

ListCS22fp_out.cpp

```
//ListCS22fp_out
//.NET用のinclude//
#include <iostream>
#include <iomanip>
#include <fstream>

//プログラム記述の簡略化//
using namespace std;

/*Visual C++6.0用のinclude
#include <iostream.h>
#include <iomanip.h>
#include <fstream.h>
*/

//プロトタイプ宣言//
int ceil(int a, int b);
int floor(int a, int b);

// 考えられる列または行を超える数字を入れること //
/*
const int CTTL = 1090000;
const int RTTL = 5000;
*/
const int CTTL = 1100000;
const int RTTL = 5000;

char pb_name[65], unicost, csttype[17];

int rttl, ctll, ww;
int i, j, k, nCount;
unsigned int G[CTTL][(RTTL / 32) + 1];
/*
データサイズの計算
CTTL * ((RTTL / 32) + 1) * 4
RTTL = 5000, CTTL = 1090000の時、
1.09M * (5000 / 32 + 1) * 4 = 1.09M * 160 * 4 = 700MB
*/

int cst[CTTL + 10];
double rdensity;
int h[CTTL], rname[500];
/*
rname[]の大きさは、
densityの計算で予測する。
*/

//配列はすべて0スタートで計算//
int main() {
    //ファイル読み込み.NET用//
    ifstream fin;
    fin.open("ListCS2.txt");
    if(fin.fail()) {
        cerr << "Cannot open the ListC file.";
        return 1;
    }
}
```

ListCS22fp_out.cpp

```

ofstream fout;
fout.open("fp_out.txt");
if(fout.fail()){
    cerr << "Cannot open the fp_out file.";
    return 2;
}
/*ファイル読み込みVisual C++6.0用
ifstream fin("ListC.txt");
if(!fin){
    cerr << "Cannot open the ListC file.";
    return 1;
}
ofstream fout("fp_out.txt");
if(!fout){
    cerr << "Cannot open the fp_out file.";
    return 2;
}
*/

//データ読み込み//
fin >> rttl >> cttl;
cout <<"rttl = " << rttl <<" cttl = " << cttl << endl;

//G[]のゼロクリア//
for( i = 0 ; i < ceil(rttl , 32) ; i++)
    for( j = 0 ; j < cttl ; j++)
        G[j][i] = 0;

cout <<"0 clear completion of G[[]]." << endl;

//非ゼロ個数のゼロクリア//
int nCount = 0;

//ListCからfp_outへのデータ変換作業//
for( k = 0 ; k < cttl ; k++){
    fin >> cst[k] >> h[k];
    nCount += h[k];
    for( i = 0 ; i < h[k] ; i++){
        fin >> rname[i];
        G[k][(rname[i] - 1) / 32] = G[k][(rname[i] - 1)
/ 32] | (1U << (31 - ((rname[i] - 1) % 32)));
    }
    //余った部分には1を代入という約束//
    for( j = rttl ; j < (ceil(rttl , 32) * 32) ; j++){
        G[k][j / 32] = G[k][j / 32] | (1U << (31 - (j % 32)));
    }
    for( j = rname[h[k] - 1] ; j < (ceil(rttl , 32) * 32) ; j++){
        G[k][(j + 1) / 32] = G[k][(j + 1) / 32] | (1U <<
(31 - (j % 32)));
    }
}
//rdensityの計算//
rdensity = nCount / ((double)(rttl) * (double)(cttl)); //もしかしたらrttl
* cttlの計算で桁数が足りなくなるかも//

cout <<"Data change working completion from ListC to fp_out.¥n" << endl;

```

ListCS22fp_out.cpp

```

//cst[]の10単位出力のための強引計算式//
if(cttl % 10) {
    for(k = 0 ; k < 10 - (cttl % 10) ; k++) {
        cst[cttl + k] = -1;
    }
}

//pb_nameの入力//
cout << "Input pb_name with in 64 characters"<< endl;
cin >> pb_name;

//cstypeの入力//
cout << "Input cstype with in 16 characters"<< endl;
cin >> cstype;

cout <<"An output working start to fp_out."<< endl;

//fp_out.txtへの出力//
fout << pb_name <<" " << rttl <<" " << cttl <<" " << cstype <<" " <<
rdensity << endl;

//cst[]出力//
for( i = 0 ; i < ceil(cttl , 10) ; i++) {
    fout << cst[(10 * i) + 0] << " " << cst[(10 * i) + 1] << " "
        << cst[(10 * i) + 2] << " " << cst[(10 * i) + 3] << " "
        << cst[(10 * i) + 4] << " " << cst[(10 * i) + 5] << " "
        << cst[(10 * i) + 6] << " " << cst[(10 * i) + 7] << " "
        << cst[(10 * i) + 8] << " " << cst[(10 * i) + 9] <<
endl;
}
//G[][]出力//
for( j = 0 ; j < cttl ; j++) {
    for( i = 0 ; i < ceil(rttl , 32) ; i++) {
        fout << G[j][i] << endl;
    }
}

cout <<"An output working end to fp_out."<< endl;

//ファイルクローズ//
fin. close();
fout. close();
}
//プロトタイプ宣言の計算式//
int ceil(int a, int b) {
    if(a % b)
        return (a / b) + 1;

    return a / b;
}
int floor(int a, int b) {
    return a / b;
}

```