**\*Thư viện:**

**0.#include<bits/stdc++.h>**

**1.#include<iostream>**

**2.#include<algorithm>**

min(a, b) và max(a, b) và swap(a, b)

sort(a, b, cmp) và accumulate(l, r, x);

tính tổng các ptử từ l đến r

\*min\_element(a, a + n)

\*max\_element(a, a + n)

-find(a, a + n, key)

-find(v.begin(), v.end(), key)

-memset(a, 0, sizeof(a))

Gán tất cả ptử = 0 () mảng a

-reverse(s.begin(), s.end()): Lật ngược xâu

-set\_union (phải sx)

-set\_intersection

-set\_difference

-set\_symmetric\_ difference

VD int a[n],b[m]

vector<int> u(n + m)

auto it = set\_ union (a, a + n, b, b + m, u.begin())

u.resize(it – u.begin());

**3.#include<cmath>**

**#include<math.h>**

abs(x): gt tuyệt đối

**4.#include<vector>**

vector<int>

vector<int> v(n)

vector<int> v(n, gt khoi tạo)

v.push\_back, v.pop\_back, size(),

v.at(x): trả về ptử x () vector

v.front(): ptử đầu tiên, v.back(): ptử cuối

sort(v.begin(), v.end())

for(auto it : v)

**5.#include<pair>**

#include<utility>

pair<kieu dl, kieu dl> p

p.first, p.second

**6.#include<set>**

set<int> s các ptử k giống nhau

s.size, s.insert(x), s.find(x), s.count(x), s.erase(x)

multiset<int> các ptử có thể giống nhau

set và multiset sắp xếp có thứ tự

unordered\_set sắp xếp 0 có thứ tự

**7.#include<iomanip>**

cout << fixed << setprecision(2)

**8.#include<map>**

map<key, value> mp

mp.insert(x), mp.count(x), mp.find(x)

-In ra tần suất xuất hiện:

map<int, int> mp

int n; cin >> n;

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

int x; cin >> x;

mp[x]++; }

for(auto it : mp){cout << it.first << “ “<<it.second;}

**9.#include<fstream>**

-ifstream IP

IP.open(“DaTa.in”)

string s

while(IP >> s){ code}

IP.close()

-Đọc đến cuối file:

string s;

while(!IP.eof()){

getline(IP, s);

IP << s; }

-ifstream IP("DaTa.in", ios::in);

-ofstream OP("DaTa.out", ios::out);

**\*binary search: phải sx tăng or giảm dần**

-sort(a, a + n, greater<kieu dl>())

-sort(a, a + n, less<kieu dl>())

-binary\_search(a + x, a + y, key)

=> a[x] -> a[y - 1]

VD: binary\_search(a, a + n, x)

-binary\_search(v.begin(), v.end(), key)

-binary\_search(v.begin()+x, v.end()+y, key)

=> v[x] -> v[y - 1]

-tìm vtri đầu tiên của mảng đã sắp xếp:

int vt dau(int a[], int n, int x){

int res = -1;

while(l <= r){

int m = (l + r)/2;

if(a[m] == x){

res = m; r = m – 1;

// l = m + 1 de tim vtri cuoi

}

else if(a[m] < x) l = m + 1;

else r = m – 1;

}

return res; }

**\*lower\_bound(iter1, iter2, key)**

Áp dụng khi đã sắp xếp, [iter1, iter2)

trả về vtri của ptử đầu tiên >= key

-auto it = lower\_bound(a.begin(), a.end() , x)

-auto it = lower\_bound(a, a + n, x)

cout << \*it << it – a(chỉ số)

**\*upper\_bound(iter1, iter2, key)**

Áp dụng khi đã sắp xếp, [iter1, iter2)

trả về vtri của ptử đầu tiên > key

-auto it = upper\_bound(a.begin(), a.end() , x)

-auto it = upper\_bound(a, a + n, x)

cout << \*it << it – a(chỉ số)

**\*kt số nguyên tố**

bool checksnt(long long n){

if(n <= 3) return n > 1;

else if(n % 2 == 0 || n % 3 == 0){

return false;

}int i = 5;

while(i \* i <= n){

if(n % i == 0 || n % (i + 2) == 0){

return false;

}

i += 6;

} return true;

}

**\*sàng nguyên tố:**

const int maxn = 1e7;

int prime[maxn + 1];

void sang(){

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

prime[i] = 1;}

prime[0] = prime[1] = 0;

for(int i = 2; i <= sqrt(maxn); i++){

if(prime[i]){

for(int j = i \* i; j <= maxn; j += i){

prime[j] = 0;}}}}

**\*fibonacci**

long long fb[100]

long long fibonacci(int n){

fb[1] = 1;

fb[2] = 1;

for(int i = 3; i <= n; i++){

fb[i] = fb[i - 1] + fb[i - 2];

}

return fb[n];

}

**\*phân tích thừa số nguyên tố:**

void pttsnt(){

int n; cin >> n

for(int i = 2; i <= sqrt(n); i++){

if(n % i == 0)

int mu = 0

while(n % i == 0){

mu++;

n /= i;

}

cout << i << " " << mu << " ";

