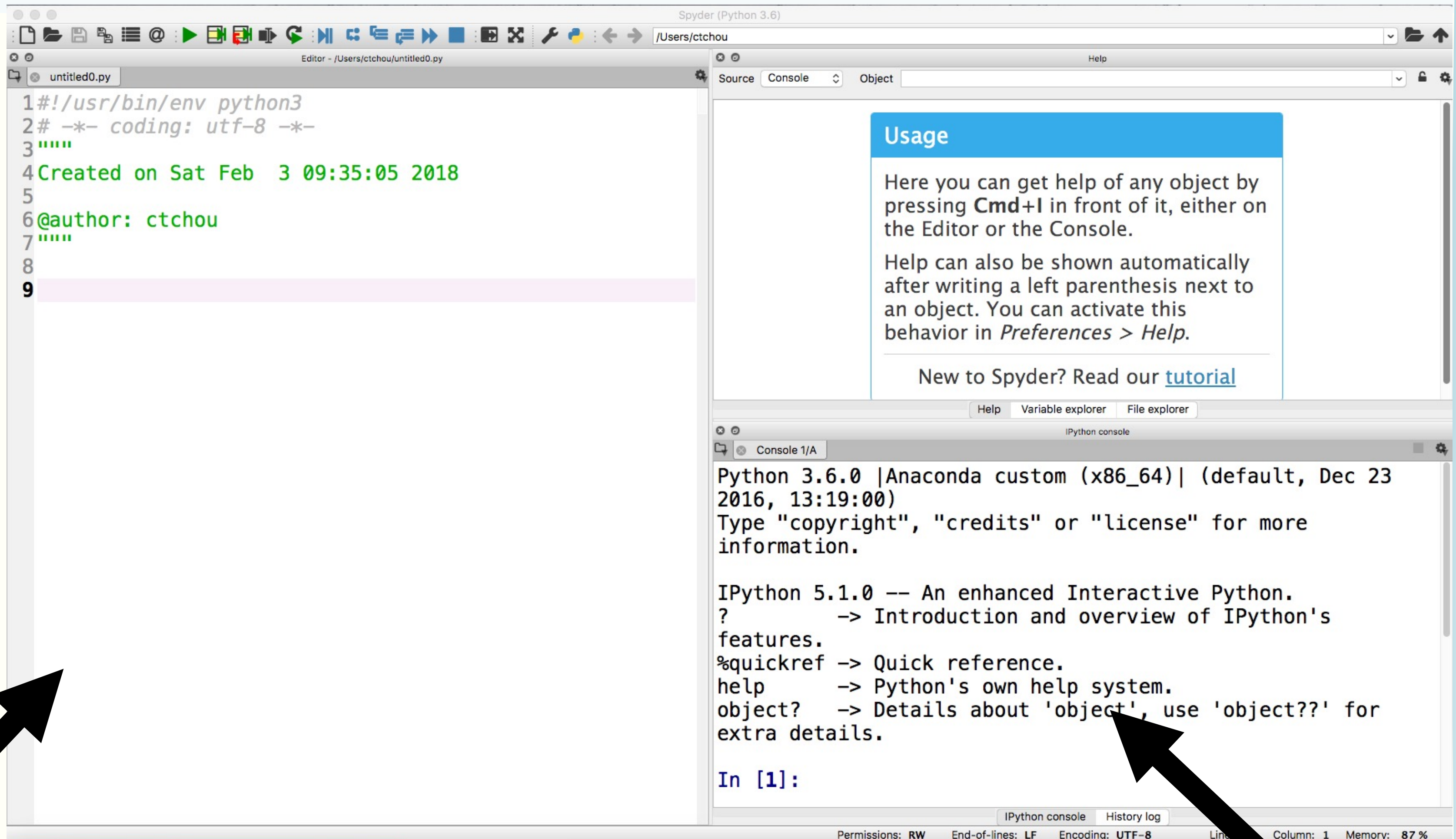
- Python will be the programming language that you will use to learn how to give instructions to computers
- It is a popular programming language and it comes with a lot of extra packages that help you to do engineering work
- There are two versions of Python. We will be using **Python 3**, not Python 2.

