Consider the following ciphering algorithm:

* For each character replace it with its code.
* Concatenate all of the obtained numbers.

Given a ciphered string, return the initial one if it is known that it consists only of lowercase letters.

**Note:** here the *character's code* means its decimal ASCII code, the numerical representation of a character used by most modern programming languages.

**Example**

For cipher = "10197115121", the output should be  
decipher(cipher) = "easy".

Explanation: charCode('e') = 101, charCode('a') = 97, charCode('s') = 115 and charCode('y') = 121.

**Input/Output**

* **[time limit] 3000ms (cs)**
* **[input] string cipher**

A non-empty string which is guaranteed to be a cipher for some other string of lowercase letters.

*Constraints:*  
2 ≤ cipher.length ≤ 100.

* **[output] string**

<https://codefights.com/arcade/code-arcade/lab-of-transformations/dB9drnfWzpiWznESA>

static string decipher(string cipher)

{

string ans = "";

int i = 0;

while (i < cipher.Length)

{

string concat = "0";

while (int.Parse(concat) < 97)

{

concat += cipher[i];

i++;

}

ans += (char)int.Parse(concat);

}

return ans;

}