In [1]:
```
# Python Exam - Begum Zubeda
# 1. Given below are a list of positive and negative words. Also a list of comments is 

positive = ['good','awesome', 'best', 'nice']
negative = ['worst','awful']
comments = ['He is a good boy', 'Food is the worst here', 'He is an awesome player', 'S

posneg = { "positive": [], "negative": [] }

for comment in comments:
    for word in comment.split():
        if word in positive:
            posneg["positive"].append(comment)
        elif word in negative:
            posneg["negative"].append(comment)

print(posneg)
```

```
{'positive': ['He is a good boy', 'He is an awesome player', 'She is the best', 'These b
urger are really nice'], 'negative': ['Food is the worst here', 'This pizza tastes awfu
l']}
```

In [4]:
```
# 2. Create a dictionary containing three lambda functions square, cube and square root

import math

dict1 = {'Square': lambda x: x**2, 'Cube': lambda x: x**3, 'Squareroot': lambda x: math

num = int(input("Enter a number: "))

sq = dict1['Square'](num)
cb = dict1['Cube'](num)
sr = dict1['Squareroot'](num)

addition = sq + cb + sr

print("Addition of square {}, cube {} and square root {} of num {} is {}".format(sq, cb
```

```
Enter a number: 16
Addition of square 256, cube 4096 and square root 4.0 of num 16 is 4356.0
```

In [5]:
```
# 3. Find the fruits that are sour in taste from the tuple given below.

fruits = (('Lemon','sour'), ('DragonFruit', 'Sweet'), ('Grapes','soUr'), ('Kiwi','Sour'
sourlist = []

for f, v in fruits:
    if v.lower() == "sour":
        sourlist.append(f)

print("Sour fruits list: ", sourlist)
```

```
Sour fruits list:  ['Lemon', 'Grapes', 'Kiwi', 'Orange', 'Limes']
```

In [7]:
```
# 4. A list of words is given. Find the words from the list that have their second char

ls = ['hello', 'Dear', 'hOw', 'ARe', 'You']

newls = [ x for x in ls if x[1].isupper() == True ]

print("Words with second character in uppercase: ", newls)
```

```
Words with second character in uppercase:  ['hOw', 'ARe']
```

In [18]:
```python
# 5. A dictionary of names and their weights on earth is given. Find how much they will
# Formula : wMoon = (wEarth * GMoon) / GEarth

# Weight of people in kg
WeightOnEarth = {'John':45, 'Shelly':65, 'Marry':35}

# Gravitational force on the Moon: 1.622 m/s2
GMoon = 1.622

# Gravitational force on the Earth: 9.81 m/s2 GEarth = 9.81
GEarth = 9.81

MWeight = dict(map(lambda x: (x, (WeightOnEarth[x] * GMoon) / GEarth), WeightOnEarth))

print("Weight on Moon: ", MWeight)
```

```
Weight on Moon:  {'John': 7.440366972477065, 'Shelly': 10.747196738022426, 'Marry': 5.78
6952089704383}
```

In [19]:
```python
# 6. Write a program to fetch the words from the given list which have their first char

namesList = ['santa Maria', 'Hello World','Merry christmas', 'tHank You']

newls = [ x for x in ls if x[0].isupper() == True ]

print("Words with 1st character in uppercase: ", newls)
```

```
Words with 1st character in uppercase:  ['Dear', 'ARe', 'You']
```

In [26]:
```python
# 7. A list containing multiple lists is given. Convert each inner list into sets and f
from functools import reduce

given_sets = [[1, 2, 3, 4, 8], [2, 3, 8, 5, 6], [8, 4, 5, 3, 7], [6, 9, 8, 3], [9, 12,

intersection = set(reduce( lambda x, y: set(x).intersection(set(y)), given_sets ))

print(intersection)
```

```
{8, 3}
```

In [25]:
```python
# 8. Find the cumulative average of the list [9,8,7,6,5] using accumulate() and lambda
from itertools import accumulate
import numpy as np

lst = [9, 5, 7, 8, 5]

cum = list(accumulate([2,4,6,3,1], lambda x, y: x + y)) / np.arange(1, len(lst) + 1)

print("Cumulative Average: ", cum)
```

```
Cumulative Average:  [2.   3.   4.   3.75 3.2 ]
```

In [27]:
```python
# 9. A list of words is given. Convert the words into uppercase. Use lambda and map fun

lsbool = ['True','FALse','tRUe','tRue','False','faLse']

upperbool = list(map( lambda x: x.upper(), lsbool ))

print("Uppercase list: ", upperbool)
```

```
Uppercase list:  ['TRUE', 'FALSE', 'TRUE', 'TRUE', 'FALSE', 'FALSE']
```

In [28]:
```
# 10. A list of dates (dd-mm-yyyy) in the form of string is given below. Create a new l

datesList = ['17-12-1997','22-04-2011','01-05-1993','19-06-2020']

yearsList = [ x[-4: ] for x in datesList ]

print("Years list: ", yearsList)
```

```
Years list:  ['1997', '2011', '1993', '2020']
```

In [ ]: