For my cpp code :

First I check the num if it become 0 that means we could find a subset that sums to number in the array and returns true in this situation

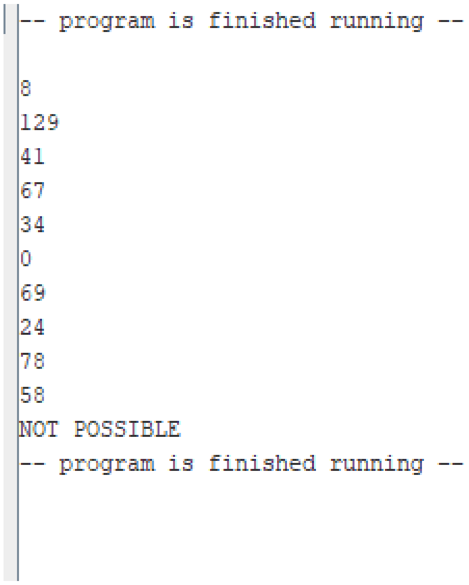
Then the next base case is checking the size of the array if it is zero that means we have iterated all over the array but didn’t find the subset and returns false in this situation

Then to call the function we do :

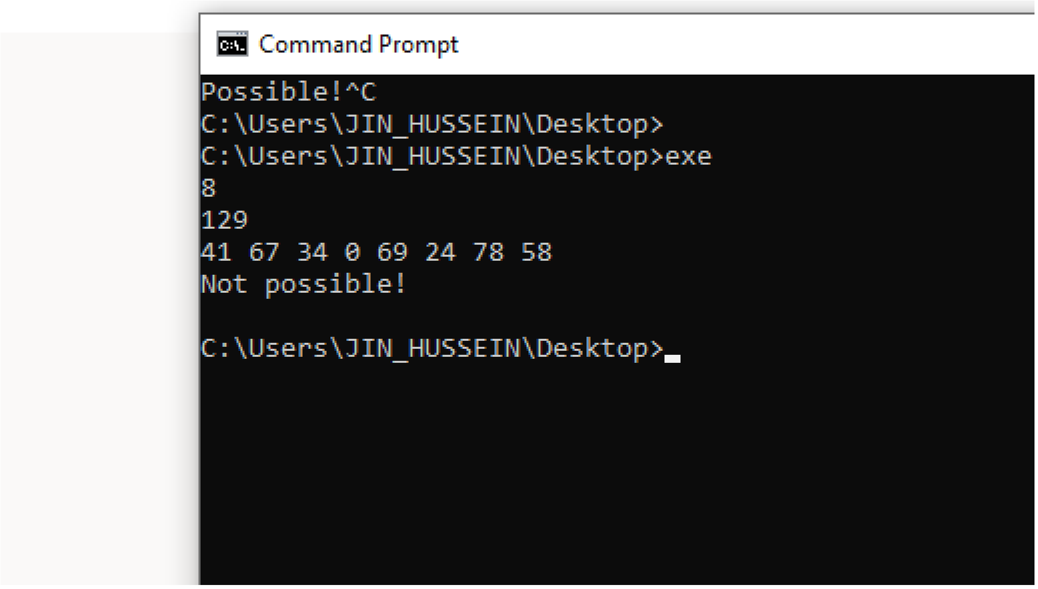
1. We include current item in the subset and recur for remaining items with remaining sum.
2. We exclude current item from subset and recur for remaining items.

For my assembly code there is a bit differences that is I could call the function only once that means only

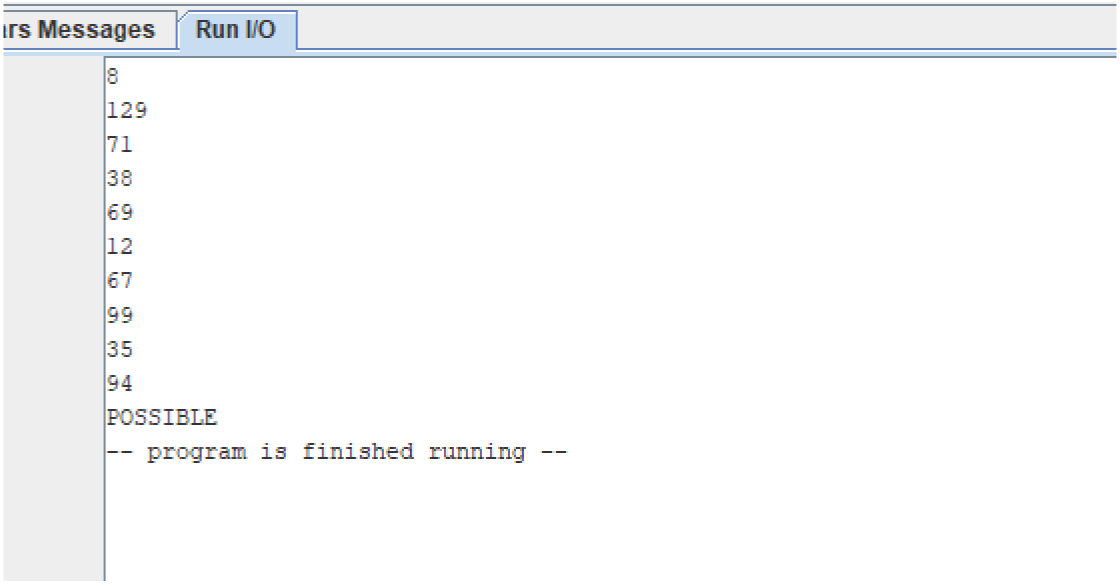
Either include current item in the subset and recur for remaining items with remaining sum.
Or will call the function excluding the current item from subset and recur for remaining items
(I didn’t have enough time to understand the usage of stacks and how to save the return value from a function)