Spyder

- We will use a program called Spyder to develop, test and run Python programs
- Spyder is available on all UNSW CSE computers
- You will also use Spyder in the lab
- If you want to use Spyder on your computer, your options are:
  - Install Anaconda on your computer
  - Use the UNSW CSE computers remotely. This requires Internet access.
  - More details in the Getting Started section of the course website

The Spyder Environment

Buttons

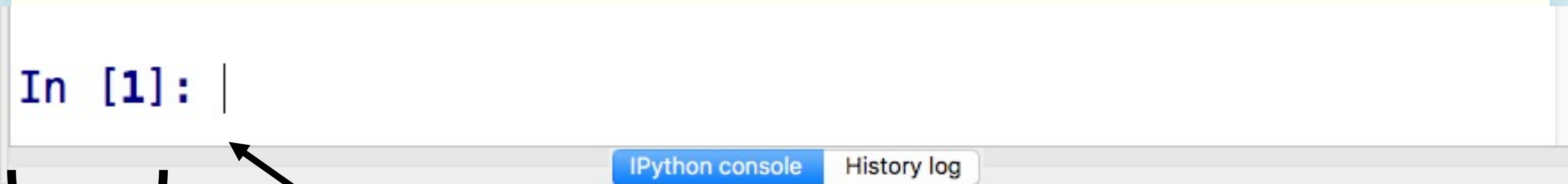


Editor for developing Python programs

iPython Console
'i' is short for interactive

Using the iPython Console

- We will simply call it the console
- You can use the console to do some simple programming
- You do that by typing commands at the prompt
  - Commands are instructions to tell the computers to do something



The prompt

You type the command at the blinking cursor. After you've finished typing, use the Enter key to tell the console to execute the commands.

If you haven't got Spyder yet,

- You can use iPython Console online at:
  - https://www.pythonanywhere.com/try-ipython/
  - https://trinket.io/console

- We will only be using iPython Console today but we will use the editor in the next lecture. So make sure you install Anaconda before that.
  - Instructions on installing Anaconda for Python 3.6 can be found under Getting Started on the course website

Using console to do arithmetic

- Type 3+4 at the console, as follows:

```
In [1]: 3 + 4
```

IPython console | History log

- And then type the Enter key
- The computer execute the instruction, which is to add 3 and 4
- The console returns the answer

```
In [1]: 3 + 4
Out[1]: 7

In [2]:
```

IPython console | History log

Arithmetic Operators in Python

Operator	Description
+	Addition or unary plus
–	Subtraction or unary minus
*	Multiplication
/	Floating point division
//	Integer division (fraction discarded)
%	Integer modulus (remainder)
**	Exponentiation (power)

Exercises:

- Type the following at the prompt and then execute the command, observe what you get and try to understand the meaning of the arithmetic operators

```
2 * 4
2 ** 4
10 / 7
10 // 7
10 % 7
10 - -7
```

Unary and binary operations

- + and – can be unary or binary
- For example,

10 - -7

Binary minus
= Subtract 2 numbers

Unary minus
= Negative sign

Precedence

- You can use the arithmetic operators to calculate complicated expressions
- You can type: `1 + 2 * 3 - 4`
  - Should this be 3 or 5?
- The computers evaluate arithmetic expressions according to the rule of precedence

Precedence

- When evaluating arithmetic expressions, order of evaluating operations determined by ***precedence***

Operator
( )
**
+ – (unary: sign)
* / % //
+ - (binary)

Higher precedence

Lower precedence

- You do not need to memorise this. Look it up when you need. We will give this to you in the exam.

