National Institute of Technology, Calicut
Department of Computer Science and Engineering
CS2094 – Data Structures Lab

Assignment 3 (Advanced)

Submission deadline (on or before):
22nd February 2015, 10:00:00 PM (for Advanced batch)

Naming Conventions for submission
Submit a single ZIP (.zip) file (do not submit in any other archived formats like .rar or .tar.gz). The name of this file must be ASSG<Number>A_<ROLLNO>_<FIRSTNAME>.zip (For example: ASSG3A_BxxyyyyCS_LAXMAN.zip). DO NOT add any other files (like temporary files, input files, etc.) except your source code, into the zip archive.
The source codes must be named as ASSG<Number>A_<ROLLNO>_<FIRST NAME>_<PROGRAM-NUMBER>.<extension> (For example: ASSG3A_BxxyyyyCS_ LAXMAN_1.c).

Questions

1. Write an in-place recursive function **reverse()** to reverse a singly linked list containing integers in the range -2^{30} to $+2^{30}$, in O(n) time. Also implement a **main()** function which creates the singly linked list by reading the values from the console, then calls **reverse()** on this singly linked list and finally prints the reversed list.
Note: Write a separate **print** function that takes the head of linked list as input and prints its contents. You must use this function to print the contents of the reversed list.

Input format :
The first line contains an integer *n*, length of the singly linked list.
The next line contains *n* space separated integers, representing the linked list.

Output format :
n space separated integers, representing the reversed linked list.

Example :
Input :

```
6
12 34 17 24 91 42
```

Output :

```
42 91 24 17 34 12
```

2. An n-digit (where, 1 <= n <= 10000) positive integer is represented by a singly linked list (SLL) of length *n* in the following way: Each digit of the number is stored in a node of SLL so that the most significant digit is at the node pointed to by the head pointer of the linked list and the least significant digit is at the tail of SLL. For example, the number 673924 is represented as 6 -> 7 -> 3 -> 9 -> 2 -> 4.

Write a recursive function **sum_of_digits()** that takes as input two SLLs representing two n-digit numbers as arguments and modifies the first SLL to contain the sum of both the input SLLs. The function should only use O(1) additional space and should run in O(n) time. Also implement **main()** function that takes two n-digit positive integers, where 1<=n<=10000, stores them as SLLs in the aforementioned manner, then calls the function **sum_of_digits()** and finally prints the contents of the modified SLL, representing the sum of the input numbers.

Note: You should not reverse either of the given lists or modify them in any way other than those specified above. The **sum_of_digits()** function must also appropriately handle the case where there is a carry from the most significant bits of the input numbers.

Input format :
The first line contains an integer *n*, with 1<=n<=10000.
Each of the next two lines contains 2 *n-digit* positive integers.

Output format :
An *n* or *n+1* digit number, containing the sum of the input numbers.

Example :
Input :

```
6
654321
532187
```

Output :

```
1186508
```

****