CS2302 - Data Structures

Fall 2020

Analyzing Recursive Functions

The Master Method: The solution to a recurrence equation of the form $T(n) = aT(n/b) + n^k$ is given by:

$$T(n) = \begin{cases} O(n^k) & \text{if } a < b^k \\ O(n^k \log n) & \text{if } a = b^k \\ O(n^{\log_b a}) & \text{if } a > b^k \end{cases}$$

For each of the following recursive methods, write a recurrence of the form $T(n) = a\,T(f(n)) + g(n)$ to describe its running time. Then solve your recurrences using the master method.

```
def r0(a):
    if len(a)>0:
        mid = len(a)//2
        print(a[mid],end='  ')
        r0(a[:mid])

def r1(a):
    if len(a)>0:
        mid = len(a)//2
        print(a[:mid],end='  ')
        r1(a[:mid])

def r2(a):
    if len(a)>0:
        mid = len(a)//2
        print(a[mid],end='  ')
        r2(a[:mid])
        r2(a[mid+1:])

def r3(a):
    if len(a)>0:
        for i in a:
            print(i)
        mid = len(a)//2
        r3(a[:mid])

def r4(a):
    if len(a)>0:
        for i in a:
            print(i)
        mid = len(a)//2
        r4(a[:mid])
        r4(a[mid+1:])

def r5(a):
    if len(a)>0:
        for i in a:
            for j in a:
                print(i,j)
        mid = len(a)//2
        r5(a[:mid])
```

```
def r6(a):
    if len(a)>0:
        for i in a:
            for j in a:
                print(i,j)
        mid = len(a)//2
        r6(a[:mid])
        r6(a[mid+1:])

def r7(a):
    if len(a)>0:
        for i in a:
            for j in a:
                print(i,j)
        mid = len(a)//2
        r7(a[:mid])
        r7(a[mid+1:])
        r7(a[mid+1:])

def r8(a):
    if len(a)>0:
        for i in a:
            for j in a:
                print(i,j)
        mid = len(a)//2
        r8(a[:mid])
        r8(a[:mid])
        r8(a[mid+1:])
        r8(a[mid+1:])

def r9(a):
    if len(a)>0:
        for i in a:
            for j in a:
                print(i,j)
        mid = len(a)//2
        for k in range(8):
            r9(a[:mid])
```