Заключительный этап

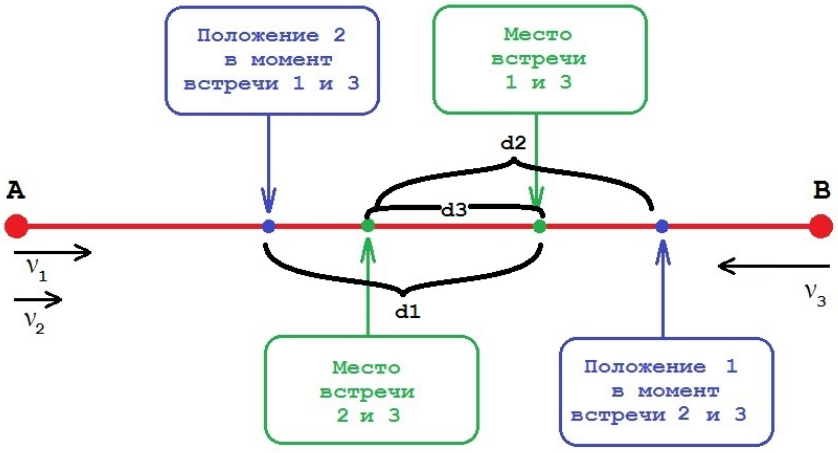
Индивидуальный предметный тур

Информатика. 8–11 класс

Задача III.1.1.1. Интеллектуальная ГИС (12 баллов)

Вы являетесь тестером в команде разработчиков новой интеллектуальной геоинформационной системы. Данная система помимо прямых расчетов расположения отслеживаемых объектов, должна уметь определять необходимые данные (расстояния, координаты и т. п.) путем сопоставления множества косвенных данных.

В данный момент тестируется следующая ситуация: есть три объекта 1, 2 и 3. Объекты 1 и 2 одновременно стартуют из точки A в направлении точки B, объект 3 в этот же момент стартует из точки B в направлении точки A. Известно, что скорость объекта 1 (обозначим ее v_1) строго больше скорости объекта 2 (обозначим ее v_2), но сами значения этих скоростей не известны. Однако известна их разность $dv = v_1 - v_2$. Очевидно, что при своем движении, объект 3 сначала встретится с объектом 1, а затем через какое-то время встретится с объектом 2. Известны следующие расстояния: d_1 — расстояние между 1 и 2 в момент встречи третьего и первого, d_2 — расстояние между 1 и 2 в момент встречи третьего и второго, d_3 — расстояние между точками встречи. Требуется по заданным четырем параметрам определить расстояние между точками A и B. В момент встречи второго и третьего первый не успевает достичь точки B.



Формат входных данных

На вход подаются четыре положительных числа dv, d_1, d_2, d_3 через пробел. Все числа целые в пределах от 1 до 10^9. $d_3 < d_1 < d_2$. Входные данные таковы, что в момент встречи второго и третьего первый еще не успевает достичь точки A.

Формат выходных данных

Вывести одно число — расстояние между точками AB. Ответ следует выводить с округлением ровно до пяти знаков после десятичной точки.

Примеры

Пример №1

Стандартный ввод
3 9 12 5
Стандартный вывод
36.00000

Пример программы-решения

Ниже представлено решение на языке C++.

```cpp
#include <bits/stdc++.h>
#define int long long
using namespace std;

int main(){
    long double dv, d1, d2, d3;
    cin >> dv >> d1 >> d2 >> d3;

    cout << fixed << setprecision(5) << d1 * d2 / (d2 - d1);
    return 0;
}
```

Пример программы-решения

Ниже представлено решение на языке Python.

```python
rom decimal import Decimal
a = list(map(int, input().split()))
d1 = Decimal(str(a[1]))
d2 = Decimal(a[2])
n = 5
a = d1 * d2 / (d2 - d1)
print(f'{a:.{n}f}')
```

Задача III.1.1.2. Самоорганизация в городе (15 баллов)

Рассмотрим следующую модель обмена ресурсами внутри некоторого города. Город состоит из n расположенных в один ряд кварталов. Каждый квартал изначально имеет a_i единиц запаса некоторого ресурса, причем никакие два квартала не обладают одинаковыми запасами. Для любых двух рядом расположенных кварталов возможна операция передачи одной единицы ресурса от большего по запасам к меньшему, но, само собой, отдающий квартал не должен стать беднее по запасам того,

