C++解压报文-为了提升数据传输的效率

题目描述：

为了提升数据传输的效率，会对传输的报文进行压缩处理。输入一个压缩后的报文，请返回它解压后的原始报文。

压缩规则：n[str]，表示方括号内部的 str 正好重复 n 次。注意 n 为正整数（0 < n <= 100），str 只包含小写英文字母，不考虑异常情况。

输入描述：

输入压缩后的报文：

1）不考虑无效的输入，报文没有额外的空格，方括号总是符合格式要求的；

2）原始报文不包含数字，所有的数字只表示重复的次数 n ，例如不会出现像 5b 或 3[8] 的输入；

输出描述：

解压后的原始报文

注：

1）原始报文长度不会超过 1000，不考虑异常的情况

补充说明：


```
#include <bits/stdc++.h>
using namespace std;


#define DEBUG 0
#define SINGLE 1
#define cstr(x) (luangao(x).c_str())
#define fastio cin.tie(0) -> sync_with_stdio(0)

void debug(const char* p){
    #if DEBUG
    freopen(p, "r", stdin);
```

```
    #else
    //fastio;
    #endif
}

string s;
string ans;

string dfs(int l, int r){
    //printf("l==%d, r==%d, s==%s\n", l, r, s.c_str());
    if(l > r) return "";
    size_t left_bracket = s.find_first_of('[', l);
    size_t right_bracket = left_bracket;
    if(left_bracket > r){
        return s.substr(l, r-l+1);
    }
    int tot = 0;
    for(int i = (int)left_bracket; i <= r; ++i){
        if(s[i] == '['){
            tot++;
        }else if(s[i] == ']'){
            tot--;
            if(tot == 0){
                right_bracket = i;
                break;
            }
        }
    }
    int base = 1;
    int coeff = 0;
    int digitarea = left_bracket-1;
    for(; digitarea >= l; --digitarea){
        if(s[digitarea] >= '0' && s[digitarea] <= '9'){
            coeff += (s[digitarea] - '0') * base;
            base *= 10;
        }else{
            break;
        }
    }
    std::string ans;
    for(int i = l; i <= digitarea; ++i) ans += s[i];
    string dfsmiddle = dfs(left_bracket+1, right_bracket-1);
    for(int i = 0; i < coeff; ++i){
        ans += dfsmiddle;
```

```
        }
        ans += dfs(right_bracket+1, r);
        return ans;
}

void solve(int test){
        cin >> s;
        ans = dfs(0, (int)s.size()-1);
        cout << ans << "\n";
}

signed main(int argc, char** argv){
        debug(argc==1?"test1.txt":argv[1]);
        int t = 1;
        int test = 1;
        while(t--){
                solve(test++);
        }
}
```