# Exercise 1 - For loops

**Question 1.**

In [2]:

```
a = 'python'
length = len(a)
for i in range(len(a)):
    print(a[i], i)
```

```
p 0
y 1
t 2
h 3
o 4
n 5
```

---

**Question 2.**

There are three mistakes here:

1. the second argument to `range` should be 11
2. the colon is missing
3. the indent is missing

In [6]:

```
total = 0
for i in range(1,11):
    total = total + 5 * i

print(total)
```

```
275
```

---

**Question 3**

In [7]:

```
n = 10
factorial = 1

for i in range(1, n+1):
    factorial *= i

print(factorial)
```

```
3628800
```

**Question 4**

In [2]:

```
num_cubes = 0

for n in range(1, 2000):

    if n**3 < 2000:
        num_cubes += 1

    else:
        break

print(num_cubes)
```

```
12
```

**Question 5.**

In [107]:

```
mult3 = "Fizz"
mult5 = "Buzz"
limit = 0

limit = int(input())

for i in range(1,limit+1):
    if i% 15 == 0:
        print(mult3+mult5)
    elif i%3 == 0:
        print(mult3)
    elif i%5 == 0:
        print(mult5)
    else:
        print(i)
```

```
5
1
2
Fizz
4
Buzz
```

# Exercise 2 - While loops

**Question 1.**

In [103]:

```
word = "Hello World"
target_letter = "W"
i = 0

while target_letter != word[i]:
    i += 1

print("Target letter is at position", i)
```

```
Target letter is at position 6
```

We need to increment $i$ to avoid an infinite loop

---

**Question 2.** It is best to use a `for` loop for this problem because we want to compare target_letter against all of the letters in word

In [104]:

```
word = "Hello World"
target_letter = "o"
i = 0

for l in word:

    if l == target_letter:

        print("Target letter is at position", i)

    i += 1
```

```
Target letter is at position 4
Target letter is at position 7
```

---

**Question 3.**

In [105]:

```
word.find('l')
```

Out[105]:

```
2
```

---

**Question 4.**

In [106]:

```
cumulative_sum=0
counter=0

while cumulative_sum<1000000:

    counter += 1

    cumulative_sum += (counter**2)

print("The answer is", str(counter))
```

```
The answer is 144
```

---

**Question 5.**

In [46]:

```
cumulativeSum=0
counter=0

while cumulative_sum<1000000:

    counter += 1

    cumulative_sum += (counter**2)

    print('The cumulative sum after', counter, 'terms is', cumulativeSum)

print("The answer is", str(counter))
```

```
The cumulative sum after 1 terms is 1
The cumulative sum after 2 terms is 5
The cumulative sum after 3 terms is 14
The cumulative sum after 4 terms is 30
The cumulative sum after 5 terms is 55
The cumulative sum after 6 terms is 91
The cumulative sum after 7 terms is 140
The cumulative sum after 8 terms is 204
The cumulative sum after 9 terms is 285
The cumulative sum after 10 terms is 385
The cumulative sum after 11 terms is 506
The cumulative sum after 12 terms is 650
The cumulative sum after 13 terms is 819
The cumulative sum after 14 terms is 1015
The cumulative sum after 15 terms is 1240
The cumulative sum after 16 terms is 1496
The cumulative sum after 17 terms is 1785
The cumulative sum after 18 terms is 2109
The cumulative sum after 19 terms is 2470
```

---

**Question 6.**

In [12]:

```
word = "Hello World"
target_letter = "W"
enum = list(enumerate(word))

for e in enum:

    if e[1] == target_letter:

        print("Target letter is at position", e[0])

        break
```

```
Target letter is at position 6
```

---

**Question 7.**

In [14]:

```
nums = [1,2,3]

total = 0

for n in nums:
    total += n

mean = total/len(nums)

print(mean)
```

```
2.0
```

---

# Exercise 3 - Modelling with loops

---

**Question 1.**

In [68]:

```
total = 100
counter = 0

while total < 400:
    counter += 1
    total = total * 1.05

print(f'After {counter} years the value of the account is £{round(total, 2)}')
```

```
After 29 years the value of the account is £411.61
```

**Question 2.**

In [97]:

```
u = 0
a = 9.81
for t in range(0, 21):
    t /= 10
    d = u*t + 1/2*a*t**2
    print(f'At time {t}, {round(d, 3)} m travelled')
```

```
At time 0.0, 0.0 m travelled
At time 0.1, 0.049 m travelled
At time 0.2, 0.196 m travelled
At time 0.3, 0.441 m travelled
At time 0.4, 0.785 m travelled
At time 0.5, 1.226 m travelled
At time 0.6, 1.766 m travelled
At time 0.7, 2.403 m travelled
At time 0.8, 3.139 m travelled
At time 0.9, 3.973 m travelled
At time 1.0, 4.905 m travelled
At time 1.1, 5.935 m travelled
At time 1.2, 7.063 m travelled
At time 1.3, 8.289 m travelled
At time 1.4, 9.614 m travelled
At time 1.5, 11.036 m travelled
At time 1.6, 12.557 m travelled
At time 1.7, 14.175 m travelled
At time 1.8, 15.892 m travelled
At time 1.9, 17.707 m travelled
At time 2.0, 19.62 m travelled
```