Evaluating Expressions – Rules of Precedence

- When evaluating expressions, operations of higher precedence are performed before those of lower precedence

```
2 + 3 * 4 = 2 + (3 * 4) = 14
```

- Otherwise operations performed from **left to right**

```
30 // 4 % 2 = (30 // 4) % 2 = 7 % 2 = 1
```

- The order of precedence for multiple powers works from **right to left**

```
2 ** 3 ** 4 = 2 ** (3** 4) = a large number
```

- Use parentheses if in any doubt

Quiz:

- You want to calculate:

$$\frac{20}{5 \times 2}$$

- Which one can you not use?

a) 20 / 5 / 2

b) 20 / 5 * 2

c) 20 / (5 * 2)

Quiz

- What is -2**2 in Python?

a) 4 i.e. (-2)**2

b) -4 i.e. –(2**2)

Operator
( )
**
+ - (unary: sign)
* / % //
+ - (binary)

Higher precedence

Lower precedence

An exception to the rule

- If a unary – or + is to the right of **, then the unary is evaluated first

- 10**-2 = 0.01

Variables and the assignment operator

- Type the following at the prompt and enter

```
In [9]: y = 5
```

- You can use y again to do computation

```
In [9]: y = 5

In [10]: 7 * y
Out[10]: 35

In [11]: y / 2
Out[11]: 2.5

In [12]: y
Out[12]: 5
```

- We say we *assign* the value of 5 to the *variable* named y
- We call = the assignment operator
- Each input you type in is a Python statement

Programming element: Variables

- Variables are stored in computer memory
- A variable has a name and a value
- A mental picture is:

Variable name	Value of variable
y	5

A program manipulates variables to achieve its goal

Note: This is a simplified view. We will introduce the more accurate view later in the course.

Expressions of variables

- You can combine variables in an expression
- Try this in the console:

```
In [24]: b = 2; c = 5; d = 10;

In [25]: f = (d/c)**b

In [26]: f
Out[26]: 4.0

In [27]: d = c**b

In [28]: d
Out[28]: 25
```

Old value of the variable d is overwritten

Execution of arithmetic expressions

- Variables are stored in memory

Name of variables	Value of variables
b	2
c	5
d	10

```
d = c ** b
```

1. Look up the value of c and b
2. Compute c to the power of b
3. Store the result in the memory for d

Assignment errors

```
In [32]: x - 6
Traceback (most recent call last):

  File "<ipython-input-32-86a84d68a48a>", line 1, in <module>
    x - 6

NameError: name 'x' is not defined
```

You must assign a value to a variable before using it

```
In [31]: c**b = d
  File "<ipython-input
    c**b = d
        ^
SyntaxError: can't assign to operator
```

Order is important.
Destination = source

Variable names are case sensitive / debugging

```
In [100]: num = 2.15

In [101]: Num
Traceback (most recent call last):

  File "<ipython-input-101-0b87731cbe51>", line 1, in <module>
    Num

NameError: name 'Num' is not defined
```

- You should read the error message and try to understand what it means so that you can fix your own code later on
  - Programmers use the term debugging to mean fixing the code. See below for a discussion on the origin of the term and a picture of the moth which apparently stopped a computer program from its execution
  - https://en.wikipedia.org/wiki/Debugging
- Don’t get upset if you get bugs in your code. It’s a fact of life in computer programming. What is important is you learn how to debug.

Don't interpret assignment as equal sign

- In mathematics, the expression $x = x + 10$ is a contradiction

- In computer programming, = is the assignment operator, so `x = x + 10` means the following operations

```
In [34]: x = 7

In [35]: x = x + 10

In [36]: x
Out[36]: 17
```

Take the value of the variable x (which is 7), add 10 to it and assign the result (which is 17) to the variable x

Quiz

- What is the value of the variable x after executing the following statements?

```
x = 10
x = x + 2
x = x + 2
x = x + 2
```

Try yourselves

- You can also try these

```
x = 10
x = x * x
x = x % 3
x = 2 / (x+7)
```

Numbers and text

- Computers can handle numbers
  - Engineering data are often in numbers
  - Data processing is important in Engineering
  - Numbers can also be used to represent
    - Images: Photos, X-ray images, medical images
    - Videos, music, speeches etc.
- Computers can also handle text
  - Data can also be in text

Strings

- In Python, text is represented as strings
- Strings are always enclosed within a pair of matching single quotes or double quotes