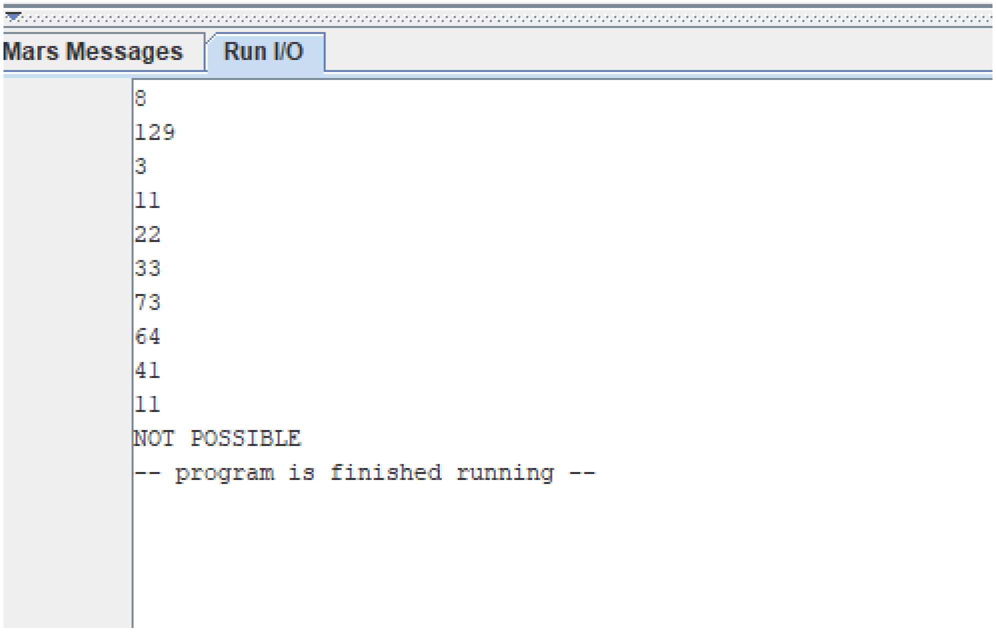
```
-- program is finished running --

8
129
41
67
34
0
69
24
78
58
NOT POSSIBLE
-- program is finished running --
```



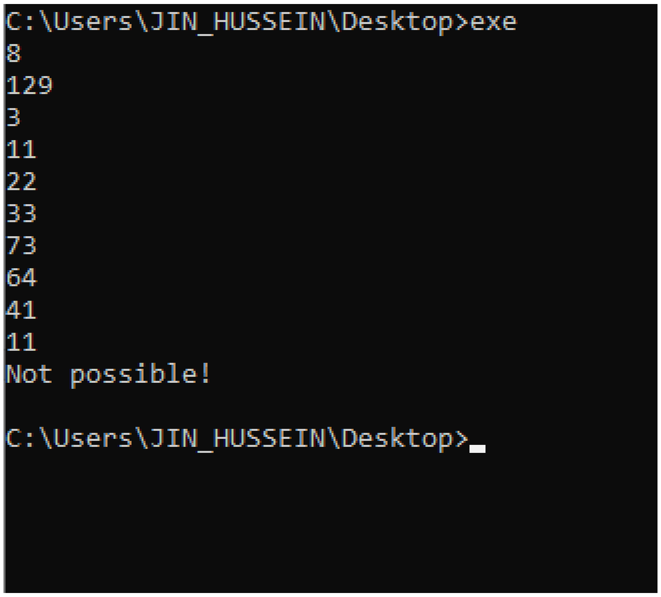
```
Possible!^C
C:\Users\JIN_HUSSEIN\Desktop>
C:\Users\JIN_HUSSEIN\Desktop>exe
8
129
41 67 34 0 69 24 78 58
Not possible!

C:\Users\JIN_HUSSEIN\Desktop>
```



