Indian Institute of Technology Kharagpur

CS29003: Algorithms Laboratory, Spring 2022

Assignment 0

2PM – 5PM 11th January, 2022

Submit a single C/C++ source file. Do not use global or static variables.

Consider the problem of distributing n (identical) sweets amongst m children. Let the character 'S' represent a sweet. When $n = 7$, line up the sweets as "SSSSSSS". If there are $m = 3$ children, then we can represent a distribution of 7 sweets amongst 3 children as "SS|S|SSSS". Here '|' is a separator. The first child gets 2 sweets, second child gets 1 sweet and the third gets 4 sweets. Another possible distrtibution is "|SSSSSS|S" where the first child gets nothing, second child gets 6 sweets and the third gets 1. As suggested by the example, any distribution of n sweets amongst m children can be represented as a string of length $n + m - 1$ containing an arrangement of $m - 1$ separators and n many occurences of 'S' .

The input consists of 2 *positive* integers n and m. Your task is to generate all possible distributions (without repetitions) of n sweets amongst m children provided some constraints are satisfied and prints the total number of possible distributions. The output should be a list of strings (consisting of n 'S's and m '|'s), each printed in a separate line followed by the total count of strings printed. These strings represent all possible distributions under the specified constaints.

(a) Define a function *print_ a* that prints distributions in which each child gets atleast 1 sweet and returns the total number of such distributions. Use simple recursion. Fill the characters 'S','|' one by one in a string of size $n + m - 1$. When i 'S's and j '|'s are placed, think of what to place at $(i + j + 1)$-st position. For every choice, recursively call the function to enumerate the valid distributions with $i+j+1$ characters placed in the array.

(b) In this part, the constraint is that no two adjacent children can get 0 sweets. Write a function *print_ b* that prints distributions satisfying the aforementioned constraint and returns the total number of such combinations. Use simlar logic as part (a).

In the *main()* function, read n and m from the user, call *print_ a* and print the total number of distributions. Then call *print_ b* and print the total number of distributions it returns.

Example

```
n = 5
m = 4

(a)
SS|S|S|S
S|SS|S|S
S|S|SS|S
S|S|S|SS

Total number of distributions = 4

(b)
SSSS||S|
SSS|S|S|
SSS|S||S
SSS||SS|
SSS||S|S
SS|SS|S|
SS|SS||S
SS|S|SS|
SS|S|S|S
SS|S||SS
SS||SSS|
SS||SS|S
SS||S|SS
S|SSS|S|
S|SSS||S
S|SS|SS|
S|SS|S|S
S|SS||SS
S|S|SSS|
S|S|SS|S
S|S|S|SS
S|S||SSS
S||SSSS|
S||SSS|S
S||SS|SS
S||S|SSS
|SSSS|S|
|SSSS||S
|SSS|SS|
|SSS|S|S
|SSS||SS
|SS|SSS|
|SS|SS|S
|SS|S|SS
|SS||SSS
|S|SSSS|
|S|SSS|S
|S|SS|SS
|S|S|SSS
|S||SSSS

Total number of distributions = 40
```