ACM ICPC 2015 国内予選　問題 C ICPC 計算機

【考察】

- 一つの式に含まれる整数の個数は９個以下であるので、演算子の個数は８個以下である。したがって、一つの式の行数は 17 以下となる。
- 演算子の個数は高々８なので、入れ子レベルも高々８である。また、整数は１桁なので、１行の長さは高々９である。
- 入れ子レベルが level の時の処理
  - 加算や乗算の場合
    - チェックしている行の入れ子レベルが level 以下なら加算や乗算終了なので、計算結果を返す
    - チュックしている行の入れ子レベルが level+1 なら、その行の評価結果を再帰的に求め、加算や乗算を行う。
    - チェックしている行の入れ子レベルがそれ以外（level+1 より大きい）の場合は、既に再帰的に評価が行われているので、何もする必要は無い。
    - 最終行でなければ、次の行以降を繰り返し評価する。
  - 整数の場合
    - 対応する数値を返す。

【プログラム例 pc.c】

```
// ACM-ICPC 2015 国内予選 Problem C ICPC 計算機
// http://icpc.iisf.or.jp/past-icpc/domestic2015/contest/all_ja.html#section_C
//   Filename:      pc.c
//   Compile:       gcc pc.c
//   Execution:     ./a.out < C0 > C0.result
//   Check:         diff C0.ans C0.result
//   Algorithm:     式の各行の値を再帰的に求める

#include <stdio.h>
#include <string.h>

#define NLINE 17                        // 式の行数の最大値
#define LEN 9                           // 式の 1 行の長さの最大値

int n;                                  // 式の行数
char buff[NLINE][LEN+1];                // 式を格納する配列

// line 行目を評価して値を返す関数
int calc(int line)
{
  char c;                               // 処理内容を示す文字（数字, *, +）
  int level;                            // 現在の行の入れ子レベル
  int val;                              // 評価結果を入れる変数
  int arg;                              // 演算の引数の値を入れる変数
  level = strlen(buff[line]) - 1;       // 自分の入れ子レベル
  c = buff[line][level];                // c はこのレベルでの処理内容
  switch(c){
  case '*':                             // 乗算
    line++;                                     // 次の行から引数の値を順に評価
    val = 1;                                    // とりあえず結果を 1 とする
    for(;;){
      if(strlen(buff[line])-1 <= level) break;  // 引数の評価は終了
      if(strlen(buff[line])-1 == level+1){      // 評価すべきレベルなら
        arg = calc(line);                       // 引数の値を再帰的に求め
        val *= arg;                             // val にかける。
      }
      line++;                                   // 次の行へ
      if(line >= n) break;                      // 全ての行をチェックした！
    }
    return val;                                 // 評価結果を返す
  case '+':                             // 加算
    line++;                                     // 次の行から引数の値を順に評価
    val = 0;                                    // とりあえず結果を 0 とする
    for(;;){
      if(strlen(buff[line])-1 <= level) break;  // 引数の評価は終了
      if(strlen(buff[line])-1 == level+1){      // 評価すべきレベルなら
```

```
            arg = calc(line);                   // 引数の値を再帰的に求め
            val += arg;                         // val にたす。
        }
        line++;                                 // 次の行へ
        if(line >= n) break;                    // 全ての行をチェックした！
      }
      return val;                               // 評価結果を返す
    default:                                // 1 桁の整数
      return c - '0';                           // 対応する整数値を返す
    }
}

int main(void)
{
    for(;;){
      int i;
      scanf("%d", &n);                          // 式の行数 n を読み込む
      if(n==0) break;                           // n が 0 なら終了
      for(i=0; i<n; i++)                        // n 行分の式を読み込む
        scanf("%s", &buff[i][0]);
      printf("%d¥n", calc(0));                  // 式を評価した結果をプリント
    }
}
```