CSE 331

Computer Organization

HW2 Report

(Esra Eryılmaz 171044046)

1.Part

C++ code file contains recursive backtracking implementation as requested. I didn't write any helper function.

C++ tests:

(I sent makefile, if you want; you can use makefile while running the code.)

```
esra@esoo:~/Desktop$ make
g++ -c hw2.cpp
g++ -o hw hw2.o
esra@esoo:~/Desktop$ ./hw
Enter size : 8
Enter target num: 129
Enter array elements: 41 67 34 0 69 24 78 58
Not possible!
esra@esoo:~/Desktop$ ./hw
Enter size : 8
Enter target num: 129
Enter array elements: 62 64 5 45 81 27 61 91
Not possible!
esra@esoo:~/Desktop$ ./hw
Enter size : 8
Enter target num: 129
Enter array elements: 95 42 27 36 91 4 2 53
Possible!
esra@esoo:~/Desktop$ ./hw
Enter size : 8
Enter target num: 129
Enter array elements: 92 82 21 16 18 95 47 26
Possible!
esra@esoo:~/Desktop$ ./hw
Enter size : 8
Enter target num: 129
Enter array elements: 71 38 69 12 67 99 35 94
Possible!
esra@esoo:~/Desktop$ ./hw
Enter size : 8
Enter target num: 129
Enter array elements: 3 11 22 33 73 64 41 11
Not possible!
esra@esoo:~/Desktop$
```

2.Part

I've reserved space for arr, size and num in the .data section.

Firstly main function is running.

After taking size and target number from user ; main function calls forLoop function which takes array elements from user.

Then main function calls CheckSumPossibility which handle the recursive part of the program.

Everything is ok until here.

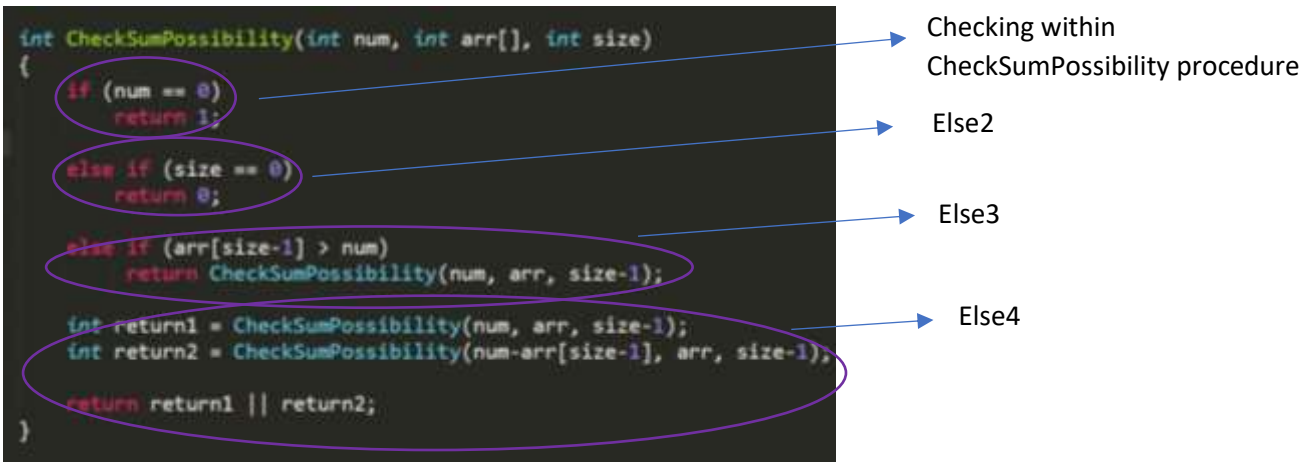
But inside the CheckSumPossibility function I think something goes wrong and I didn't figure it out. It doesn't give any error.

This function returns 1(possible) regardless of what elements are entered for array elements.

I thinks this is about $ra register.

Maybe I can't keep the old return values well when calling function recursively. I used a stack but couldn't figure it out.

other called procedure names:



I split the CheckSumPossibility function into several parts.

I write above, along with the names of the parts, which parts of the function they are related to.

(I did not attempt to make bonus parts.)

Assembly tests:

```
--------- WELCOME ---------
Enter the array size : 8
Enter the target number: 129
Enter array elements: 41
67
34
0
69
24
78
58

Possible!

-- program is finished running (dropped off bottom) --
```

1

```
--------- WELCOME ---------
Enter the array size : 8
Enter the target number: 129
Enter array elements: 62
64
5
45
81
27
61
91

Possible!
```

2

```
--------- WELCOME ---------
Enter the array size : 8
Enter the target number: 129
Enter array elements: 55
42
27
36
91
4
2
53

Possible!
```

3

```
--------- WELCOME ---------
Enter the array size : 8
Enter the target number: 129
Enter array elements: 92
82
21
16
18
95
47
26

Possible!
```

4

```
--------- WELCOME ---------
Enter the array size : 8
Enter the target number: 129
Enter array elements: 71
38
69
12
67
99
35
94

Possible!
```

5

```
--------- WELCOME ---------
Enter the array size : 8
Enter the target number: 129
Enter array elements: 3
11
22
33
73
64
41
11

Possible!
```

6