W10\_Azhar\_Turashbayeva\_SE-2105

1001

#include <iostream>

#include <cstring>

#include <string>

using namespace std;

struct Train {

    int ID;

    char\* Location;

    char\* Time;

};

void bigletta(char\* name) {

    int arrlength = strlen(name);

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

        name[i] = toupper(name[i]);

    }

}

void train\_data(Train\* tripname, int n, char\* name) {

    bool Train = false;

    for (int i = 0; i < n; i++) {

        if (strcmp((tripname + i)->Location, name) == 0) {

            cout << (tripname + i)->ID << '\_' << (tripname + i)->Location << '\_' << (tripname + i)->Time;

            Train = true;

        }

    }

    if (Train == false) {

        cout << "Impossible" << endl;

    }

}

int main() {

    int n;

    cin >> n;

    char tripname[21];

    Train\* TRAIN = new Train[n];

    for (int i = 0; i < n; i++) {

        (TRAIN + i)->Location = new char[21]; // establishing dynamic array

        (TRAIN + i)->Time = new char[21];

        cin >> (TRAIN + i)->ID >> (TRAIN + i)->Location >> (TRAIN + i)->Time;

        bigletta((TRAIN + i)->Location); // converting letters to uppercase

    }

    cin >> tripname;

    bigletta(tripname);

    train\_data(TRAIN, n, tripname); // finding request from the list

}

1002

#include <iostream>

#include <cstring>

#include <string>

using namespace std;

void isko(char\* word) {

    for (int i = 0; i < strlen(word); i++) {

        word[i] = toupper(word[i]);

    }

}

struct Train {

    int ID;

    char\* Location = new char[1000];

    char\* Time = new char[6];

};

int main() {

    int n;

    char\* Address = new char[1000];

    cin >> n;

    Train\* one\_of\_trains = new Train[n];

    for (int i = 0; i < n; i++) {

        cin >> one\_of\_trains[i].ID >> one\_of\_trains[i].Location >> one\_of\_trains[i].Time;

        isko(one\_of\_trains[i].Location); // converting letters to uppercase

    }

    cin >> Address;

    isko(Address);

    int time\_mini = 1440, minimumdex = -1;

    for (int i = 0; i < n; i++) {

        if (strcmp(Address, one\_of\_trains[i].Location) == 0) {

            int time = (((int)one\_of\_trains[i].Time[0] - 48) \* 600) + (((int)one\_of\_trains[i].Time[1] - 48) \* 60) +

                (((int)one\_of\_trains[i].Time[3] - 48) \* 10) + ((int)one\_of\_trains[i].Time[4]); // receiving information and converting it into seconds

            if (time < time\_mini) {

                time\_mini = time;

                minimumdex = i;

            }

        }

    }

    if (minimumdex == -1) {

        cout << "Impossible" << endl;

    }

    else {

        cout << one\_of\_trains[minimumdex].ID << "\_" << one\_of\_trains[minimumdex].Location << "\_" << one\_of\_trains[minimumdex].Time << endl;

    }

}

1003

#include <iostream>

#include <cstring>

#include <string>

using namespace std;

struct Student {

    int ID;

    char\* surname;

    double grade;

};

void student\_for\_input(Student\* s, int n) {

    for (Student\* it = s; it != s + n; it++) {

        it->surname = new char[100];

        cin >> it->ID >> it->surname >> it->grade;

    }

}

void student\_sort(Student\* s, int n) {

    for (int i = 0; i < n; i++) {

        for (int j = 0; j < n; j++) {

            if ((s + i)->grade > s[j].grade) {

                swap(s[i], \*(s + j));

            }

            if ((s + i)->grade == s[j].grade) {

                if ((s + i)->ID < s[j].ID) {

                    swap(s[i], \*(s + j));

                }

            }

        }

    }

}

void succesfull\_student(Student\* s, int n) {

    cout << s->ID << " " << s->surname << " " << s->grade << endl;

}

int main() {

    int n;

    cin >> n;

    Student\* students = new Student[n];

    student\_for\_input(students, n);

    student\_sort(students, n);

    succesfull\_student(students, n);

}

1004

#include <iostream>

#include <cstring>

#include <string>

using namespace std;

struct Student {

    int ID;

    double grade;

};

void input\_of\_students(Student\* s, int n) {

    for (Student\* it = s; it != s + n; it++) {

        cin >> it->ID >> it->grade;

