CSE 331/503
Computer Organization
Homework 2 Report

Berk PEKGÖZ
171044041

Function explanations

Main: Very similar to given function in assignment pdf. Some user interface adjustments used (e.g., Please enter size). And dynamic memory allocation added (In MIPS assembly too). Basically, takes integers from user and calls CheckSumPossibility (checksum in MIPS assembly) function. If that function returns 0, then prints “Not possible”. If returns 1, then prints the numbers which are the part of summation and prints “Possible”.

CheckSumPossibility / checksum: This is a recursive function. Takes 3 parameters.

- First parameter is “num” ($a0). “Num” is the number which is given as input by user. “Num” can change when another recursion called because to find the values function subtract that value from “num”.
- Second Parameter is “arr” ($a1). “Arr” is the array which keeps the integers given as input by user. Actually, arr never change because reaching integers by index.
- Third parameter is “size” ($a2). Size is size of the array which given as input by user (there is no max size. User can decide the size until MIPS assembly capacity). Function call other recursion with size-1.

In MIPS assembly part, procedure stores the values of S registers (used ones) and $ra to stack to do not lose the content of them.

Then, checks base cases. There are three base cases. For possible return (1) num must be 0.

Then checks if the current index bigger then sum or not. If it is bigger then goes to the next step directly (optimization).

Then there are two recursive calls if first one returns 1 then second one will not be called.

Finally returns returnValue.

Test cases

For each case first screenshot is C++ output and second screenshot is MARS output.

Case 1:

```
Please enter size of the array: 3
Plese enter the target number: 5
2
3
7
Result is: 2 3 are the elements gives summation. Possible!
```


```
Please enter size of the array: 3
Plese enter the target number: 5
2
3
7
Result is: 2 3 are the elements gives summation. Possible!
```


Case 2:

```
Please enter size of the array: 3
Plese enter the target number: 5
2
4
6
Result is: Not possible!
```


```
Please enter size of the array: 3
Plese enter the target number: 5
2
4
6
Result is: Not possible!
```


Case 3:

```
Please enter size of the array: 5
Plese enter the target number: 24
12
14
8
3
1
Result is: 12 8 3 1 are the elements gives summation. Possible!
```


```
Please enter size of the array: 5
Plese enter the target number: 24
12
14
8
3
1
Result is: 12 8 3 1 are the elements gives summation. Possible!
```


Case 4:

```
Please enter size of the array: 5
Plese enter the target number: 24
19
7
4
21
2
Result is: Not possible!
```

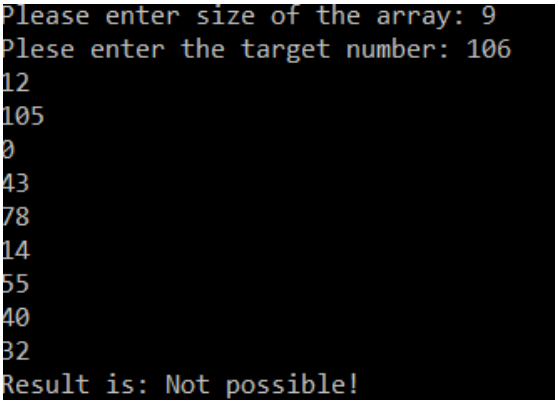

```
Please enter size of the array: 5
Plese enter the target number: 24
19
7
4
21
2
Result is: Not possible!
```

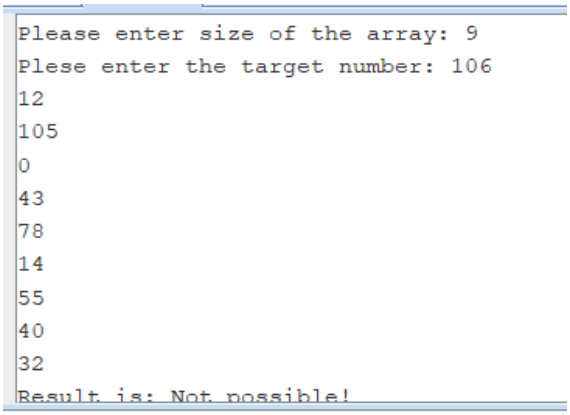

Case 5:

```
Please enter size of the array: 9
Plese enter the target number: 106
24
76
77
3
7
12
13
107
2
Result is: 76 3 12 13 2 are the elements gives summation. Possible!
```


```
Please enter size of the array: 9
Plese enter the target number: 106
24
76
77
3
7
12
13
107
2
Result is: 76 3 12 13 2 are the elements gives summation. Possible!
```


Case 6:





Bonus parts

Optimizations: There are two optimizations.

- First one, while going on the array if the number is bigger than sum, then directly skips that integer.
- Second one, while going on array if the sum is found then function starts to return (but important point is the searching starts from end of array).

Dynamic Memory: In both C++ and MIPS Assembly memory allocation is dynamic instead of constant array max size. In MIPS Assembly, allocates 4*size memory according to size which is given as an input by user.

Print: If the array has the integers which gives the summation then prints them to console (e.g., in test cases).