**INSTRUCTIONS:**

1. **You have to cover all possible corner cases, else you will only get partial score**
2. **Abide by the time and space complexity else the solution won’t be considered. Zero marks will be awarded.**
3. **Any instance of cheating/plagiarism will disqualify you from further workshop.**
4. **Questions are clear , so you are not allowed to ask doubts from the invigilator.**

**SECTION 1**

Q. Given a sorted array and a number x, find the pair in array whose sum is closest to x. You have to print 3 values as your output: index1, index2 and the sum i.e. a[index1] + a[index2]. (25 Marks)

Your input will be ‘n’ denoting the size of array followed by the contents of the array and the value of x.

For e.g.

Input:

6

10 22 28 29 30 40

55

Output:

1 4 52

Expected Time Complexity: O(n)

Expected Space Complexity: O(1)

Q. A "0/1 string" is a string in which every character is either 0 or 1. There are two operations that can be performed on a 0/1 string: **Asked by Google in 2015** (60 Marks)

* **switch**: Every 0 becomes 1 and every 1 becomes 0. For example, "100" becomes "011".
* **reverse**: The string is reversed. For example, "100" becomes "001".

Consider this infinite sequence of 0/1 strings:  
  
S0 = ""  
  
S1 = "0"  
  
S2 = "001"  
  
S3 = "0010011"  
  
S4 = "001001100011011"  
  
...  
  
SN = SN-1 + "0" + **switch**(**reverse**(SN-1)).

You need to figure out the Kth character of Sn, where ‘k’ and ‘n’ are user inputs.

Your input will be two integers ‘k’ and ‘n’. Our indexing starts from 0, hence k=0 implies 1st character.

For e.g.

Input

5 4

Output

1

Expected Time Complexity: O(Log(length\_of\_String))

Expected Space Complexity: O(length\_of\_String)

Q.3. You are given 2 binary strings i.e. consisting of 0/1. You have to print the string obtained on summing them up. (summing doesn’t mean string concatenation rather it means arithmetic sum).

**Asked by Facebook** (15 Marks)

Input

S1=”110”

S2=”11”

Output:

“1001”

Expected Time Complexity: O(MAX(length\_of\_S1, length\_of\_S2))

Expected Space Complexity: O(1)

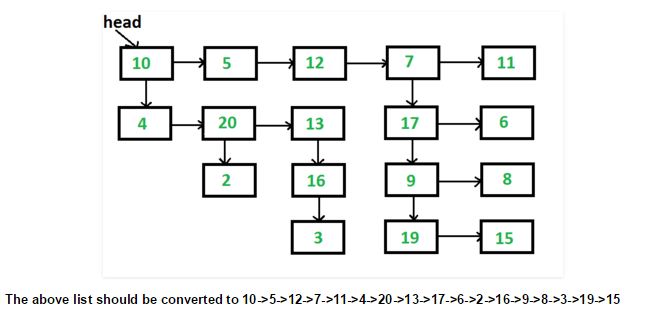
SECTION 2 (For this section, you have to create your own structure. You are not allowed to copy/paste previous template. If found doing so, you will be debarred forever. Please don’t try to fool the evaluators by simply using an array and printing the answer. You will be surely caught if you do so as the evaluator will check your structure very carefully and you will be debarred forever)

Q.1. Flatten a multilevel linked list

Given a linked list where in addition to the next pointer, each node has a child pointer, which may or may not point to a separate list. These child lists may have one or more children of their own, and so on, to produce a multilevel data structure, as shown in below figure.You are given the head of the first level of the list. Flatten the list so that all the nodes appear in a single-level linked list. You need to flatten the list in way that all nodes at first level should come first, then nodes of second level, and so on.

Each node is a C struct with the following definition.

|  |
| --- |
| struct list  {      int data;      struct list \*next;      struct list \*child;  }; |



**Asked by Amazon** (50 Marks)

Expected Time Complexity: O(total\_no\_of\_nodes)

Expected Space Complexity: O(1)