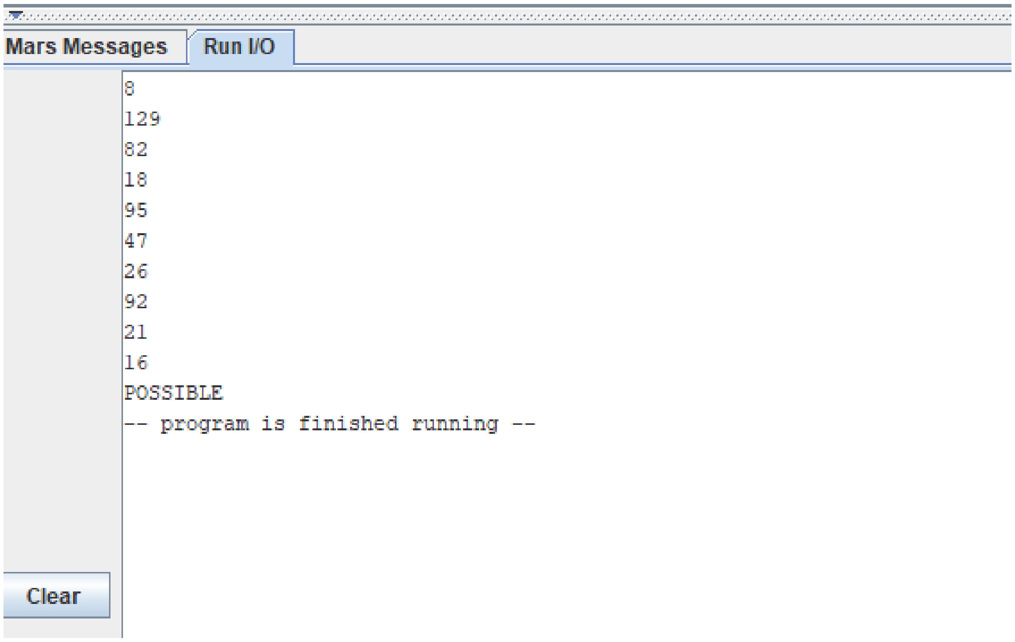
```
8
129
71
38
69
12
67
99
35
94
POSSIBLE
-- program is finished running --
```



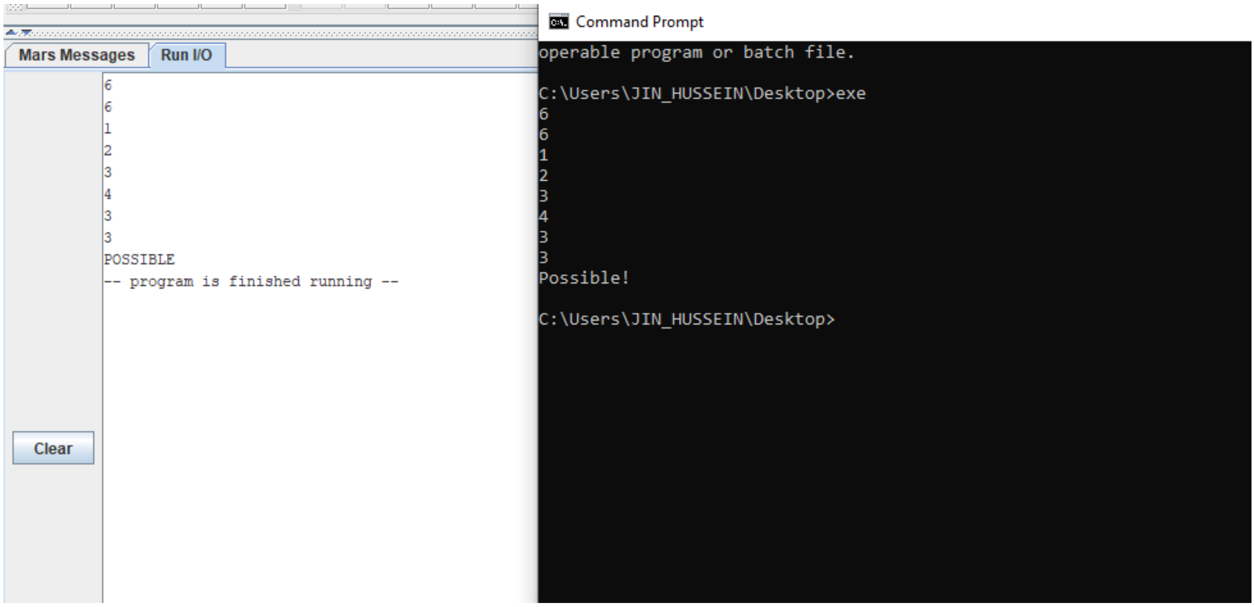
```
8
129
3
11
22
33
73
64
41
11
NOT POSSIBLE
-- program is finished running --
```



```
C:\Users\JIN_HUSSEIN\Desktop>exe
8
129
3
11
22
33
73
64
41
11
Not possible!

C:\Users\JIN_HUSSEIN\Desktop>
```



```
8
129
82
18
95
47
26
92
21
16
POSSIBLE
-- program is finished running --
```



```
6
6
1
2
3
4
3
3
POSSIBLE
-- program is finished running --
```

```
operable program or batch file.

C:\Users\JIN_HUSSEIN\Desktop>exe
6
6
1
2
3
4
3
3
Possible!

C:\Users\JIN_HUSSEIN\Desktop>
```