    }

}

void student\_sort(Student\* s, int n) {

    for (int i = 0; i < n; i++) {

        for (int j = 0; j < n; j++) {

            if ((s + i)->grade > s[j].grade)

                swap(s[i], \*(s + j));

            if ((s + i)->grade == s[j].grade) {

                if ((s + i)->ID < s[j].ID)

                    swap(s[i], \*(s + j));

            }

        }

    }

}

void succesfull\_student(Student\* s, int n) {

    for (Student\* it = s; it != s + n; it++) {

        cout << it->ID << " " << it->grade << endl;

    }

}

int main() {

    int n;

    cin >> n;

    Student\* students = new Student[n];

    input\_of\_students(students, n);

    student\_sort(students, n); // sorting in descending order

    succesfull\_student(students, n);

}

1005

#include <iostream>

#include <string>

using namespace std;

struct Luggage {

    char\* description = new char[1000];

    int count;

    double weight;

};

struct Passenger {

    char\* luggageSize = new char[1000];

    int name;

    Luggage\* luggage;

};

int main() {

    int n;

    cin >> n;

    int passangiir, bagajI;

    double max = 0;

    Passenger\* passengers = new Passenger[n];

    for (int i = 0; i < n; i++) {

        cin >> passengers[i].luggageSize >> passengers[i].name;

        passengers[i].luggage = new Luggage[passengers[i].name];

        for (int j = 0; j < passengers[i].name; j++) {

            cin >> passengers[i].luggage[j].description >> passengers[i].luggage[j].count >> passengers[i].luggage[j].weight;

            if (max < passengers[i].luggage[j].count \* passengers[i].luggage[j].weight) {

                max = passengers[i].luggage[j].count \* passengers[i].luggage[j].weight;

                passangiir = i;

                bagajI = j;

            }

        }

    }

    cout << passengers[passangiir].luggageSize << endl;

    cout << passengers[passangiir].luggage[bagajI].description << endl;

}

1006

#include <iostream>

#include <cstring>

using namespace std;

struct train {

    int id;

    char\*fraddr = new char[100];

    char\* toaddr = new char[100];

    char\* deptime = new char[100];

};

bool strong (char \* a, char \* b) {

    for (int i = 0; i <60; i ++) {

        if (a [i] <b [i]) return false;

        if (a [i]> b [i]) return true;

    }

    if (strlen (a) <strlen (b)) {

        return true;

    }

    return false;

}

int main () {

    int n; cin >> n;

    train \* tr = new train [n];

    for (int i = 0; i <n; i ++) {

        cin >> tr [i] .id >> tr [i] .fraddr >> tr [i] .toaddr >> tr [i] .deptime;

        // converting to uppercase

        for (int j = 0; j <strlen (tr [i] .fraddr); j ++) {

            tr [i] .fraddr [j] = toupper (tr [i] .fraddr [j]);

        }

        for (int j = 0; j <strlen (tr [i] .toaddr); j ++) {

            tr [i] .toaddr [j] = toupper (tr [i] .toaddr [j]);

        }

    }

    for (int i = 0; i <n - 1; i ++) {

        // sorting list

        for (int j = 0; j <n - i - 1; j ++) {

            if (strong ((tr + j) -> toaddr, (tr + j + 1) -> toaddr)) {

                swap (\* (tr + j), \* (tr + j + 1));

            }

        }

    }

    for (int i = 0; i <n; i ++) {

        cout << tr [i] .id<<' '<< tr [i] .fraddr<<' '<< tr [i] .toaddr <<' '<< tr [i] .deptime << endl;

    }

}

1007

#include <iostream>

#include <cstring>

#include <string>

using namespace std;

struct futureStudent {

    int id;

    char\* name = new char[101];

    char\* surname = new char[101];

    int subject\_1;

    int subject\_2;

    int subject\_3;

    int select\_subject;

    int special\_case;

    char\* specCas = new char[4];

};

void foo(futureStudent\* applicants, int numberApplicants) {

    for (int i = 0; i < numberApplicants - 1; i++)

    {

        for (int j = 0; j < numberApplicants - i - 1; j++)

        {

            if (applicants[j].special\_case < applicants[j + 1].special\_case)

            {

                swap(applicants[j], applicants[j + 1]);

            }

            else if (applicants[j].special\_case == applicants[j + 1].special\_case)

