Dr. Gunjan Mansingh
Department of Computing
UWI

July 4- 29, 2022

Week 2 - Day 3, Session 1

Common Patterns of List Computations

We will now look at common operations on lists

Recursive structure of list makes it easy to write recursive functions to process them.

What should we do

- if list is empty
- if list is not empty,
  - What to do to the head of the list (i.e. first element)
  - What to do with the rest of the list (i.e. all of the list except the head)

Traversing a List Recursively

sumList([1,2,3,4])
10

```
def sumList(lst):
    if lst == [ ]:
        return 0
    else:
        return lst[0] + sumList(lst[1:])
```

```
sumList([1,2,3,4])
  1+ sumList([2,3,4])
    2+ sumList([3,4])
      3+ sumList([4])
        4+ sumList([])
          0
```

Traversing a List

Write a function called myLen that takes a list, and returns the number of elements in the list.

```
myLen ([ 2, 3, 5, 7, 11, 13])
6
```

```
def myLen(lst):
    if lst == []:
        return 0
    else:
        return 1 + myLen(lst[1:])
```

Traversing a List

Write a function called evenList that takes a list, and returns a list of even numbers. For simplicity assume that the list contains integers.

evenList([2, 3, 6, 4,1]) → [2,6,4]

```
def evenList(lst):
    if lst == [ ]:
        return [ ]
    elif lst[0]%2==0:
        return [lst[0]]+evenList(lst[1:])
    else:
        return evenList(lst[1:])
```

Traversing a List

Write a function called maxList that takes a list, and returns the maximum value in the list. For simplicity assume that the list contains integers.

maxList [2, 3, 6, 4,1] → 6

```
def maxList(lst):
    if lst==[]:
        return 0
    else:
        return max(lst[0], maxList(lst[1:]))
```

Traversing a List

Write a function called revList that takes a list, and returns a reversed list. For simplicity assume that the list contains integers.

revList ([2, 3, 6, 4,1]) → [1,4,6,3,2]

```
def revList(lst):
    if lst==[]:
        return []
    else:
        return revList(lst[1:]) + [lst[0]]
```

Traversing a String

Write a function called revStr that takes a string, and returns it reversed.

revStr (“hello”)→ “olleh”

```
def revStr(slst):
    if slst== "":
        return ""
    else:
        return revStr(slst[1:]) + slst[0]
```