**Question 3.A:** The code to compute the 10th Fibonacci numbers is:

In [77]:

```
n = 10
a = 0
b = 1

print(a)
print(b)

for x in range(n-2):
    c = a + b # Calculate next Fibonacci number
    a = b     # Update values of last two Fibonacci numbers
    b = c
    print(c)
```

```
0
1
1
2
3
5
8
13
21
34
```

In [78]:

```
n = 10
fib = [0, 1]

for i in range(n-2):

    f = fib[i+1] + fib[i] # Calculate next Fibonacci number

    fib.append(f)         # Add Fibonaaci number to series

print(fib)
```

```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

---

**Question 3.B:** The code to compute the number of Fibonacci numbers is:

In [5]:

```
# first Fibonacci number
a = 0

# second Fibonacci number
b = 1

# number of Fibonacci numbers
counter = 2

while a + b < 100:

    # compute next Fibonacci number and update counter
    c = a + b
    counter += 1

    # update the values of a and b
    a = b
    b = c

print('There are', counter, 'Fibonacci numbers below 100')
```

```
There are 12 Fibonacci numbers below 100
```

In [6]:

```
while a + b < 1_000:

    # compute next Fibonacci number and update counter
    c = a + b
    counter += 1

    # update the values of a and b
    a = b
    b = c

print('There are', counter, 'Fibonacci numbers below 1,000')
```

```
There are 17 Fibonacci numbers below 1,000
```

In [7]:

```
while a + b < 10_000:

    # compute next Fibonacci number and update counter
    c = a + b
    counter += 1

    # update the values of a and b
    a = b
    b = c

print('There are', counter, 'Fibonacci numbers below 10,000')
```

```
There are 21 Fibonacci numbers below 10,000
```

You can make the code more efficient by using a `for` loop to cycle through question (b) i, ii, iii

In [8]:

```
# first Fibonacci number
a = 0

# second Fibonacci number
b = 1

# number of Fibonacci numbers
counter = 2

for i in [100, 1000, 10000]:

    while a + b < i:

        # compute next Fibonacci number and update counter
        c = a + b
        counter += 1

        # update the values of a and b
        a = b
        b = c

    print(f'There are {counter} Fibonacci numbers below {i}')
```

```
There are 12 Fibonacci numbers below 100
There are 17 Fibonacci numbers below 1000
There are 21 Fibonacci numbers below 10000
```

---

**Question 4.**

In [73]:

```
for i in range(1, 150):

    # Default status
    status = 'prime'

    for j in range(2, i):
        if i % j :
            pass

        else:
            status = 'not prime'
            break

    if status == 'prime' and i!=1:
        print(i, end=', ')
```

```
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71,
73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149,
```

In [1]:

```
for i in range(1, 150):

    # Default status
    status = 'prime'

    for j in range(2, i):
        if i % j == 0:
            status = 'not prime'
            break


    if status == 'prime' and i!=1:
        print(i, end=', ')
```

```
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71,
73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149,
```

In [14]:

```
for i in range(1, 150):

    for j in range(2,i):

        if i%j == 0:
            break

    else:
        if i!=1:
            print(i, end=', ')
```

```
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71,
73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149,
```

---

# Exercise 3 - More loops

---

**Question 1**

In [9]:

```
s = 0

for i in range(1,11):
    for j in range(0,6):
        s += j * j * (i + j)

print(S)
```

```
5275
```

---

**Question 2**

In [98]:

```
for i in range(1, 6):
    for j in range(i):
        print('*', end='')
    print()

for i in range(5, 0, -1):
    for j in range(i):
        print('*', end='')
    print()
```

```
*
**
***
****
*****
*****
****
***
**
*
```

---

**Question 3**. The code to compute $\pi_N$ for a fixed value of $N$ is:

In [99]:

```
# create a variable for the approximate value of pi
pi_N = 0

# number of terms in the series
N = 100

# use a for loop to compute the series
for n in range(N):
    pi_N += 8 / (4 * n + 1) / (4 * n + 3)

print(pi_N)
```

```
3.1365926848388144
```

# Exercise 5 - Lists

**Question 1.**

In [111]:

```
a = [1, 2]
b = [3, 4]
```

---

**Question 2.**

In [112]:

```
a[0] = 5
```

---

**Question 3.**

In [113]:

```
nested = [a, b]
print(nested)
```

```
[[5, 2], [3, 4]]
```

---

**Question 4.**

In [114]:

```
# the first loop; l is an element of nested so will be a list
for l in nested:
    # the second list; e is an element of the list l
    for e in l:
        print(e)
```

```
5
2
3
4
```

---

**Question 5.** See the list_words.py file

---

**Question 6.**

In [115]:

```
l = [e for e in range(101) if e % 2 != 0]
print(l)
```

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39,
41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 75, 77,
79, 81, 83, 85, 87, 89, 91, 93, 95, 97, 99]
```