Strings: examples

```
In [6]: s = 'U'

In [7]: s
Out[7]: 'U'

In [8]: my_uni = 'UNSW'

In [9]: my_uni
Out[9]: 'UNSW'

In [10]: liar = 'He said that he was born on 29/02/2003. What a liar!'

In [11]: liar
Out[11]: 'He said that he was born on 29/02/2003. What a liar!'
```

- The variable s is a string of one character
- The variable my_uni is a string with 4 characters

String manipulations

- You can
  - Concatenate strings using +
  - Repeat strings using *

```
In [15]: str1 = 'He is a '; str2 = 'great violinist'

In [16]: str3 = str1 + str2

In [17]: str3
Out[17]: 'He is a great violinist'
```

- Try the following yourselves

```
In [19]: num_ten = 10; 'This is ' + 'so ' * num_ten + 'yummy!'
```

Limitation of the console

- You have used the console to
  - Assign variables
  - Perform some simple computation
  - Manipulate strings
- The console is good for testing one or few lines of statement
- A more powerful method is to put the Python statements into a file, or a Python program

Program to convert Fahrenheit to Celsius

- We will write a program to convert a temperature F in Fahrenheit to its equivalent temperature C in Celsius
- The temperatures F and C are related by

$$(F - 32)\frac{5}{9}$$

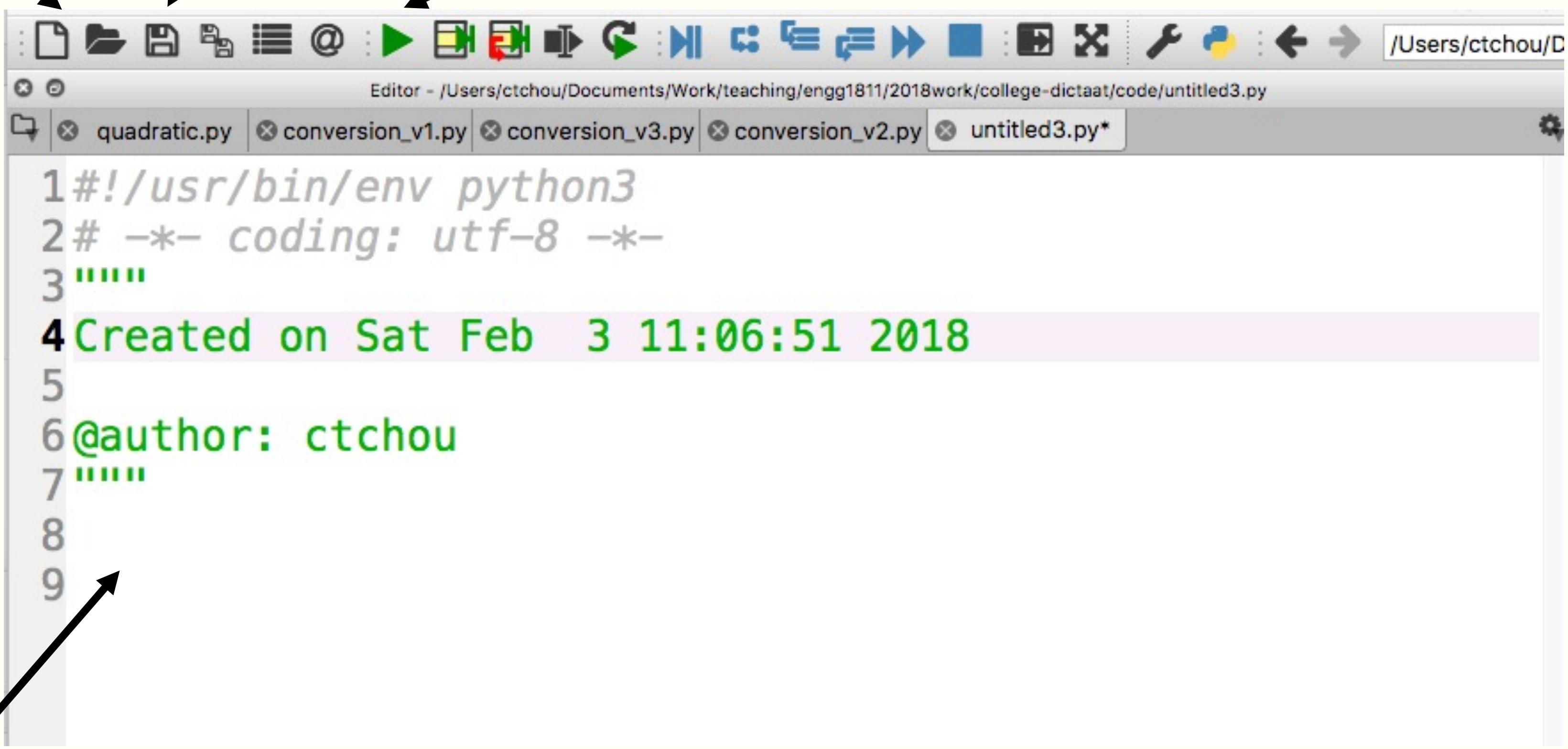
- We will develop the program step by step
- We will type the program using the editor in Spyder

