**Largest number possible: -**

Easy Accuracy: 20.26% Submissions: 104K+ Points: 2

Given two numbers '**N**' and '**S**' , find the **largest number** that can be formed with '**N**' digits and whose sum of digits should be equals to '**S**'. Return -1 if it is not possible.

**Example 1:**

**Input:** N = 2, S = 9

**Output:** 90

**Explaination:** It is the biggest number

with sum of digits equals to 9.

**Example 2:**

**Input:** N = 3, S = 20

**Output:** 992

**Explaination:** It is the biggest number

with sum of digits equals to 20.

**Your Task:**  
You do not need to read input or print anything. Your task is to complete the function **findLargest()** which takes N and S as input parameters and returns the largest possible number. Return**-1** if no such number is possible.

**Expected Time Complexity:** O(N)  
**Exepcted Auxiliary Space:** O(1)

**Constraints:**  
1 ≤ N ≤ 104  
0 ≤ S ≤ 105

**Code: -**

//{ Driver Code Starts

// Initial Template for C++

#include <bits/stdc++.h>

using namespace std;

// } Driver Code Ends

// User function Template for C++

class Solution{

public:

string findLargest(int N, int S){

if(S==0 and N>1) return "-1";

string ans = "";

while(N--){

if(S > 9){

ans.push\_back('9');

S = S - 9;

}

else{

ans.push\_back('0' + S);

S = S - S;

}

}

if(S == 0) return ans;

else return "-1";

}

};

//{ Driver Code Starts.

int main(){

int t;

cin>>t;

while(t--){

int N, S;

cin>>N>>S;

Solution ob;

cout<<ob.findLargest(N, S)<<"\n";

}

return 0;

}

// } Driver Code Ends

**T.C: - O(N), N = count of digits**

**S.C: - O(1)**