Term Project

**Subject**: Algorithm Design (SC3231)

Submitted to

Asst. Prof. Dr. Thitipong Tanprasert

Submitted by

Vorapat Phorncharroenroj  5612690
Pitiwat Arunruviwat  5613739

This report is part of the subject Algorithm Design

Semester 1/2016
Problem: Super long Sum

Difficult: 183

Time limit: 2.0 second
Memory Limit: 16MB

Input
The first line contains a single integer $N$ that is the length of the given integers (1 ≤ $N$ ≤ 1 000 000). It is followed by these integers written in columns. That is, the next $N$ lines contain two digits each, divided by a space. Each of the two given integers is not less than 1, and the length of their sum does not exceed $N$. The integers may contain leading zeroes.

Output
Output exactly $N$ digits in a single line representing the sum of these two integers.

Input
Output
4 4750
0 4
4 2
6 8
3 7

Input
Output
4 0024
0 0
0 0
1 1
2 2

Input
Output
3 417
1 2
5 6
4 3

Input
Output
4 7862
3 4
1 7
2 4
1 1
Solution

```java
@SuppressWarnings("deprecation")
public static void main(String[] args) throws NumberFormatException, IOException {
    StreamTokenizer in = new StreamTokenizer(new BufferedReader(System.in));

    int n, a, b;
    in.nextToken();
    n = (int) in.nval;

    byte[] c = new byte[n];
    for (int i = 0; i < n; i++) {
        in.nextToken();
        a = (int) in.nval;
        in.nextToken();
        b = (int) in.nval;
        c[i] = (byte) (a + b);
    }
    for (int i = n - 1; i > 0; i--) {
        if (c[i] >= 10) {
            c[i] -= 10;
            ++c[i - 1];
        }
        c[i] += '0';
    }
    c[0] = (byte) (((int) c[0]) % 10);
    c[0] += '0';
    System.out.println(new String(c));
}
```

Example Execution

```
in.nextToken();
    n = (int) in.nval;

    byte[] c = new byte[n];

    for (int i = 0; i < n; i++) {
        in.nextToken();
        a = (int) in.nval;
        in.nextToken();
        b = (int) in.nval;
        c[i] = (byte) (a + b);
    }
```

Use to read input that split by space work with StreamTokenizer.

Keep sum of input value into array in each column.
for (int i = n - 1; i > 0; i--) {
    if (c[i] >= 10) {
        c[i] -= 10;
        ++c[i - 1];
    }
    c[i] += '0';
}

```java
C[0] = (byte) (((int) C[0]) % 10);
C[0] += '0';
System.out.println(new String(C));
```  

Test Case

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>0000</td>
</tr>
<tr>
<td>8 1</td>
<td></td>
</tr>
<tr>
<td>4 5</td>
<td></td>
</tr>
<tr>
<td>2 7</td>
<td></td>
</tr>
<tr>
<td>7 3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>07133</td>
</tr>
<tr>
<td>1 9</td>
<td></td>
</tr>
<tr>
<td>2 4</td>
<td></td>
</tr>
<tr>
<td>3 7</td>
<td></td>
</tr>
<tr>
<td>8 4</td>
<td></td>
</tr>
<tr>
<td>5 8</td>
<td></td>
</tr>
</tbody>
</table>

In this 2 pictures, it is an example of test case. It shown that user input n (integer) in first line that is length of output. When output is out of length of input in first line, it will cut it off.
Time Analysis

Time that use to solve this question is $O(2n)$ because we use 2 loops to load the input follow by n input.

```java
for (int i = 0; i < n; i++) {
    in.nextToken();
    a = (int) in.nval;
    in.nextToken();
    b = (int) in.nval;
    c[i] = (byte) (a + b);
}
for (int i = n - 1; i > 0; i--) {
    if (c[i] >= 10) {
        c[i] -= 10;
        ++c[i - 1];
    }
```
Term Project

Algorithm Design (SC3231)

1048. Superlong Sums

Difficulty: 183

Assumption University
Find the sum of two numbers with maximal size of 1,000,000 digits which problem cannot use type integer to keep them because limit of its is 1,000 digits
- First line is integer N is length (1 <= n <= 1,000,000)
- Next N lines contain 2 digits each, split by space.
- Result : output exactly N digits —> the sum of these two integers.

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 4</td>
<td>4750</td>
</tr>
<tr>
<td>4 2</td>
<td></td>
</tr>
<tr>
<td>6 8</td>
<td></td>
</tr>
<tr>
<td>3 7</td>
<td></td>
</tr>
</tbody>
</table>
### Problem

**Time limit:** 2.0 second  
**Memory limit:** 16 MB

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 8 1 4 5 2 7 7 3</td>
<td>0000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 0 0 0 0 1 1 2 2</td>
<td>0024</td>
</tr>
</tbody>
</table>
```python
a = 0
b = 0
result = ""
n = input()

n1 = ""
n2 = ""
count = 0

if n >= 1 and n <= 1000000:
    for i in range(n):
        num = raw_input().split()
        if count == 0:
            n1 = n1 + "" + str(int(num[0]) + int(num[1]))
        else:
            n1 = n1 + "," + str(int(num[0]) + int(num[1]))
        count = 1

print n1

for i in range(n-1,-1,-1):
    k = int(n1.split(\'/\')[i])
    if i == n-1:
        a = k
        b = 0
    if n == 1:
        if a >= 10:
            a = a-10
        result = str(a)
    else:
        b = a
        a = k
        if b >= 10:
            a = a + 1
            b = b - 10
        result = str(b) + result
    if i == 0:
        if a >= 10:
            a = a-10
            result = str(a) + result

print result
```

**1048. Superlong Sums**

<table>
<thead>
<tr>
<th>Python 2.7</th>
<th>Memory limit exceeded</th>
<th>Time</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>0.078</td>
<td>16588 KB</td>
</tr>
</tbody>
</table>
Problem 1048. Superlong Sums

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solve</th>
</tr>
</thead>
<tbody>
<tr>
<td>Python 2.7</td>
<td>Memory limit exceeded</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solve</th>
</tr>
</thead>
<tbody>
<tr>
<td>String = “”</td>
<td>String = “”</td>
</tr>
</tbody>
</table>

**Loop**

Problem: `for i in range(n):`

Solve: `while(i < n):`

- It uses 1 of n per 8 bytes

**Keeping Number**

Problem: `string = “”`

Solve: `arr = bytearray(n)`
Python

```python
import sys
# from array import array
import time

st = time.clock()

n = input()

# arr = array('B')
arr = bytearray(n)
i = 0

while i < n:
    line = raw_input().split()
    arr[i] = (int(line[0]) + int(line[1]))
i += 1

result = ""
i = n-1
while i > 0:
    if arr[i] > 9:
        arr[i] -= 10
        arr[i-1] += 1
    arr[i] = str(arr[i])
i -= 1

arr[0] = arr[0]%10
arr[0] = str(arr[0])

print arr

print time.clock()-st

1048. Superlong Sums  Python 2.7  Time limit exceeded  4  2.028  1132 KB
```
Java

- After calculate convert numbers to byte
- keep numbers with using array byte instead of String to concat the numbers
- use print array byte in library instead of loop print
THANK YOU

Assumption University