In [1]:

```
# Conditions for permutations
# All the element that are present in one string should be in the other string
# 1a.python - pythno -> a permutation
# 1b.python - phy ton -> a permutation
# 2.python - Python -> not a permutation
# 3.python - cython -> not a permutation
# 4.python - ppython -> not a permutation
# 5.python - pythop -> not a permutation
# 6.python - ton -> not a permutaiton

def is_permutation(oranges, apples):
    oranges = oranges.replace(' ','')
    apples = apples.replace(' ','')
    print(list(oranges),list(apples))
    if len(list(oranges)) == len(list(apples)):
        test_apples = apples
        test_oranges = oranges
        for i in oranges:
            if i in test_apples:
                test_apples = test_apples.replace(i,'')
        for i in apples:
            if i in test_oranges:
                test_oranges = test_oranges.replace(i,'')
        if len(test_apples) == len(test_oranges) == 0:
            return 'A permutation'
        else:
            return 'Not a permutation'
    else:
        return 'Not a permutation by condition 6 and 4'
```

In [2]:

```
s1 = 'python'
s2 = 'phy ton'
is_permutation(s1,s2)
```

```
['p', 'y', 't', 'h', 'o', 'n'] ['p', 'h', 'y', 't', 'o', 'n']
```

Out[2]:

```
'A permutation'
```

In [3]:

```
s3 = 'be ar'
s4 = 'bare0'
is_permutation(s3,s4)
```

```
['b', 'e', 'a', 'r'] ['b', 'a', 'r', 'e', '0']
```

Out[3]:

```
'Not a permutation by condition 6 and 4'
```

In [4]:

```
s5 = 'bear'
s6 = 'bare'
is_permutation(s5,s6)
```

```
['b', 'e', 'a', 'r'] ['b', 'a', 'r', 'e']
```

Out[4]:

```
'A permutation'
```