21/01/08

1. 문자열 압축

**문제 설명**

데이터 처리 전문가가 되고 싶은 **어피치**는 문자열을 압축하는 방법에 대해 공부를 하고 있습니다. 최근에 대량의 데이터 처리를 위한 간단한 비손실 압축 방법에 대해 공부를 하고 있는데, 문자열에서 같은 값이 연속해서 나타나는 것을 그 문자의 개수와 반복되는 값으로 표현하여 더 짧은 문자열로 줄여서 표현하는 알고리즘을 공부하고 있습니다.  
간단한 예로 aabbaccc의 경우 2a2ba3c(문자가 반복되지 않아 한번만 나타난 경우 1은 생략함)와 같이 표현할 수 있는데, 이러한 방식은 반복되는 문자가 적은 경우 압축률이 낮다는 단점이 있습니다. 예를 들면, abcabcdede와 같은 문자열은 전혀 압축되지 않습니다. 어피치는 이러한 단점을 해결하기 위해 문자열을 1개 이상의 단위로 잘라서 압축하여 더 짧은 문자열로 표현할 수 있는지 방법을 찾아보려고 합니다.

예를 들어, ababcdcdababcdcd의 경우 문자를 1개 단위로 자르면 전혀 압축되지 않지만, 2개 단위로 잘라서 압축한다면 2ab2cd2ab2cd로 표현할 수 있습니다. 다른 방법으로 8개 단위로 잘라서 압축한다면 2ababcdcd로 표현할 수 있으며, 이때가 가장 짧게 압축하여 표현할 수 있는 방법입니다.

다른 예로, abcabcdede와 같은 경우, 문자를 2개 단위로 잘라서 압축하면 abcabc2de가 되지만, 3개 단위로 자른다면 2abcdede가 되어 3개 단위가 가장 짧은 압축 방법이 됩니다. 이때 3개 단위로 자르고 마지막에 남는 문자열은 그대로 붙여주면 됩니다.

압축할 문자열 s가 매개변수로 주어질 때, 위에 설명한 방법으로 1개 이상 단위로 문자열을 잘라 압축하여 표현한 문자열 중 가장 짧은 것의 길이를 return 하도록 solution 함수를 완성해주세요.

**제한사항**

* s의 길이는 1 이상 1,000 이하입니다.
* s는 알파벳 소문자로만 이루어져 있습니다.

입출력 예

| **s** | **result** |
| --- | --- |
| "aabbaccc" | 7 |
| "ababcdcdababcdcd" | 9 |
| "abcabcdede" | 8 |
| "abcabcabcabcdededededede" | 14 |
| "xababcdcdababcdcd" | 17 |

**입출력 예에 대한 설명**

**입출력 예 #1**

문자열을 1개 단위로 잘라 압축했을 때 가장 짧습니다.

**입출력 예 #2**

문자열을 8개 단위로 잘라 압축했을 때 가장 짧습니다.

**입출력 예 #3**

문자열을 3개 단위로 잘라 압축했을 때 가장 짧습니다.

**입출력 예 #4**

문자열을 2개 단위로 자르면 abcabcabcabc6de 가 됩니다.  
문자열을 3개 단위로 자르면 4abcdededededede 가 됩니다.  
문자열을 4개 단위로 자르면 abcabcabcabc3dede 가 됩니다.  
문자열을 6개 단위로 자를 경우 2abcabc2dedede가 되며, 이때의 길이가 14로 가장 짧습니다.

**입출력 예 #5**

문자열은 제일 앞부터 정해진 길이만큼 잘라야 합니다.  
따라서 주어진 문자열을 x / ababcdcd / ababcdcd 로 자르는 것은 불가능 합니다.  
이 경우 어떻게 문자열을 잘라도 압축되지 않으므로 가장 짧은 길이는 17이 됩니다.

* 내 답

#include <string>

#include <vector>

#include <cctype>

#include <iostream>

using namespace std;

int solution(string s) {

int answer = 0;

int min\_len = s.size();

for(int i=0;i<s.size()/2;i++) {

int len;

int num\_len=0;

string p\_s;

p\_s = s.substr(0,i+1);

for(int j=i+1;j<s.size();) {

if(s.substr(j,i+1) == p\_s.substr(p\_s.size()-(i+1),i+1))

if(isdigit(p\_s[p\_s.size()-(i+2)])) {

int num = stoi(p\_s.substr(p\_s.size()-(i+1)-num\_len,num\_len));

num++;

p\_s = p\_s.substr(0,p\_s.size()-(i+1+num\_len))+to\_string(num)+p\_s.substr(p\_s.size()-(i+1),i+1);

num\_len = (to\_string(num)).size();

}

else {

p\_s = p\_s.substr(0,p\_s.size()-(i+1))+"2"+p\_s.substr(p\_s.size()-(i+1),i+1);

num\_len=1;

}

else

p\_s += s.substr(j,i+1);

j+=(i+1);

}

len = p\_s.size();

if(len < min\_len)

min\_len = len;

}

return min\_len;

}

예외 케이스를 생각을 잘 못했는데, 첫번째로 9가 넘어가면 string 한 자리에 10이상이 못들어간느 경우를 생각 못했다. to\_string, stoi 등을 잘 활용해야 할 것 같다. 그리고 인덱스를 잘 생각을 해서 null\_point exception이 발생하지 않도록 유의해야 한다.