Assignment 2

Task 1

1.1 Write a Python Program to implement your own myreduce() function which works exactly like Python's built-in function reduce()

In [9]:
```
# Custom reduce function
def myreduce(customfunc, seq):
    result = seq[0]
    for item in seq[1:]:
        result = customfunc(result, item)

    return result

# Function to find max item between a and b
def find_max_item(a, b):
    if a > b:
        return a
    else:
        return b


if __name__ == '__main__':
    ls_data = [2, 30, 75, 45, 9]
    print('Maximum number from list {} is {}'.format(ls_data, myreduce(find_ma
x_item, ls_data)))
```

```
Maximum number from list [2, 30, 75, 45, 9] is 75
```

1.2 Write a Python program to implement your own myfilter() function which works exactly like Python's built-in function filter()

In [10]:
```
# Custom filter function
def my_filter(customfunc, seq, ch):
    ls_result = []
    for item in seq:
        if customfunc(item, ch):
            ls_result.append(item)

    return ls_result


# Function to filter and return list as per character 'ch'
def filter_by_char_j(ls_data, ch):
    ls_temp = []
    for data in ls_data:
        if data[0].lower() == ch:
            ls_temp.append(data)

    return ls_temp


if __name__=='__main__':
    ch = 'j'
    ls_data = 'Jack and jill went up the hill'.split(' ')
    print('List of items filter by {}: {}'.format(ch, my_filter(filter_by_char
_j, ls_data, ch)))
```

```
List of items filter by j: ['Jack', 'jill']
```

2. Implement List comprehensions to produce the following lists.

Write List comprehensions to produce the following Lists

['A', 'C', 'A', 'D', 'G', 'I', ’L’, ‘ D’]

['x', 'xx', 'xxx', 'xxxx', 'y', 'yy', 'yyy', 'yyyy', 'z', 'zz', 'zzz', 'zzzz']

['x', 'y', 'z', 'xx', 'yy', 'zz', 'xx', 'yy', 'zz', 'xxxx', 'yyyy', 'zzzz']

[[2], [3], [4], [3], [4], [5], [4], [5], [6]]

[[2, 3, 4, 5], [3, 4, 5, 6], [4, 5, 6, 7], [5, 6, 7, 8]]

[(1, 1), (2, 1), (3, 1), (1, 2), (2, 2), (3, 2), (1, 3), (2, 3), (3, 3)]

In [11]:
```
if __name__ == '__main__':
    # 1
    ls_temp = []
    ls_temp = [i for i in 'ACADGILD']
    print(ls_temp)

    # 2
    ls_temp.clear()
    ls_temp = [i*j for i in 'xyz' for j in range(1,5)]
    print(ls_temp)

    # 3
    ls_temp.clear()
    ls_temp = [i*j for j in range(1,5) for i in 'xyz']
    print(ls_temp)

    # 4
    ls_temp.clear()
    ls_temp = [[i+j] for j in range(0, 3) for i in range(2, 5)]
    print(ls_temp)

    # 5
    ls_temp.clear()
    ls_temp = [[i+j for i in range(1,5)] for j in range(1,5)]
    print(ls_temp)

    # 6
    ls_temp.clear()
    ls_temp = [(j,i) for i in range(1,4) for j in range(1,4)]
    print(ls_temp)
```

```
['A', 'C', 'A', 'D', 'G', 'I', 'L', 'D']
['x', 'xx', 'xxx', 'xxxx', 'y', 'yy', 'yyy', 'yyyy', 'z', 'zz', 'zzz', 'zzz
z']
['x', 'y', 'z', 'xx', 'yy', 'zz', 'xxx', 'yyy', 'zzz', 'xxxx', 'yyyy', 'zzz
z']
[[2], [3], [4], [3], [4], [5], [4], [5], [6]]
[[2, 3, 4, 5], [3, 4, 5, 6], [4, 5, 6, 7], [5, 6, 7, 8]]
[(1, 1), (2, 1), (3, 1), (1, 2), (2, 2), (3, 2), (1, 3), (2, 3), (3, 3)]
```

3.Implement a function longestWord() that takes a list of words and returns the longest one

In [12]:
```
def longest_word(ls_words):
    output = ''
    temp = 0
    for word in ls_words:
        if len(word) > temp:
            temp = len(word)
            output = word

    return output

if __name__ == '__main__':
    ls_words = 'This is an Assignment 2 Task 2 Program 3'.split(' ')
    lg_word = longest_word(ls_words)
    print('Longest word from list: "{}"'.format(lg_word))
```

```
Longest word from list: "Assignment"
```

Task 2

1.1 Write a Python Program(with class concepts) to find the area of the triangle using the below formula.

area = (s*(s-a)*(s-b)*(s-c)) ** 0.5

Function to take the length of the sides of triangle from user should be defined in the parent class and function to calculate the area should be defined in subclass.

