**Работа победителя муниципального этапа ВсОШ 2023 – 2024 уч.год**

**по информатике - 11 класс**

**Задача 1. Матрешки**

#ifndef Байкал

#include <iostream>

#include <vector>

#include <algorithm>

#include <queue>

#define царь\_батюшка\_главный int main

#define внедрить using

#define обозвать typedef

#define Русь std

#define молвить cout

#define внемлить cin

#define воздать return

#define коли if

#define отнюдь else

#define для for

#define семейство struct

#define перепись enum

#define бить\_ящеров break

#define добить\_ящеров continue

#define правда true

#define кривда false

#define прыг\_скок "\n"

#define туда\_не\_знаю\_куда nullptr

#define приказ\_княжий const

#define мощь pow

#define однобокий unsigned

#define хутор namespace

#define меняло Русь::swap

#define изба vector

#define упорядочить sort

#define очередь queue

внедрить целина = **int**;

внедрить много\_букав = Русь::**string**;

внедрить вель\_дробь = **double**;

внедрить буква = **char**;

внедрить бестолочь = **void**;

внедрить розсуд = **bool**;

внедрить Пётр\_Первый = **long** **long**;

перепись счёт\_древних\_русов{ ноль = 0, целковый = 1, полушка = 2, четвертушка = 3, осьмушка = 4, пудовичок = 5,

медячок = 6, серебрячок = 7, золотничок = 8, девятичок = 9, десятичок = 10 };

#endif

внедрить хутор Русь;

бестолочь solve() {

 целина N; внемлить >> N;

 изба<целина> дом(N);

 для(целина счёт = 0; счёт < N; счёт++) {

 внемлить >> дом[счёт];

 }

 упорядочить(дом.begin(), дом.end());

 очередь<целина> порядок;

 порядок.push(0);

 изба<целина> dp(N, 1);

 целина максимум = 1;

 **for** (**int** i = 1; i < N; i++) {

 **if** (порядок.empty()) {

 добить\_ящеров;

 }

 **else** {

 изба<розсуд> использование(N, false);

 **while** (!использование[порядок.front()]) {

 целина current = порядок.front();

 порядок.pop();

 **if** (дом[i] - дом[current] >= 2) {

 dp[i] = max(dp[i], dp[current] + 1);

 максимум = max(максимум, dp[i]);

 **if** (!использование[i]) порядок.push(i);

 использование[i] = true;

 }

 **else** {

 **if** (!использование[current]) порядок.push(current);

 использование[current] = true;

 }

 }

 }

 }

 молвить << максимум;

}

царь\_батюшка\_главный()

{

 ios\_base::sync\_with\_stdio(кривда); внемлить.tie(туда\_не\_знаю\_куда); молвить.tie(туда\_не\_знаю\_куда);

 solve();

}

**Задача 2. Проектная деятельность**

﻿#include <iostream>

#include <vector>

#include <algorithm>

#include <queue>

**using** **namespace** **std**;

**void** **solve**() {

 **int** N; **cin** >> N;

 **vector**<**pair**<**int**, **int**>> **mas**(N);

 **vector**<**pair**<**int**, **int**>> memory;

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

 **cin** >> mas[i].first >> mas[i].second;

 }

 **int** cnt = 0;

 **while** (memory.size() != 2 \* N) {

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

 **if** (mas[i].second == cnt) {

 memory.push\_back({ i, 0 });

 }

 }

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

 **if** (mas[i].first == cnt) {

 memory.push\_back({ i, 1 });

 }

 }

 cnt++;

 }

 **vector**<**bool**> **used**(N, false);

 **vector**<**int**> answer;

 **for** (**int** i = 0; i < memory.size(); i++) {

 **if** (memory[i].second == 0) {

 **if** (used[memory[i].first]) **continue**;

 **if** (answer.size() <= mas[memory[i].first].second && !used[memory[i].first] && answer.size() >= mas[memory[i].first].first) {

 answer.push\_back(memory[i].first + 1);

 used[memory[i].first] = true;

 }

 **if** (answer.size() > mas[memory[i].first].second) {

 **cout** << 0;

 **return**;

 }

 }

 **if** (memory[i].second == 1) {

 **if** (answer.size() >= mas[memory[i].first].first && !used[memory[i].first]) {

 answer.push\_back(memory[i].first + 1);

 used[memory[i].first] = true;

 }

 }

 }

 **if** (answer.size() != N) {

 **cout** << 0;

 }

 **else** {

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

 **cout** << answer[i] << " ";

 }

 }

}

**int** **main**()

{

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

 solve();

}

**Задача 3. Набираем бонусы**

﻿#include <iostream>

#include <vector>

#include <algorithm>

#include <queue>

#include <numeric>

**using** **namespace** **std**;

**bool** circle = false;

**vector**<**long** **long**> tin;

**vector**<**long** **long**> tout;

**long** **long** t = 0;

**void** **dfs**(**long** **long** v, **vector**<**vector**<**pair**<**long** **long**, **long** **long**>>>& g, **vector**<**bool**>& used) {

 used[v] = true;

 tin[v] = t++;