            {

                if (applicants[j].select\_subject < applicants[j + 1].select\_subject)

                {

                    swap(applicants[j], applicants[j + 1]);

                }

            }

        }

    }

}

int main() {

    int app\_num, hold\_num;

    cin >> app\_num >> hold\_num;

    futureStudent\* applicant = new futureStudent[app\_num];

    futureStudent\* holder = new futureStudent[hold\_num];

    int grant = 0;

    for (int i = 0; i < app\_num; i++) {

        cin >> applicant[i].id;

        cin >> applicant[i].name;

        cin >> applicant[i].surname;

        cin >> applicant[i].subject\_1;

        cin >> applicant[i].subject\_2;

        cin >> applicant[i].subject\_3;

        cin >> applicant[i].select\_subject;

        cin >> applicant[i].specCas;

        applicant[i].special\_case = applicant[i].subject\_1 + applicant[i].subject\_2 + applicant[i].subject\_3 + applicant[i].select\_subject;

        // check whether specific applicant has special case

        if (applicant[i].specCas[0] == 'Y') {

            holder[grant] = applicant[i];

            grant++;

        }

    }

    foo(applicant, app\_num); // sorting applicants by their total score

    int count = 0;

    while (grant < hold\_num) {

        if (applicant[count].specCas[0] == 'N') {

            holder[grant] = applicant[count];

            grant++;

        }

        count++;

    }

    foo(holder, hold\_num);

    for (int i = 0; i < hold\_num; i++) {

        cout << holder[i].id << " ";

        cout << holder[i].name << " ";

        cout << holder[i].surname << " ";

        cout << holder[i].special\_case << endl;

    }

}

1010

#include <iostream>

#include <string>

using namespace std;

struct Color {

    int f=0;

    int g=0;

    int r=0;

    int h=0;

};

int main() {

    char table[8][8];

    Color num\_color;

    for (int i = 0; i < 8; i++)

        for (int j = 0; j < 8; j++) {

            cin >> table[i][j];

            if (table[i][j] == '0') {

                num\_color.f++;

            }

            else if (table[i][j] == '1') {

                num\_color.g++;

            }

            else if (table[i][j] == '2') {

                num\_color.r++;

            }

            else if (table[i][j] == '3') {

                num\_color.h++;

            }

        }

    if (num\_color.f != 0 && num\_color.g != 0 && num\_color.r != 0 && num\_color.h != 0) {

        for (int i = 0; i < 8; i++) {

            for (int j = 0; j < 8; j++) {

                if (table[i][j] == '2') {

                    cout << table[i][j];

                }

                else {

                    cout << '-';

                }

            }

            cout << endl;

        }

        cout << endl << num\_color.f << ' ' << num\_color.g << ' ' << num\_color.r << ' ' << num\_color.h;

    }

    else {

        cout << "BAD INPUT LIST";

    }

}

1011

#include <iostream>

#include <cstring>

using namespace std;

int main() {

    char\* a = new char[100001];

    cin >> a;

    int lenght= strlen(a);

    int x = 0, y = 0;

    for (int i = 0; i < lenght; i++)

    {

        if (a[i] == '(')

            x++;

        else

            y++;

        if (y > x)

            break;

    }

    if (y == x)

        cout << "VALID" << endl;

    else

        cout << "INVALID" << endl;

    delete[] a;

}

1012

#include <iostream>

#include <string>

#include <stack>

using namespace std;

bool checker (string sort) {

stack<char>stk;

char x;

for (int i = 0; i <sort.length (); i ++) {

    if (sort [i] == '(' || sort [i] == '[' || sort [i] == '{') {

        stk.push (sort [i]);

    }

    if (stk.empty ())

        return false ;

    switch (sort [i]) {

        case ')': {

            x = stk.top ();

            stk.pop ();

            if (x == '{' || x == '[')

            return false;

            break;

        }

        case '}': {

            x = stk.top ();

            stk.pop ();

            if (x == '(' || x == '[')

            return false;

            break;

        }

        case ']': {

            x = stk.top ();

            stk.pop ();

            if (x == '(' || x == '{')

            return false;

            break;

        }

        if (stk.empty ())

            return false;

    }

}

return (stk.empty ());

}

int main () {

int a;

cin >> a;

char \* check = new char [a];

cin >> check;

if (checker (check))

    cout<< "Yes";

else

    cout << "No";

}