In [13]:
```
class Triangle(object):
    def __init__(self):
        print('Ctor of Parent Class - Triangle')


    def set_sides(self, a, b, c):
        if (a+b > c) and (b+c > a) and (a+c > b):
            print('Setting values of a, b, c')
            self.a = a
            self.b = b
            self.c = c
            return True
        else:
            print('ERROR - Value of a, b, c is not set!!!')
            return False


class SubTriangle(Triangle):
    def __init__(self):
        Triangle.__init__(self)
        print('Ctor of Subclass - SubTriangle')


    def get_area(self):
        s = (self.a + self.b + self.c) / 2
        area = (s * (s - self.a) * (s - self.b) * (s - self.c)) ** 0.5
        return area


if __name__ == '__main__':
    print('Calculating Area of Triangle having side 14, 18, 16')
    obj_triangle = SubTriangle()
    if obj_triangle.set_sides(14,18,16):
        print()
        print('Area of Triangle(14,18,16): {}'.format(obj_triangle.get_area
()))
```

```
Calculating Area of Triangle having side 14, 18, 16
Ctor of Parent Class - Triangle
Ctor of Subclass - SubTriangle
Setting values of a, b, c

Area of Triangle(14,18,16): 107.33126291998991
```

1.2 Write a function filter_long_words() that takes a list of words and an integer n and returns the list of words that are longer than n.

In [14]:
```
def filter_long_words(ls_words, n):
    ls_temp = []
    for word in ls_words:
        if len(word) > n:
            ls_temp.append(word)
    return ls_temp

def filter_long_words_using_filter(ls_words, n):
    ls_temp = filter(lambda x: len(x) > n, ls_words)
    return [word for word in ls_temp]

if __name__ == '__main__':
    n=5
    ls_words = 'This is an Assignment 2 Task 2 Program 3'.split(' ')
    print('List of words:                {}'.format(ls_words))

    # Using filter()
    print()
    data = filter_long_words_using_filter(ls_words, n)
    print('List of words greater than {} (using filter()): {}'.format(n, data
))

    # Using for-loop
    print()
    print('List of words greater than {}: {}'.format(n, filter_long_words(ls_w
ords, n)))
```

```
List of words:                ['This', 'is', 'an', 'Assignment', '2', 'Task',
'2', 'Program', '3']

List of words greater than 5 (using filter()): ['Assignment', 'Program']

List of words greater than 5: ['Assignment', 'Program']
```

2.1 Write a Python program using function concept that maps list of words into a list of integers representing the lengths of the corresponding words.

In [15]:
```
def words_length(ls_words):
    ls_temp = []
    for word in ls_words:
        ls_temp.append(len(word))

    return ls_temp

def words_length_using_map(ls_words):
    return map(len, ls_words)

if __name__ == '__main__':
    ls_words = 'This is an Assignment 2 Task 2 Program 3'.split(' ')
    print('List of words:            {}'.format(ls_words))

    # Using map()
    print()
    data = words_length_using_map(ls_words)
    print('Words length using map(): {}'.format([d for d in data]))

    # Using for-loop
    print()
    print('List of integers(count):  {}'.format(words_length(ls_words)))
```

```
List of words:            ['This', 'is', 'an', 'Assignment', '2', 'Task',
'2', 'Program', '3']

Words length using map(): [4, 2, 2, 10, 1, 4, 1, 7, 1]

List of integers(count):  [4, 2, 2, 10, 1, 4, 1, 7, 1]
```

2.2 Write a Python function which takes a character (i.e. a string of length 1) and returns True if it is a vowel, False otherwise.

In [16]:
```
def is_vowel(ch):
    if ch.lower() in ['a','e','i','o','u']:
        return True
    return False

if __name__ == '__main__':
    ls_test_chars = ['r','h','a','k','i','g', 'o']
    for each_char in ls_test_chars:
        print(is_vowel(each_char))
```

```
False
False
True
False
True
False
True
```