The Spyder editor

New file

Save file

Run file



Start typing in program here

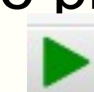
F to C conversion (version 1)

```
temp_fahrenheit = 80

temp_celsius = (temp_fahrenheit - 32) * (5/9)

print(temp_fahrenheit,' in F = ',temp_celsius,' in C')
```

- Tip for typing: the Tab key can complete variable name for you

- After typing the program, you can run the program using the run button 
  - Spyder will ask you to save the file first. Do give the program a meaningful name.
  - Note that Python programs have the extension .py
  - Don’t forget to save the file regularly when you work on Spyder
- Results will be displayed in the console

The print function

```
temp_fahrenheit = 80

temp_celsius = (temp_fahrenheit - 32) * (5/9)

print(temp_fahrenheit,' in F = ',temp_celsius,' in C')
```

- print is a function in Python to display results
- Any text within single quotes will be displayed as is
  - You can also use double quotes. They are strings.
- If print sees a variable name, it will display the value of the variable
- The displayed output is the concatenation of the parts separated by commas

Program execution

```
9  temp_fahrenheit = 80
10
11 temp_celsius = (temp_fahrenheit - 32) * (5/9)
12
13 print(temp_fahrenheit,' in F = ',temp_celsius,' in C')
```

- This program consists of 3 statements
  - At lines 9, 11 and 13
- The statements are executed in the order that they appear

Identifiers

Words like temp_celsius in the example program are called **identifiers**

- Identifiers are used for names of variables
- Identifiers are sequences of letters (a-z, A-Z), digits (0-9) and underscores (_)
- Identifier can only begin with a letter
- Examples of valid identifiers

```
module1  x42   temp   y_origin
```

Quiz: Which of the following identifiers are valid?

```
day 2day day_of_the_week day2 $24 see-saw
```

Keywords

- Python has a number of keywords or reserved words
- You cannot use them as variable names
- Don't worry about memorising them now, you will see them a lot later on and will know them as your friends ☺

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

https://www.programiz.com/python-programming/keywords-identifier

Rules for choosing identifiers

- Rule 1: Must be valid
- Rule 2: Avoid keywords

- The program will run if it doesn't violate Rules 1 and 2

- Rule 3: Choose **meaningful** identifiers

Identifier Conventions

- Identifier conventions have been devised to make programs more readable
  - Use meaningful variable names, most Python programmers use lower case words separated by underscore for readability

```
temperature     num_count
mass_in_kg          is_within_normal_range
```

  - OK to use short names for minor or short-lived data

Notes

- Software readability is an important issue. Here is a style guide to writing Python program, known as PEP8:
  - https://www.python.org/dev/peps/pep-0008/
  -
- Note that for some other computer languages, programmers use camel case as the style for identifiers
  - Camel case: first word is all lower case, the first letter of subsequent words in upper case, e.g. isWithinNormalRange, thisYear

