Name: Section: Rollno:

ESC101A: Fundamentals of Computing(Minor Quiz 3)

1st April, 2014

Total Number of Pages: 3 Total Points 23

Question	Points	Score
1	15	
2	8	
Total:	23	

Instructions

1. Read these instructions carefully.
2. Write you name, section and roll number on all the pages of the answer book.
3. Write the answers cleanly in the space provided. There is space left on the back of the answer book for rough work.
4. Do not exchange question books or change the seat after obtaining question paper.
5. Using pens (blue/black ink) and not pencils. Do not use red pens for answering.
6. Even if no answers are written, the answer book has to be returned back with name and roll number written.

Helpful hints

1. The questions are *not* arranged according to the increasing order of difficulty. Do a quick first round where you answer the easy ones and leave the difficult ones of the subsequent rounds.
2. For fill in the blanks type of questions, read the comments in the code. They usually have helpful remarks.

Name: Section: Rollno:

Question 1. (15 points) Consider the program given below.

```
#include <stdio.h>
#define N 2

int g=0, h=0;

int fun1(int n) {
  g++;
  printf("S: %d\n",n);
  if (n == 0) return 0;
  return (fun1(n-1) + 2*n - 1);
}

int fun2(int n) {
  h++;
  printf("T: %d\n",n);
  if (n == 0) return 0;
  return (fun2(n-1) + 3*fun1(n) - 3*n + 1);
}

int main(){
  int ans;
  ans = fun2(N);
  printf("%d %d %d\n", g, h, ans);
  return 0;
}
```

What is the output of the above program? Note that the addition operator (+) and the subtraction operator (-) have the same precedence and their order of associativity is from left to right.

Solution:

```
T: 2
T: 1
T: 0
S: 1
S: 0
S: 2
S: 1
S: 0
5 3 8
```

Name: Section: Rollno:

Question 2. (8 points) Given a set A having n elements and a number k, we want to compute how many subsets does A have that sum exactly to k. Here is a recursive definition for solving the problem. Consider an array $A[0,\ldots,n-1]$ with n elements and the number k. Let $\text{NS}(A[0,\ldots,n-1],k)$ be the numbers of subsets of the array A that sums to k. Then,

$$\begin{aligned}
\text{NS}(A[0,\ldots,n-1],k) &= \text{NS}(A[1,\ldots,n-1],k) + \text{NS}(A[1,\ldots,n-1],k-A[0]) \qquad \text{for } n>0, k>0\\
\text{NS}(A[0,\ldots,n-1],k) &= 0 \qquad \text{for } n == 0\\
\text{NS}(A[0,\ldots,n-1],k) &= 0 \qquad \text{for } k<0\\
\text{NS}(A[0,\ldots,n-1],0) &= 1
\end{aligned}$$

Assume that all elements of the set A are distinct and positive.

Below is a partially filled C function that takes as argument an array `ar`, a positive integer `n` (size of the array `ar`) and a number `k` and computes the number of subsets of `ar` that sum to `k`. Once again assume that all elements of `ar` are distinct and positive.

```
1 int num_subsets(int *ar, int n, int k){
2
3   if (_________________) return 1;
4
5   if (_________________ || _________________) return 0;
6
7   return (num_subsets(________,________,k) + num_subsets(________,________,
       ________));
8 }
```

Solution:

```
1 int num_subsets(int *ar, int n, int k){
2
3   if (k==0) return 1;
4
5   if (k<0 || n==0) return 0;
6
7   return (num_subsets(ar+1,n-1,k) + num_subsets(ar+1,n-1,k-*ar));
8 }
```