}

if(n == 1) break;

}

if(n > 1) cout << n << " 1";

cout << endl;

}

**\*tổ hợp chập k của n**

int c[1005][1005];

// luu to hop chap j cua i

const int MOD = 1e9 + 7;

void check(){

for(int i = 0; i <= 1000; i++){

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

if(j == 0 || i == j){

c[i][j] = 1;

}else{

c[i][j] = (c[i - 1][j] + c[i - 1][j - 1]) % MOD;

}}}}

**\*liệt kê các hoán vị:**

int vs[100], a[100], n;

void HvDq(int m){

if (m > n){

for int i = 1; i <= n; i++)

cout << a[i] << “ “;

cout << endl; return;}

for (i = 1; i <= n; i++)

if (vs[i] == 0){

a[m] = i;

vs[i] = 1;

HvDq(m+1);

vs[i] = 0;}}

void LkDqHv(int a[], int n){

for (int i = 1; i <= n; i++)

vs[i] = 0;

HvDq(1);}

**\*liệt kê các tổ hợp**

int a[100], n, k;

void ThDq(int m) {

if (m > k){

for(int i = 1; i <= k; i++)

cout << a[i] << “ “;

cout << endl; return;}

for(i = a[m-1] + 1; i <= n-k+m; i++){

a[m] = i;

ThDq(m+1); }

}

void LkDqTh(int a[],int n, int k){

ThDq(1);

}

**\*liệt kê xâu nhị phân**

int x[100], n;

void XnDq(int m) {

if (m > n) {

for int i = 1; i <= n; i++)

cout << x[i];

cout << endl; return;}

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

x[m] = i;

XnDq(m+1); }

}

void LkDqXn(int x[], int n){

XnDq(1);

}

**\*Cửa sổ trượt:**

int a[n]

int n, k;

long long sum = 0;

for(int i = 0; i < k; i++) sum += a[i];

long long res = sum, idx = 0;

for(int i = k; i < n; i++){

sum = sum - a[i - k] + a[i];

if(sum > res){

res = sum;

idx = i - k + 1;}

cout << res << endl;

//in ra tong

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

cout << a[idx + j] << " ";

}

//in ra bo so co tong lon nhat

}

**\*tổng hai số nguyên lớn:**

string tong(string a, string b){

if(a.length() < b.length()){

a.insert(0, b.length() - a.length(), '0');

}

if(b.length() < a.length()){

b.insert(0, a.length() - b.length(), '0');

}

string res = "";

int nho = 0;

for(int i = a.length() - 1; i >= 0; i--){

nho = (a[i] - '0') + (b[i] - '0') + nho;

res.insert(0, 1, nho % 10 + '0');

nho /= 10;

}

if(nho > 0){

res.insert(0, 1, nho + '0');

}

return res;

}

**\*hiệu 2 số nguyên lớn:**

string hieu(string a, string b){

if(a.length() < b.length()){

a.insert(0, b.length() - a.length(), '0');

}

if(b.length() < a.length()){

b.insert(0, a.length() - b.length(), '0');

}

if(a < b){

swap(a, b);

}

string res = "";

int nho = 0;

for(int i = a.length() - 1; i >= 0; i--){

int tmp = (a[i] - '0') - (b[i] - '0') - nho;

if(tmp < 0){

nho = 1;

tmp += 10;

}else{

nho = 0;

}

res = (char)(tmp + '0') + res;

}

**\*Danh sách sinh viên thực tập**

struct ThucTap{

int stt;

string msv, ht, lop, email, dn;

};

bool cmp(ThucTap a, ThucTap b){

return a.msv < b.msv;

}

void nhap(ThucTap a[], int n){

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

a[i].stt = i + 1;

getline(cin, a[i].msv);

getline(cin, a[i].ht);

getline(cin, a[i].lop);

getline(cin, a[i].email);

getline(cin, a[i].dn);

}

}

void sapxep(ThucTap a[], int n){

sort(a, a + n, cmp);

}

int main(){

int n;

cin >> n;

ThucTap a[n];

cin.ignore();

nhap(a, n);

sapxep(a, n);

int t;

cin >> t;

while(t--){

string s;

cin >> s;

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

if(a[i].dn == s){

cout << a[i].stt << " " << a[i].msv << " " << a[i].ht << " " << a[i].lop << " " << a[i].email << " " << a[i].dn << endl;

}

}

}

return 0;}

**\*Sắp xếp theo mã sv**

class SinhVien{

private:

string msv, ht, lop, email;

public:

string getMsv(){

return this->msv;

}

friend istream& operator >> (istream& in, SinhVien &a){

scanf("\n");

getline(cin, a.msv);

getline(cin, a.ht);

cin >> a.lop >> a.email;

return in;

}

friend ostream& operator << (ostream& out, SinhVien a){

cout << a.msv << " " << a.ht << " " << a.lop << " " << a.email << endl;

return out;

}

};

bool cmp(SinhVien a, SinhVien b){

return a.getMsv() < b.getMsv();

}

int main(){

SinhVien a[1000];

int n = 0;

while(cin >> a[n]) n++;

sort(a, a + n, cmp);

