EnumString.h

```
/*
 * File:   EnumString.h
 * Author: tadaki
 *
 * Created on 2013/05/30, 13:21
 */
#include <list>
#ifndef EnumString_H
#define EnumString_H
using namespace std;

class EnumString {
public:
    EnumString(const char[], int);
    EnumString(const EnumString& orig);
    virtual ~EnumString();
    int enumStringMain();
    list<char*> getStringList();
private:
    int enumString(list<char>, int,list<char>);
    char* charList;
    list<char*> strList;
    bool debug;
    char* listToStr(list<char>);
};

#endif  /* EnumString_H */
```

EnumString.cpp

```
/*
 * File:   EnumString.cpp
 * Author: tadaki
 *
 * Created on 2013/05/30, 13:21
 */

#include "EnumString.h"

EnumString::EnumString(const char chars[], int n) {
    charList = new char[n + 1];
    for (int i = 0; i < n; i++)charList[i] = chars[i];
    charList[n] = '¥0';
    debug = true;
}

EnumString::EnumString(const EnumString& orig) {
}

EnumString::~EnumString() {
}

int EnumString::enumStringMain() {
    //最長文字列に対応した文字列領域を確保し、NULLを書き込む
    char* str;
    list<char> strListTmp;
    str = strdup(charList);
    list<char> available;
    for (int i = 0; i < strlen(charList); i++) {
        available.push_back(charList[i]);
    }
    strList.clear(); //文字列一覧の消去
    return enumString(strListTmp, 0, available); //列挙開始
}

/**
 * 文字列列挙
 * @param strListTmp 生成中の文字のリスト
 * @param n   生成した文字列の総数
 * @param available 使用できる文字のリスト
 * @return    生成した文字列総数
 */
int EnumString::enumString(list<char> strListTmp,
        int n, list<char> available) {
    char* str = listToStr(strListTmp);
    if (debug) {
```

```
            cout << str;
            cout << " ";
        }
        if (available.size() == 0) {//使用できる文字が残っていない
            strList.push_back(strdup(str));
            n++;
            if (debug) {
                cout << "¥n";
            }
        } else {
            //使用できる文字列のリストのコピー
            list<char> newAvailable;
            for (list<char>::iterator j = available.begin();
                    j != available.end(); j++) {
                newAvailable.push_back(*j);
            }

            for (list<char>::iterator j = available.begin();
                    j != available.end(); j++) {
                strListTmp.push_back(*j);
                newAvailable.remove(*j);
                n = enumString(strListTmp, n, newAvailable);
                strListTmp.pop_back();
                newAvailable.push_back(*j);
            }
        }
        return n;
}

/**
 * 文字リストから文字列生成
 * @param in 文字リスト
 * @return 文字列
 */
char* EnumString::listToStr(list<char> in) {
    int m = in.size();
    char* str = new char[m + 1];
    int j = 0;
    for (list<char>::iterator i = in.begin(); i != in.end(); i++) {
        str[j] = *i;
        j++;
    }
    str[m] = '¥0';
    return str;
}
```

EnumString.cpp

```
list<char*> EnumString::getStringList() {
    return strList;
}
```