In [116]:

```
l = [e for e in range(101) if e % 3 == 0]
print(l)
```

```
[0, 3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48, 51, 54, 57,
60, 63, 66, 69, 72, 75, 78, 81, 84, 87, 90, 93, 96, 99]
```

In [117]:

```
l = [e for e in range(101) if all(e % x != 0 for x in range(2, e)) and e > 1]
print(l)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71,
73, 79, 83, 89, 97]
```

# Exercise 6 - Tuples

---

**Question 1.**

In [130]:

```
fondue_ingredients = ('gruyere', 'vacherin')
```

---

**Question 3.**

In [131]:

```
print(fondue_ingredients)
```

```
('gruyere', 'vacherin')
```

---

**Question 4.** It is not possible to change "gruyere" to "cheddar" becauase tuples are immutable

---

**Question 5.** There is no function to remove the last element of a tuple. Instead, you can convert a tuple to a list and then convert it back.

In [132]:

```
fondue_ingredients = list(fondue_ingredients)
fondue_ingredients = fondue_ingredients[:-1]
fondue_ingredients = tuple(fondue_ingredients)
```

In [133]:

```
print(fondue_ingredients, type(fondue_ingredients))
```

```
('gruyere',) <class 'tuple'>
```

# Exercise 7 - Sets

---

**Question 1.**

In [138]:

```
s1 = {1, 2, 5, 5, 8}
s2 = {1, 2, 4, 9, 2}
```

---

**Question 2.**

In [139]:

```
print(s1)
print(s2)
```

```
{8, 1, 2, 5}
{1, 2, 4, 9}
```

There are no duplicate entries

---

**Question 3.** No, elements of sets cannot be accessed with indices

---

**Question 4.**

In [140]:

```
4 in s1
```

Out[140]:

```
False
```

In [141]:

```
4 in s2
```

Out[141]:

```
True
```

---

**Question 5.** & is intersection, | is union, - is difference, and ^ is symmetric difference

In [68]:

```
s1 & s2
```

Out[68]:

```
{1, 2}
```

In [69]:

```
s1 | s2
```

Out[69]:

```
{1, 2, 4, 5, 8, 9}
```

In [70]:

```
s1 - s2
```

Out[70]:

```
{5, 8}
```

In [71]:

```
s1 ^ s2
```

Out[71]:

```
{4, 5, 8, 9}
```

---

**Question 6.**

In [142]:

```
s1.remove(1)
s1.add(6)
print(s1)
```

```
{2, 5, 6, 8}
```

# Exercise 8 - Dictionaries

**Question 1.**

In [143]:

```
ages = {'Xiaohan':21, 'Christian':20, 'Grace':20, 'Sajid':21}
```

**Question 2.** The `keys` method can be used to create a list with all of the keys in a dictionary

In [146]:

```
for l in ages.keys():
    print(l)
```

```
Xiaohan
Christian
Grace
Sajid
Nicola
```

**Question 3.** A key-value pair can be added as follows:

In [147]:

```
ages['Nicola'] = 19
print(ages)
```

```
{'Xiaohan': 21, 'Christian': 20, 'Grace': 20, 'Sajid': 21, 'Nicola': 19}
```

**Question 4.** The `pop` method can be used to remove a key-value pair

In [148]:

```
ages.pop('Grace')
print(ages)
```

```
{'Xiaohan': 21, 'Christian': 20, 'Sajid': 21, 'Nicola': 19}
```

---

**Question 5.** It is not possible to create duplicate entries in a dictionary

In [149]:

```
ages['Sajid'] = 22
print(ages)
```

```
{'Xiaohan': 21, 'Christian': 20, 'Sajid': 22, 'Nicola': 19}
```

---

**Question 6.** The `in` keyword can be used to check whether a key exists in a dictionary

In [150]:

```
print('Harry' in ages)
```

```
False
```

---

# Exercise 9 - Modelling using data structures

In [172]:

```
planets = {'Mercury': [4_878, 0.06, 1_407.6],
           'Venus': [12_100, 0.82, 5_832],
           'Earth': [12_756, 1.00, 23.934],
           'Mars': [6_794, 0.11, 24.623],
           'Jupiter': [142_800, 317.89, 9.842],
           'Saturn': [120_000, 95.17, 10.233],
           'Uranus': [52_400, 14.56, 16],
           'Pluto': [2_445, 0.002, 153.36]
          }
```

---

**Question 1.**

In [173]:

```
pi = 3.142

earth_rot = planets['Earth'][2]

for k, p in planets.items():
    if p[2] < earth_rot:
        print(k, p[2])
```

```
Jupiter 9.842
Saturn 10.233
Uranus 16
```

---

# Exercise 10 - More data structures

In [180]:

```


my_list = [1,4,7,8,9]

print(sorted(my_list))

print(sorted(my_list, reverse=True))
```

```
[1, 4, 7, 8, 9]
[9, 8, 7, 4, 1]
```

In [ ]:

```

```

In [ ]:

```

```