Variables

A variable is a named storage location in the computer’s memory. We use variables as a place to store information and to access the information we have stored.

Good Variable Names

1) Meaningful, descriptive, e.g. amount, age
2) Lowercase letters and numbers, e.g. day1 and day2
3) Cannot start with a number
4) Whole words rather than abbreviations, e.g. amount rather than a.
5) Separate words with underscore, e.g. amount_paid

Assigning a value to a variable

We use the assignment operator =

This means “put whatever is on the right hand side of the “=” into the variable on the left hand side”.

For good style, surround operators with a space.

Examples:

amount = 20

amount = amount * 2 + 9

Order of Evaluation (BEDMAS)

Brackets (  )
Exponents **
Division/Multiplication/Modulus / (floating point divide), // (integer divide), * (multiply), % (modulus (mod) sometimes called remainder)
Addition/Subtraction + (add), - (subtract)
Examples of Order of Evaluation

```python
import math

value1 = 3 + 4 + 7 / 2 + 7 // 2 + 5  # 18.5
value2 = 0.5 * 3 // 2 + 3 / 2  # 1.5
value3 = 4 + 2 // 3 + 2 ** 3 + (3 + 2)  # 17
value4 = 6 * 3 + math.pow(2, 3) // 2  # 22.0
value5 = math.floor(2.7) + math.ceil(3.2 * 2 )  # 9
value6 = 11 % 3 + math.floor(math.pi) * 10 + 2 % 3  # 34
```

Comments

Used (sparingly) to explain what a piece of code is doing if it is not obvious. Code should usually be self-documenting (by using good variable names, etc.) but comments can be used to explain any code that is not obvious. Comments begin with a hash sign and finish at the end of the line. For example:

```
weight = 63.2  # weight in kilograms
```

docstring

Special string of text to explain what the code does. It uses 3 double quotes to start and end the text. Example of a docstring at the beginning of a module:

```
"""
This program calculates the area of a circle.
"""
import math
radius = 10
area_of_circle = math.floor(math.pi * radius ** 2)
print("Area of circle with a radius of ", radius, 
is ", area_of_circle, ".", sep = ")
```