F to C conversion (Version 2)

```
# The temperature in Fahrenheit to be converted
temp_fahrenheit = 80 # Change here if needed

# Convert to Celsius using standard formula
temp_celsius = (temp_fahrenheit - 32) * (5/9)

# Output the temperature in Celsius
print(temp_fahrenheit,' in F = ',temp_celsius,' in C')
```

- Comments are added to explain how a program works
  - All text after the # symbol is comment
- Comments are ignored when a program is executed
- Comments are for people to read

F to C conversion (version 3)

```python
# Constants
MELTING_POINT_FAHRENHEIT = 32
RATIO = 5/9 # Scaling factor for conversion

# The temperature in Fahrenheit to be converted
temp_fahrenheit = 80 # Change here if needed

# Convert to Celsius using standard formula
temp_celsius = (temp_fahrenheit - MELTING_POINT_FAHRENHEIT) * RATIO

# Output the temperature in Celsius
print(temp_fahrenheit,'in F = ',temp_celsius,'in C')
```

- Fixed or **constant** values are often required at several places in a program
- By giving a name to the constant…
  - The reader understands what the value *means*
    - for example, only hard-core physicists would recognise **1.3806503e–23** in a calculation (it's Boltzmann's constant)
- Name format convention: ALL_CAPS
- Define the constants at the beginning of the program

Why documenting a program

- Say, you've written a program that does some fabulous work for you. It is possible that you may need to modify it a few months later. You may have difficulty figuring out how you did it earlier if you haven't documented it

- Use Python docstrings

Python docstring

```
"""
Purpose: To convert temperature from Fahrenheit to Celsius

Author: Mary Poppins
Date: 3/2/18

Data Dictionary:
    temp_fahrenheit  temperature in Fahrenheit to be converted to
                     Celsius
    temp_celsius     temperature in Celsius (final result)

Method:
    Use the formula Celsius = (Fahrenheit - 32) * 5 / 9

"""
```

- Docstring is enclosed a pair of tripe double quotes or triple single quotes
- Spyder typesets it in green
- The contents are comments, i.e. not executed

Documentation

- Begin with:
  - Purpose, author, date
- Then data dictionary
  - list of variables used and how they are used
- Then problem parameter assignments if applicable
- Program description

- Expectations:
  - Lab programs must be reasonably documented
  - Documentation carries marks in assignments

Mathematical functions

- Standard Python has a limited set of maths operators: + - * / // % **
- Sometimes you want to use sin(), cos(), log(), exp(), etc.
- In Python, these operations are found in the math library

Example: Solving quadratic equation

- We will write a program to solve the quadratic equation

$$ax^2 + bx + c = 0$$

- using the formula

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

- We will use a function to compute the square root from the math library

Python code

```
31 # Import the math module - Need that for square root
32 import math
33
34 # Specify the coefficients of the quadratic equation
35 a = 2; b = 5; c = 1 # Enter the coefficients on this line
36
37 # Compute the square root of the discriminant
38 root_discriminant = math.sqrt(b**2-4*a*c)
39
40 # Compute the root
41 root1 = (-b + root_discriminant)/(2*a)
42 root2 = (-b - root_discriminant)/(2*a)
43
44 # Display the answers
45 print('The roots are ',root1,' and ', root2)
```

- You must import the math library before using its functions
- Line 40 shows the usage of math.sqrt()
  - Let us try some examples in the console

The math library

- The math library also contain functions for:
  - Trigonometry and radian/degree conversion
    - Radian is assumed
  - Exponential and log
  - Etc.
- The file math_examples.py contains examples

- For a complete list, see
  - https://docs.python.org/3/library/math.html
  - https://www.programiz.com/python-programming/modules/math

Summary

- Spyder development environment
  - iConsole, editor, program execution, saving files
- Programming
  - Arithmetic operators and precedence
  - Variables and naming convention
  - Assignment operator =
  - Statements are executed one after another in a computer program
  - Writing computer programs in a file
  - The math library