HW3: Data Structure in Mathematics

Saen-Anan Bunyasiwa (SCIM 6105626)

January 27, 2020

R - 4.2

Draw the recursion trace for the computation of power(2,5), using the traditional function implemented in Code Fragment 4.11.

Code

```python
def power(x, n):
    if n == 0:
        return 1
    else:
        return x * power(x, n-1)

print(power(3,6))
```

Output:

```
729
```

Trace

let x = 2 and n = 5:

power(2, 5) will return 16 * x = 16 * 2 = 32

power(2, 4) will return 8 * x = 8 * 2 = 16

power(2, 3) will return 4 * x = 4 * 2 = 8

power(2, 2) will return 2 * x = 2 * 2 = 4

power(2, 1) will return 2 * x = 2 * 1 = 2

power(2, 0) will return 1

R - 4.3

Draw the recursion trace for the computation of *power*(2,18), using the repeated squaring algorithm, as implemented in Code Fragment 4.12.

Code Fragment

```python
def power(x, n):
    if n==0:
        return 1
    else:
        partial = power(x, n//2)
        result = partial*partial
        if n%2 == 1:
            result *= x
        return result


print(power(2, 18))
```

Output:

```
262144
```

Recursion Trace

We let x = 2 and n = 18:

power (2, 18) will return 512 * 512 = 262144

power (2, 9) will return 2 * 16 * 16 = 512

power (2, 4) will return 4 * 4 = 16

power (2, 2) will return 2 * 2 = 4

power (2, 1) will return (1 * 1) * 2 = 2

power (2, 0) will return 1

R - 4.4

Draw the recursion trace for the execution of function *reverse*(S, 0, 5)(Code Fragment 4.10) on $S = [4, 3, 6, 2, 6]$

Code Fragment

```
def reverse(S, start, stop):
        if start < stop-1:
                S[start], S[stop-1] = S[stop-1], S[start]
                reverse(S, start+1, stop-1)

# From the question initial:
S = [4, 3, 6, 2, 6]
reverse(S, 0, 5)
print(S)
```

Output:

```
[6, 2, 6, 3, 4]
```

Recursion Trace

Calling the function reverse on the initial $S = [4, 3, 6, 2, 6]$

Firstly, we call *reverse*(S, 0, 5)

reverse(S, 0, 5) return None (S is now [6, 2, 6, 3, 4] at the return)

produces [6, 3, 6, 2, 4]

recurse *reverse*(S, 1, 4) return None (S is now [6, 2, 6, 3, 4] at the return)

produces [6, 2, 6, 3, 4]

recurse *reverse*(S, 2, 3) return None (S is [6, 2, 6, 3, 4])

Note: the recursion, the swap actually happen before the next recursion occurs.

C - 4.16

Write a short recursive Python function that takes a character strings and outputs its reverse. For example, the reverse of `‘pots & pans’` would be `‘snap & stop’` .

Code

```
def reversestring(X, index = 0):
        if index == len(X)-1:
                return [X[index]]
        else:
                ans = reversestring(X, index + 1)
                ans.append (X[index])
                if index == 0:
                        ans = ''.join(ans)
                return ans


reversestring('pots&pans')
```

Output:

```
'snap&stop'
```