Изначально производится чтение файла и размещение результатов ответов испытуемых в пользовательский тип данных, представляющий и себя пару – массив битов (ответы ‘+’ – 1, ‘-’ - 0) и значение (полученный балл):

typedef pair<int,bitset<1000>> oneresult;

…

//создание структуры для хранения входных данных

oneresult \*data = new oneresult[count\_var];

//создание структуры для ответа

oneresult \*out = new oneresult;

//пока не конец файла и не все варианты просмотрены

while(!file.eof() && i<count\_var)

{

int curball,it = 0;

string curstr;

do{

getline(file,curstr);

}while(curstr.empty());

//считываем текущий балл за тест

file>>curball;

if(file.fail()) break;

data[i].first = curball;

while(curstr[it])

{

if(curstr[it]=='+') data[i].second[it]=1;

else if(curstr[it]=='-') data[i].second[it]=0;

else cerr << "Invalid data";

it++;

}

i++;

}

При анализе входных данных рассматриваются ответы попарно каждого из двух участников, находятся в них отличия за счет булевой операции «Исключающее ИЛИ», учитывая разность полученных участниками баллов и разность их ответов, каждому ответу присваивается некоторый относительный «вес», отражающий вероятность его правильности:

void analyze(oneresult \*data,oneresult \*out)

{

//коэффициент вероятности успеха,

int koeff=0,

//число отличий в ответах двух разных испытуемых

commonans =0,

//разность итоговых баллов двух разных испытуемых

relball = 0;

//массив битов для операции "Исключающее ИЛИ"

bitset<1000> cmp(0);

//массив вычисления вероятности правильности ответа

float sucess[1000];

memset(sucess,0,1000);

//для каждой пары испытуемых

for(int i=0;i<count\_var;i++)

{

for(int j=i+1;j<count\_var;j++)

{

commonans = 0;

koeff = 0;

//вычисляем результрующий вектор для

//операции "Исключающее ИЛИ"

cmp = data[i].second ^ data[j].second;

relball = data[i].first - data[j].first;

for(int k=0;k<count\_ans;k++)

{

if(cmp[k])commonans++;

}

//если количество отличий совпадает с разностью баллов

//значит можно с полной уверенностью узнать верный ответ

if(int(abs(relball)) == commonans) koeff = 1000;

for(int k=0;k<count\_ans;k++)

{

if(!cmp[k]) continue;

//если есть отличия

if(relball>0)

{

//если балл первого в паре больше, чем у второго

if(data[i].second[k]) sucess[k]+=koeff\*relball/commonans;

else sucess[k]-=koeff\*relball/commonans;

}

else if(relball<0)

{

//если балл первого в паре меньше, чем у второго

if(data[j].second[k]) sucess[k]-=koeff\*relball/commonans;

else sucess[k]+=koeff\*relball/commonans;

}

}

}

}

//составив сводный массив вероятностей sucess,

//можно выдавать рекоммендации по ответам на тест

cout << "Answer:" << endl;

for(int k=0;k<count\_ans;k++)

{

if(sucess[k]>0) out->second[k] = 1;

else out->second[k] = 0;

out->second[k] ? cout << "+": cout << "-";

cout << " ";

}

cout << endl;

}

Исходный код программы:

#include <iostream>

#include <ctime>

#include <vector>

#include <fstream>

#include <windows.h>

#include <bitset>

#include <ctime>

#include <math.h>

using namespace std;

typedef pair<int,bitset<1000>> oneresult;

//кол-во вариантов, кол-во вопросов в тесте

int count\_var=0,

count\_ans=0;

void analyze(oneresult \*data,oneresult \*out)

{

//коэффициент вероятности успеха,

int koeff=0,

//число отличий в ответах двух разных испытуемых

commonans =0,

//разность итоговых баллов двух разных испытуемых

relball = 0;

//массив битов для операции "Исключающее ИЛИ"

bitset<1000> cmp(0);

//массив вычисления вероятности правильности ответа

float sucess[1000];

memset(sucess,0,1000);

//для каждой пары испытуемых

for(int i=0;i<count\_var;i++)

{

for(int j=i+1;j<count\_var;j++)

{

commonans = 0;

koeff = 0;

//вычисляем результрующий вектор для

//операции "Исключающее ИЛИ"

cmp = data[i].second ^ data[j].second;

relball = data[i].first - data[j].first;

for(int k=0;k<count\_ans;k++)

{

if(cmp[k])commonans++;

}

//если количество отличий совпадает с разностью баллов

//значит можно с полной уверенностью узнать верный ответ

if(int(abs(relball)) == commonans) koeff = 1000;

for(int k=0;k<count\_ans;k++)

{

if(!cmp[k]) continue;

//если есть отличия

if(relball>0)

{

//если балл первого в паре больше, чем у второго

if(data[i].second[k]) sucess[k]+=koeff\*relball/commonans;

else sucess[k]-=koeff\*relball/commonans;

}

else if(relball<0)

{

//если балл первого в паре меньше, чем у второго

if(data[j].second[k]) sucess[k]-=koeff\*relball/commonans;

else sucess[k]+=koeff\*relball/commonans;

}

}

}

}

//составив сводный массив вероятностей sucess,

//можно выдавать рекоммендации по ответам на тест

cout << "Answer:" << endl;

for(int k=0;k<count\_ans;k++)

{

if(sucess[k]>0) out->second[k] = 1;

else out->second[k] = 0;

out->second[k] ? cout << "+": cout << "-";

cout << " ";

}

cout << endl;

}

//функция чтения из файла

bool rfile(const string& name)

{

ifstream file;

file.open(name);

int i =0;

if(file.is\_open()) {

file >> count\_ans >> count\_var ;

if(count\_var>1000 || count\_ans>15)

{

cerr << "Very high input data" << endl;

return 0;

}

//создание структуры для хранения входных данных

oneresult \*data = new oneresult[count\_var];

//создание структуры для ответа

oneresult \*out = new oneresult;

//пока не конец файла и не все варианты просмотрены

while(!file.eof() && i<count\_var)

{

int curball,it = 0;

string curstr;

do{

getline(file,curstr);

}while(curstr.empty());

//считываем текущий балл за тест

file>>curball;

if(file.fail()) break;

data[i].first = curball;

while(curstr[it])

{

if(curstr[it]=='+') data[i].second[it]=1;

else if(curstr[it]=='-') data[i].second[it]=0;

else cerr << "Invalid data";

it++;

}

i++;

}

file.close ();

//вывод на экран входных данных

for(int i=0;i<count\_var;i++)

{

cout << data[i].first << " ";

for(int j=0;j<count\_ans;j++)

if(data[i].second[j]) cout << "+" << " ";

else cout << "-" << " ";

cout << endl;

}

//анализ входных данных

analyze(data,out);

return 1;

}

else

{

cerr << "File not found" << endl;

return 0;

}

}

int main(int argc, char \*argv[])

{

//время выполнения

int start = clock();

//чтение из файла

if(!rfile("input.txt")) return 0;

//останов

int stop = clock();

cout << ((float)(stop-start)) / CLOCKS\_PER\_SEC << " seconds" << endl;

return 0;

}