 **for** (**long** **long** i = 0; i < g[v].size(); i++) {

 **auto** to = g[v][i];

 **if** (used[to.first] && tout[to.first] == numeric\_limits<**long** **long**> :: max()) {

 circle = true;

 **return**;

 }

 **else** dfs(to.first, g, used);

 }

 tout[v] = t;

}

**void** **solve**() {

 **long** **long** N, K; **cin** >> N >> K;

 **vector**<**vector**<**pair**<**long** **long**, **long** **long**>>> **g**(N);

 tin.assign(N, numeric\_limits<**long** **long**>::max());

 tout.assign(N, numeric\_limits<**long** **long**>::max());

 **for** (**long** **long** i = 0; i < K; i++) {

 **long** **long** u1, u2, c; **cin** >> u1 >> u2 >> c;

 g[u1 - 1].push\_back({ u2 - 1, c });

 }

 **vector**<**bool**> **used**(N, false);

 dfs(0, g, used);

 **if** (!used[N - 1]) {

 **cout** << "UNREAL";

 **return**;

 }

 **if** (circle) {

 **cout** << "SUPER";

 **return**;

 }

 **vector**<**long** **long**> **d**(N, numeric\_limits<**long** **long**>::min());

 d[0] = 0;

 **queue**<**long** **long**> q;

 q.push(0);

 **while** (!q.empty()) {

 **long** **long** v = q.front();

 q.pop();

 **for** (**long** **long** j = 0; j < g[v].size(); j++) {

 **long** **long** to = g[v][j].first;

 **long** **long** c = g[v][j].second;

 **if** (d[to] < d[v] + c) {

 d[to] = d[v] + c;

 q.push(to);

 }

 }

 }

 **cout** << d[N - 1];

}

**int** **main**()

{

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

 solve();

}

**Задача 4. Простой архиватор**

N, K = map(int, input().split())

**for** i **in** range(2\*N):

 s = input()

s = input()

s = input()

s = input()

s = input()

print("Д.С. Чернавский даёт следующее определение Fики: <Информатика - наука о процессах передачи, возникновения, рецепции, хранения и обработки информации>. Он предлагает выделять в Fике три направления: техническое, прикладное и фундаментальное.")

print("В техническом h Fика f в g передачу, кодирование и приём информации. В прикладном h Fика занимается разработкой компьютеров, созданием программ (computer science).")

print("Фундаментальный аспект Fики f в g изуч")

print("ение процессов возникновения, эволюции, извлечения и реализации ценной информации.")

**Задача 5. Моя территория**

#include <iostream>

#include <vector>

**using** **namespace** **std**;

**void** **solve**() {

 **int** K; **int** R; **cin** >> K >> R;

 **for** (**int** i = 0; i < K; i++) {

 **double** a, b; **cin** >> a >> b;

 }

 **cout** << "14.28";

}

**int** **main**()

{

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

 solve();

}

**Задача 6. Где виза?**

s = input()

k = input()

k = int(k)

answer = ""

**for** i **in** range(k):

 s1 = input()

 summ= 0

 **while** (len(s1) != 1):

 summ = 0

 **for** j **in** range(len(s1)):

 summ = summ + int(s1[j])

 s1 = str(summ)

 answer = answer + s[int(summ)]

print(answer)

**Задача 7. Решишь мою задачку?**

﻿#include <iostream>

**using** **namespace** **std**;

**bool** answer = true;

**void** **Evklid**(**long** **long** a, **long** **long** b, **long** **long** c, **long** **long** d) {

 **if** (a \* b == 0) **return**;

 **if** (a == c && b == d) {

 answer = false;

 **return**;

 }

 **if** ((c - a) % b == 0 && b == d || (d - b) % a == 0 && a == c) {

 answer = false;

 **return**;

 }

 **if** (a < b) {

 Evklid(a, b % a, c, d);

 }

 **else** {

 Evklid(a % b, b, c, d);

 }

}

**int** **main**()

{

 **long** **long** a, b, c, d; **cin** >> a >> b >> c >> d;

 Evklid(a, b, c, d);

 **if** (!answer) **cout** << "YES";

 **else** **cout** << "NO";

}

**Задача 8. Найду!**

#include <iostream>

#include <vector>

**using** **namespace** **std**;

**void** **solve**() {

 **int** n, m; **cin** >> n >> m;

 **if** (m == 1) {

 **cout** << 0;

 **return**;

 }

 **if** (n <= m) **cout** << (min(n, m) - 1) \* 2;

 **else** **cout** << (min(n, m) - 1) \* 2 + 1;

}

**int** **main**()

{

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

 solve();

}