Generating Lists

```python
#-------------------------------------------------------------------------------

def Positives( numbers ) :

    # The list of positive items in the numeric list 'numbers'

    positives = [ ]

    for n in numbers :
        if n > 0 :
            positives += [ n ]

    return positives
```

```
>>> Positives( [ 7, -2, -4, 3, 5, -1, 6, -8 ] )
[7, 3, 5, 6]

>>> Positives( [ 1, 2, 3, 4, 5 ] )
[1, 2, 3, 4, 5]

>>> Positives( [ ] )
[]

>>> Positives( [ -1, -2, -3 ] )
[]
```

```python
#-------------------------------------------------------------------------------

def Reverse( lst ) :

    # A copy of the list 'lst' with items in reverse order

    reverse = [ ]

    for item in lst :
        reverse = [ item ] + reverse

    return reverse
```

```
>>> Reverse( [ 1, 2, 3, 4, 5 ] )
[5, 4, 3, 2, 1]

>>> Reverse( [ 7 ] )
[7]

>>> Reverse( [ ] )
[]

>>> Reverse( [ "yoda", "is", "indeed", "wise" ] )
['wise', 'indeed', 'is', 'yoda']
```

```python
#-------------------------------------------------------------------------------

def Peaks( lst ) :

    # The list of items in list 'lst' which exceed all previous items,
    # ordered by their appearance in 'lst'

    peaks = [ ]

    for item in lst :
        if peaks == [ ] or item > maximum :
            peaks   += [ item ]
            maximum  = item

    return peaks
```

```
>>> Peaks( [ 3, 2, 5, 1, 4, 7, 6, 9, 8 ] )
[3, 5, 7, 9]

>>> Peaks( [ 1, 2, 3, 4, 5 ] )
[1, 2, 3, 4, 5]

>>> Peaks( [ 5, 4, 3, 2, 1 ] )
[5]

>>> Peaks( [ ] )
[]

>>> Peaks( [ "fairy", "tale", "of", "new", "york" ] )
['fairy', 'tale', 'york']
```

```python
#-------------------------------------------------------------------------------

def Replicate( count, lst ) :

    # A copy of the list 'lst' with each item replicated 'count' times

    replicate = [ ]

    for item in lst :
        for _ in range( count ) :
            replicate += [ item ]

    return replicate

#-------------------------------------------------------------------------------
```

```
>>> Replicate( 3, [ "uggy", "oi" ] )
['uggy', 'uggy', 'uggy', 'oi', 'oi', 'oi']

>>> Replicate( 3, [ 5, 2, 8, 4 ] )
[5, 5, 5, 2, 2, 2, 8, 8, 8, 4, 4, 4]

>>> Replicate( 1, [ 5, 2, 8, 4 ] )
[5, 2, 8, 4]

>>> Replicate( 0, [ 5, 2, 8, 4 ] )
[]
```