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

cout << a[i];

}

}

**\*liệt kê sinh viên theo ngành**

class SinhVien {

public:

string msv, ten, lop, email;

friend istream &operator >> (istream &is, SinhVien &a) {

scanf("\n");

getline(cin, a.msv);

getline(cin, a.ten);

getline(cin, a.lop);

getline(cin, a.email);

return is;

}

friend ostream &operator << (ostream &os, SinhVien a) {

cout << a.msv << " " << a.ten << " " << a.lop << " " << a.email << "\n";

return os;

}

};

string solve(string s) {

if (s == "Ke toan") return "KT";

if (s == "Cong nghe thong tin") return "CN";

if (s == "An toan thong tin") return "AT";

if (s == "Vien thong") return "VT";

if (s == "Dien tu") return "DT";

}

int main() {

int n;

cin >> n;

SinhVien a[1000];

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

cin >> a[i];

}

int q;

cin >> q;

while (q--) {

scanf("\n");

string s;

getline(cin, s);

string major = solve(s);

transform(s.begin(), s.end(), s.begin(), ::toupper);

cout << "DANH SACH SINH VIEN NGANH " << s << ":\n";

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

if (major == "CN" || major == "AT") {

if (a[i].msv[5] == major[0] && a[i].msv[6] == major[1] && a[i].lop[0] != 'E')

cout << a[i];

}

else {

if (a[i].msv[5] == major[0] && a[i].msv[6] == major[1])

cout << a[i];

}

}

}

return 0;

}

**\*in xâu đảo ngược khi đọc dấu ‘.’**

int main(){

string s;

getline(cin, s);

vector<string> v;

string tmp;

stringstream ss(s);

while(getline(ss, tmp, '.')){

v.push\_back(tmp);

}

string res = "";

for(int i = v.size() - 1; i >= 0; i--){

res += v[i];

res += ".";

}

res.erase(res.length() - 1);

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

cout << res[i];

}

}

**10.#include<string>**

s.length(), s.find(r): kt xâu r có () xâu s 0

s.substr(x, y): lấy ra 1 xâu ở vtri x có độ dài là y

s.erase(x, n): xóa n kí tự bắt đầu tại vtri x

s.replace(x, n, str): thay kế kí tự n tại vị trí bắt đầu là x bằng xâu str

s1.compare(s2): so sánh s1 với s2

s1.swap(s2): hoán đổi 2 xâu

s1.insert(x, s2): thêm s2 vào s1 tại sau vtri x

stoi(x), stoll(x), isdigit(x), isalpha(x)

to\_string, tolower(x), toupper(x)

s.pop\_back(x): xóa ptử cuối () xâu

**\*danh sách đối tượng sv**

int i = 1;

string chuanhoa(string s){

string res = "";

string tmp;

stringstream ss(s);

while(ss >> tmp){

res += toupper(tmp[0]);

for(int i = 1; i < tmp.length(); i++){

res += tolower(tmp[i]);

}

res += " ";

}

res.erase(res.length() - 1);

return res;

}

class SinhVien{

private:

string msv, ht, lop, ns;

float gpa;

public:

SinhVien(){

msv = ht = lop = ns = "";

gpa = 0;

}

float getGpa(){

return this->gpa;

}

friend istream& operator >> (istream& in, SinhVien &a){

scanf("\n");

getline(cin, a.ht);

cin >> a.lop >> a.ns >> a.gpa;

if(i < 10) a.msv = "B20DCCN00";

else a.msv = "B20DCCN0";

a.msv = a.msv + to\_string(i);

i++;

a.ht = chuanhoa(a.ht);

if(a.ns[1] == '/') a.ns = "0" + a.ns;

if(a.ns[4] == '/') a.ns.insert(3, "0");

return in;

}

friend ostream& operator << (ostream& out, SinhVien a){

cout << a.msv << " " << a.ht << " " << a.lop << " " << a.ns << " " << fixed << setprecision(2) << a.gpa << endl;

return out;

}

};

bool cmp(SinhVien a, SinhVien b){

return a.getGpa() > b.getGpa();

}

void sapxep(SinhVien a[], int n){

sort(a, a + n, cmp);

}

int main(){

SinhVien ds[50];

int N, i;

cin >> N;

for(i=0;i<N;i++){

cin >> ds[i];

}

sapxep(ds, N);

for(i=0;i<N;i++){

cout << ds[i];

}

return 0;

}

**\*sắp xếp già nhất trẻ nhất**

bool cmp(NhanVien a, NhanVien b){

string s1 = a.getNs(), s2 = b.getNs();

int th1 = (s1[0] - '0') \* 10 + (s1[1] - '0'), ng1 = (s1[3] - '0') \* 10 + (s1[4] - '0'), na1 = stoi(s1.substr(6));

int th2 = (s2[0] - '0') \* 10 + (s2[1] - '0'), ng2 = (s2[3] - '0') \* 10 + (s2[4] - '0'), na2 = stoi(s2.substr(6));

if(na1 != na2) return na1 < na2;

if(th1 != th2) return th1 < th2;

return ng1 < ng2;

}