|  |
| --- |
| main.cpp |

#define endl '\n'

#define dtype long long

#define mod 1000000007

#define case(t, c) cout<<"Case "<<(t)<<":"<<c

int main() {

ios\_base::sync\_with\_stdio(false), cin.tie(nullptr);

#ifndef ONLINE\_JUDGE

freopen("input.txt", "r", stdin);

freopen("output.txt", "w", stdout);

#endif

return 0;

}

|  |
| --- |
| gf.cpp |

#define endl '\n'

#define hi 100000

#define lo 0

#define inp dist6(rng)

int main() {

ios\_base::sync\_with\_stdio(false), cin.tie(nullptr);

freopen("input.txt", "w", stdout);

auto seed = chrono::high\_resolution\_clock::now().time\_since\_epoch().count();

std::mt19937 rng(seed);

std::uniform\_int\_distribution<int> dist6(lo, hi); // distribution in range [1, 6]

}

|  |
| --- |
| bf.cpp |

#define fileh freopen("input.txt", "r", stdin),freopen("correctoutput.txt", "w", stdout);

int main() {

fileh;

ios\_base::sync\_with\_stdio(false), cin.tie(nullptr);

}

|  |
| --- |
| test.cpp |

void getans(ifstream &x, vector<string> &vc) {

string str, word;

while (getline(x, str)) {

stringstream ss(str);

while (ss >> word)

vc.emplace\_back(word);

}

}

int main() {

try {

system("g++ -o zgf gf.cpp");

system("zgf.exe");

system("g++ -o main main.cpp");

system("main.exe");

system("g++ -o zbf bf.cpp");

system("zbf.exe");

ifstream out("output.txt"), crout("correctoutput.txt"), in("input.txt");

vector<string> vc1, vc2;

getans(out, vc1);

int lim=max(vc1.size(),vc2.size());

int inp\_startsfrom=1,inpsize=1,acsize=1;

for (int i = 0; i <lim ; ++i) {

if (vc1[i] != vc2[i]) {

string str;

for (int j = 0; j <inp\_startsfrom; ++j) {

getline(in,str);

cout<<str<<endl;

}

for (int j = inp\_startsfrom; j <=i ; ++j) {

for (int k = 0; k < inpsize; ++k) {

getline(in,str);

}

}

for (int j = 0; j < inpsize; ++j) {

getline(in,str);

cout<<str<<endl;

}

cout<<"CORRECT OUTPUT"<<endl;

for (int j = 0; j < acsize; ++j) {

cout<<vc2[j+i]<<endl;

}

cout<<"YOURS"<<endl;

cout<<vc1[i]<<endl;

return 0;

}

}

} catch (const exception &ex) {

cerr << ex.what() << endl;

}

return 0;

}