COP3502 10/10/2023

Fall 2021

Consider the following problem: You are given a set of weights, $\{w_0, w_1, w_2, \ldots, w_{n-1}\}$ and a target weight T. The target weight is placed on one side of a balance scale. The problem is to determine if there exists a way to use some subset of the weights to add on either side of the balance so that the scale will perfectly balance or not. For example, if T = 12 and the set of weights was {6, 2, 19, 1}, then one possible solution would be to place the weights 6 and 1 on the same side of the balance as 12 and place the weight 19 on the other side.

Below is a function that solves this problem recursively, with a wrapper function to make the initial recursive call. In terms of n, the size of the input array of weights, **with proof,** determine the **worst case** run time of the wrapper function. **(Note: Since only the run time must be analyzed, it's not necessary to fully understand WHY the solution works. Rather, the analysis can be done just by looking at the structure of the code.)**

```
int makeBalance(int weights[], int n, int target) {
    return makeBalanceRec(weights, n, 0, target);
}

int makeBalanceRec(int weights[], int n, int k, int target) {
    if (k == n) return target == 0;     BASE CASE O(1)
    int left = makeBalanceRec(weights, n, k+1, target-
weights[k]);                                          T(n-1)
    if (left) return 1; O(1) false in worst case.
    int right = makeBalanceRec(weights, n, k+1,
target+weights[k]);                                   T(n-1)
    if (right) return 1; false worst case
    return makeBalanceRec(weights, n, k+1, target);   T(n-1)
}
```

Argue $O(3^n)$ by 3 choices for each weight

$$T(n) = 3T(n-1) + O(1)$$
$$= 3(3T(n-2)+1)+1$$
$$= 9T(n-2) + (1+3)$$
$$= 9(3T(n-\cancel{2}\,3)+1) + (1+3)$$
$$= 27\,T(n-\cancel{2}\,3) + (1+3+9)$$

After k steps

$$= 3^k T(n-k) + \sum_{i=0}^{k-1} 3^i$$

$T(0)=1$ Plug in $n=k$

$$= 3^n T(0) + \sum_{i=0}^{n-1} 3^i$$
$$= 3^n + \frac{3^n-1}{2} = O(3^n)$$

Spring 2021

What is the run-time of the function `hash_func` shown below, in terms of *n*, the length of its input string? Please provide sufficient proof of your answer. (9 out of the 10 points are awarded for the proof. 1 point is awarded for the answer.)

```
#include <stdio.h>
#include <string.h>
#define MOD 1072373
#define BASE 256

int hash_func(char* str);
int hash_func_rec(char* str, int k);

int hash_func(char* str) {
    return hash_func_rec(str, strlen(str));
}

int hash_func_rec(char* str, int k) {
    if (k == -1) return 0;
    int sum = 0;
    for (int i=k-1; i>=0; i--)
        sum = (BASE*sum + str[i])%MOD;
    return (sum + hash_func_rec(str, k-1))%MOD;
}
```

$O(n)$

$O(1)$

$O(1)$

$O(n) +$

$T(n-1)$

Total

$O(n^2)+O(n)$

$= \boxed{O(n^2)}$

$T(n) = \cancel{O(n)} + O(n) + T(n-1)$

$T(n) = T(n-1) + n$

$= T(n-2) + (n-1) + n$

$= T(n-3) + (n-2) + (n-1) + n$

After k steps $= T(n-k) + \sum_{i=n-k+1}^{n} i$

$T(1) = 1 \longrightarrow T(1) + \sum_{i=2}^{n} i = 1+2+3+\dots+n = \frac{n(n+1)}{2}$

