Python Basics

Comments

```
''' Comment
your code '''
```
multi-line comment

```
# Comment your code
```
single line comment

Variables

Variables are containers that store values. When naming variables remember the 'no caps, no gaps' rule.

Use a single '=' to assign a value to a variable.

```
first_prime=2
nz_capital="Wellington"
your_name=input("What is your name? ")
best_num=int(input("Choose a number: "))
```

Output

There are several ways to output data. One of the most convenient is to use the format method. The {} are place holders which are filled by the variables in the .format() part of the code.

```
fav_num=42
fav_colour="pink"

print("My favourite number is {} and my favourite \
colour is {}.".format(fav_num,fav_colour))
```

We can format numbers to 2dp by using {:.2f} as follows...

```
length=5
cost=25.50
price=cost*length

print("The price is ${:.2f}".format(price))
```

Escape Sequences

To split code over multiple lines, use the \ character

\n	new line	\t	tab
\'	single quote	\"	double quote

Operators

```
added = 3 + 2 # addition
minus = 3 - 2 # subtraction
times = 3 * 2 # multiplication
divide = 3 / 2 # 'normal' division
whole_only = 3 // 2 # shows whole number
remainder = 3 % 2 # calculates remainder
power = 3 ** 2 # for exponents
```

Comparing Values...

To compare values, use the following symbols. Note the double '==' that is used to find out if a variable is the same as another variable.

==	equal to	!=	not equal to
>	more than	>=	more than or equal to
<	less than	<=	less than or equal to

Use the above when creating while loops and if statements.

If / elif / else <decisions>

Use an 'if' block to make decisions. Note that if the first condition is not met, Python will look at the second condition and so on. Here's an example (note the colons and indenting)...

```
age=int(input("How old are you? "))
if age<0:
    print("That is impossible!")
elif age<=18:
    print("You are quite young")
elif age<=40:
    print("You are an adult")
elif age<65:
    print("You are 'over the hill'")
else:
    print("Congratulations - you win a Gold card")
```

Here is another example...

```
response = int(input("Choose a number between \
1 and 20: "))

if 1 <= response <= 20:
    print("Thank You")
else:
    print("Eeek!")
```

While Loops

Loop code while a condition is met. Everything that is indented is part of the loop.

```
keep_going="yes"
while keep_going=="yes":
    print("you chose to keep going")
    keep_going=input("Again?")
```

Make sure that there is a way for the loop to end. Usually this involves updating a counter or asking for user input (similar to the example above).

For Loops

Loop code a certain number of times. Useful for iterating through the values in a list.

```
for item in range (1,10,2):
    print(item)

pet_list=["cat","dog","mouse"]
for item in pet_list:
    print(item)
```

String Methods

Here are some useful string methods.

len(string)	Gives the length of a string
lower()	Converts string to lowercase
upper()	Converts string to uppercase
title()	Makes first letter of each word uppercase
replace(old,new,[max])	Replaces 'old' character in string with 'new' character. Can limit this to a maximum number of replacements
isdigit()	Can be used to check if string contains numbers only

The above is just a taste of the available methods. Please google / see the Python documentation for more information.

Number Operations

round(x[, n])	rounds a number to n digits. Defaults to 0dp if n not specified
abs(x)	makes 'x' positive
int(x)	makes 'x' an integer
float(x)	makes 'x' a float
pow(x,y)	'x' to the power 'y'

Data Validation (Try / Except)

Useful for making sure unexpected input does not cause your program to crash. Note the indents!

```
try:
    best_int=int(input("Best Integer?? "))
except ValueError:
    print("You did not enter an integer")
```

Common Errors

If you get a syntax error, try looking both at the line that has been identified and the line directly before it. Often the error is in the preceding line of code.

NameError	Variable name spelled incorrectly
EOL while scanning literal	Missing quotes at beginning / end of a string
Invalid Syntax	Missing colon / using '=' instead of '=='

Note: words which are in orange / purple are 'reserved' and errors will result if you try and use those words as variable names.

Python Short Hand

We often need to increase / decrease counters. Here is the short hand...

increase by 1	count+=1
decrease by 1	count-=1

The number to the right of the equals does not have to be one. This shorthand also works for multiplication.

Lists

Lists can be created as follows...

```
pet_list=["cat","dog","bird","mouse"]
```

Each item in a list is assigned a number based on its position in the list (starting with 0). Below are some methods that we can apply to a list...

```
# Find the position of 'dog'
print(pet_list.index("dog"))

# Print the first 3 items
print(pet_list[0:3])

# Print the last item
print(pet_list[-1])

# Add an item to a list
pet_list.append("tiger")
print(pet_list)

# Sort the list alphabetically
pet_list.sort()
print(pet_list)

# Reverse the list
pet_list.reverse()
print(pet_list)

# Find length of a list
print(len(pet_list))
```

List items can be lists (giving us a table-like structure)

```
results=[
    ["Jane",95,72,85,81,92],
    ["David",65,62,64,68,72],
    ["Aroha",75,72,81,68,77],
    ["Hone",68,71,75,73,81,72]
    ]

# prints the name and last result
# the first row
print(results[0][0],results[0][5])

# Loop prints names and last result for
# all students in list.
count=0
while count<len(results):
    print(results[count][0],results[count][5])
    count+=1
```

Functions

Functions allow us to reuse code and often result in programs which are easy to read and modify.

Functions ALWAYS go befor the main routine. They must include a 'return' statement. They can return either a single variable or a list. Here's how to define and then call a function.

```
# Function goes here (top of code)
def int_check(question,low,high):
    error = "Please enter an integer between \
{} and {}".format(low, high)
    valid=False
    while not valid:
        try:
            response=int(input(question))
            if low <= response <= high:
                return(response)
            else:
                print(error)

        except ValueError:
            print(error)

# Main routine starts here...

num_1=int_check("Choose 'low' <1 - 10>: ",1,10)
num_2=int_check("Choose 'high' <10 - 20>: ",10,20)

print()
print("Low: {} ".format(num_1))
print("High: {} ".format(num_2))
```

Random Stuff!

Below is some code you can use to generate random integers and choose random items from a list.

```
# import statements go here
import random

# Functions would go here

# Main routine below
secret_num = random.randrange(1,10)
print(secret_num)

items = ["rock", "paper", "scissors"]
rps_choice = random.choice(items)
print(rps_choice)
```