CSE 331/503

Computer Organization

Homework 2

Due Date 27/11/2020 Friday 17:00

In this assignment you will write a MIPS assembly program and test it using MARS instruction set simulator.

You will be given an integer array **arr** and a number **num**. The task is to find if a subset of array elements can sum up to the target **num**. If not possible you will output "Not possible!". If it is possible, output "Possible!". You can use every array element only once. Only positive integers are allowed as array elements. Finding only one combination is enough to output "Possible!".

1. (30pts) In C++, you will write a recursive function named **CheckSumPossibility** as below:

```
int CheckSumPossibility(int num, int arr[], int size)
```

The function respectively gets the target number, the array and the size of the array. It returns 1 if a subset of the array can sum up to the target number and otherwise if it is not possible it outputs a 0. The sample output for six sample runs with 8 element array examples is as shown below. The numbers in parenthesis (black text) are for showing the possibility. Blue text is user input, red text is program output. 8 is the array size, 129 is the target number:

```
8 129
41 67 34 0 69 24 78 58
Not possible!
8 129
62 64 5 45 81 27 61 91
Not possible!
8 129
95 42 27 36 91 4 2 53
Possible! (36 91 2)
8 129
92 82 21 16 18 95 47 26
Possible! (92 21 16)
8 129
71 38 69 12 67 99 35 94
Possible! (35 94)
8 129
3 11 22 33 73 64 41 11
Not possible!
```

The main function is as below:

```
int main()
{
    int arraySize;
    int arr[MAX_SIZE];
    int num;
    int returnVal;

    cin >> arraySize;
    cin >> num;

    for(int i = 0; i < arraySize; ++i)
    {
        cin >> arr[i];
    }

    returnVal = CheckSumPossibility(num, arr, arraySize);

    if(returnVal == 1)
    {
        cout << "Possible!" << endl;
    }
    else
    {
        cout << "Not possible!" << endl;
    }


    return 0;
}
```

2. (70pts) Write your program in MARS assembly with the guidance of the C++ code you wrote for part 1. You will read the input array from the console as shown in above **main()**.

RULES:

- You can use pseudo instructions in MARS.
- Comment your assembly using your C++ code. For instance:

```
#int returnVal = 0
add $s0, $zero, $zero

#if(size == 1)
addi $t0, $zero, 1
bne $a2, $t0, SizeNotOne
```

- Obey the contract, which defines the usage of MIPS registers.
- You have to write **CheckSumPossibility** as a procedure in assembly.
- Recursive backtracking implementation will get full credit. Any other accurately working implementation will get -15pts.
- For possible cases, if you also print the corresponding array elements in your assembly program (similar to shown in parenthesis in the above sample output) you get 10 extra points.
- For recursive backtracking you can check the lecture given by Marty Stepp from Stanford University: https://youtu.be/cR8YjEJrb-A
- Anyone using a compiler instead of writing the assembly manually will get 0pts.
- Do not cheat, do not help anyone either. Otherwise both sides will get 0pts.

Hint for C++ part, which also affects the assembly program:

1. Actually the recursive backtracking is an easier solution. An accurate implementation of the **CheckSumPossibility** function is even less than 20 lines of C/C++ code.
2. As an optimization,
    a. you must ignore the next recursive calls when the sum exceeded the target number **num**.
    b. your program should stop when only one possible combination is found.
3. You do not need any other helper function than the **CheckSumPossibility** function.