$O(n^2)$

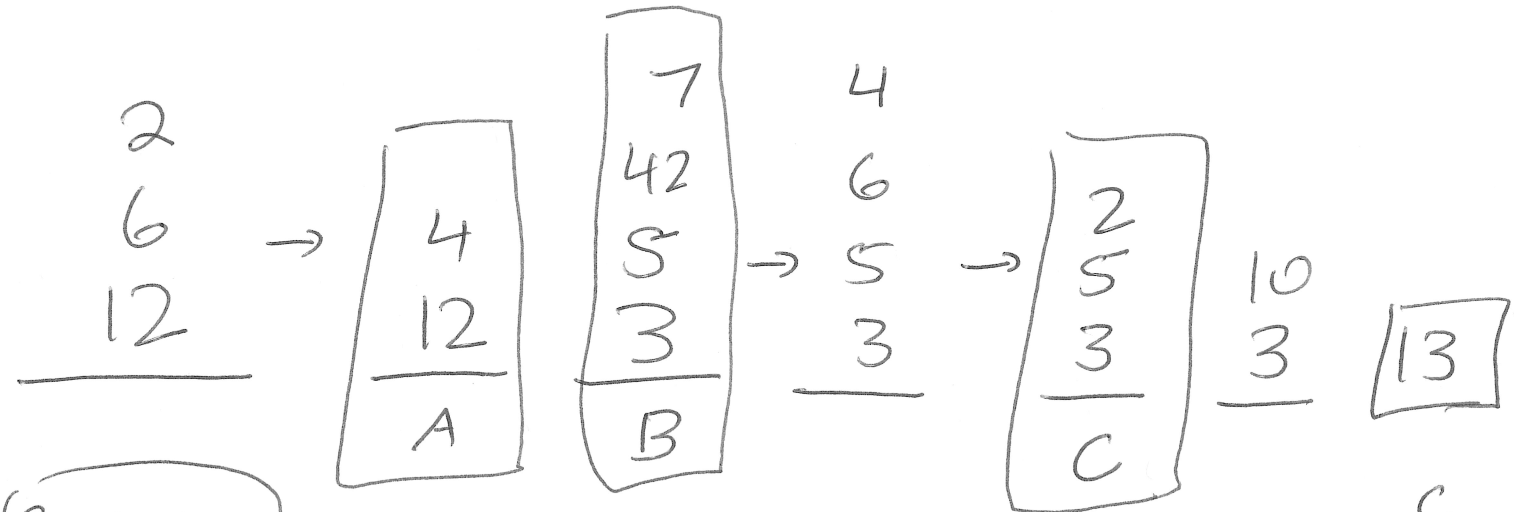
Spr 16 E2 Q1

char letter;
struct letnode* next;

```
int sinc(letnode* word) {

    if (word == NULL || word->next == NULL)
        return 1;

    while (word->next != NULL) {
        if ( word->letter >= word->next->letter)
            return 0;
        word = word->next;
    }
    return 1;
}

int stringlen(letnode* word) {
    if (word == NULL) return 0;
    return 1 + stringlen(word->next);
}

int stringcmpRec( letnode* first, letnode* second) {
    if (first == NULL && second == NULL) return 0;
    if (first == NULL) return -1;
    if (second == NULL) return 1;
  * if (first->letter != second->letter)
        return first->letter - second->letter;
    return stringcmpRec(first->next, second->next);
}
```

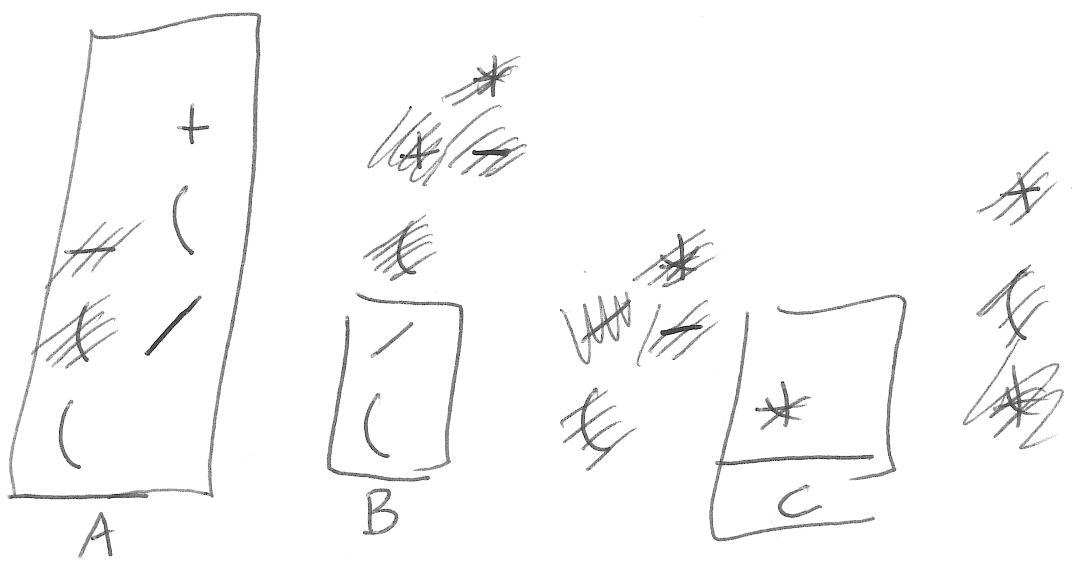
Q4 Spr 16

12 6 2 - / 5 42 7 / 4 - * +



Spr 16 Q6

((42 - 9) / (2 + 5 - 2 * 2) - 2 * 3) * (3 + 4)



42 9 - 2 5 + 2 2 * - / 2 3 * - 3 4 + *

Sum 2020
Final Exam

$$\sum_{i=n}^{3n-1}(5i+7) = \sum_{i=1}^{3n-1} 5i - \sum_{i=1}^{n-1} 5i + \sum_{i=n}^{3n-1} 7$$

$$= \frac{5(3n-1)3n}{2} - \frac{5(n-1)n}{2} + 7(3n-1-n+1)$$

$$= \frac{5n}{2}\Big(3(3n-1) - (n-1)\Big) + 7(2n)$$

$$= \frac{5n}{2}(9n-3-n+1) + 14n$$

$$= \frac{5n}{2}(8n-2) + 14n$$

$$= 5n(4n-1) + 14n$$

$$= 20n^2 - 5n + 14n$$

$$= \boxed{20n^2 + 9n}$$