кому он отдает. Основным требованием властей города является разнообразие запасов, что означает что ни в какой момент времени никакие два квартала не должны обладать одинаковыми по величине запасами. Таким образом, операция передачи может быть осуществлена только тогда, когда после ее выполнения новые величины запасов обоих участвовавших в операции передачи ресурса кварталов не совпадают ни с одним другим кварталом (в том числе и между собой). Если есть несколько пар рядом стоящих кварталов, имеющих возможность передачи, то выбирается пара с наибольшей разностью, а если и таких несколько, то среди них выбирается пара с минимальным значением запаса у получающего помощь квартала. За одну единицу времени возможен ровно один обмен. Такие операции поочередно производятся до тех пор, пока это не нарушает правила разнообразия. Требуется выяснить, каким станет распределение ресурса после окончания обменов.

Формат входных данных

В первой строке содержится число n — количество кварталов. Во второй строке находятся n натуральных чисел a_i через пробел — запасы ресурса у соответствующего квартала. Все величины запасов попрано различны. Все числа на входе в пределах от 1 до 1000.

Формат выходных данных

Вывести в одну строку через пробел n чисел — запасы в кварталах после того, как станет невозможно производить операции передачи ресурса.

Примеры

Пример №1

Стандартный ввод
6 2 10 9 3 6 12

Стандартный вывод
3 9 8 5 7 10

Пояснение к примеру 1

Приведем последовательность перераспределений:
0) 2 10 **9 3** 6 12
1) **2 10** 8 4 6 12
2) 3 9 8 4 **6 12**
3) 3 9 8 **4 7** 11
4) 3 9 8 5 **6 11**
5) 3 9 8 5 7 10

В нулевой строке исходные значения. Самая большая разница между 10 и 2, но эту операцию запрещено проводить, так как тогда во втором квартале будет 9, но 9 уже есть в третьем. Следующая по величине разница между 9 и 3, а так же между 12 и 6. Тогда производится помощь наиболее нуждающемуся в этих двух вариантах,

а именно $9 \to 3$. В первой строке результат передачи. Теперь самая большая разница между 2 и 10 и ничто не мешает произвести передачу $10 \to 2$. Во второй строке результат. Теперь снова есть два варианта $9 \to 3$ и $12 \to 6$, но теперь первый запрещен по причине того, что после этого получатся числа 4 и 8, но эти числа уже во второй строке есть. Тогда выполняется второй вариант $12 \to 6$, получается третья строка. По прежнему нельзя $9 \to 3$, но и $11 \to 7$ нельзя, так как тогда будет повторение 8. Следующей по разности парой будет $7 \to 4$, которая и выполнит операцию. Получим четвертую строку. $3 \to 9$ снова запрещено, но теперь можно $11 \to 6$, и получим пятую строку. И теперь ни одной операции не получится произвести, то есть эта строка и будет ответом.

Пример программы-решения

Ниже представлено решение на языке C++.

```
#include <bits/stdc++.h>

using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef long double ld;

vector<int> mask;

bool ok_turn(int a, int b){
    if(a > b)
        swap(a, b);

    if(mask[a+1] == 0 && mask[b-1] == 0 && a+2 != b)
        return 1;

    return 0;
}

int main(){
    ios::sync_with_stdio(0), cin.tie(0), cout.tie(0);

    int n;
    cin >> n;
    vector<int> v(n);
    mask.resize(1001, 0);
    for(int i = 0; i < n; i++){
        cin >> v[i];
        mask[v[i]] = 1;
    }

    while(1){
    int big = -1, small = -1;

    for(int i = 1; i < n; i++)
        if(ok_turn(v[i-1], v[i])){
                int tbig = i-1;
                int tsmall = i;
                if(v[tbig] < v[tsmall])
                    swap(tbig, tsmall);
            if(big == -1){
                big = tbig;
```

```
                small = tsmall;
            }
            else{
                if(v[big] - v[small] < v[tbig] - v[tsmall] ||
                    (v[big] - v[small] == v[tbig] - v[tsmall] && v[tsmall] < v[small])){
                    big = tbig;
                    small = tsmall;
                }
            }
        }

        if(big == -1) break;
        mask[v[big]] = 0;
        v[big]--;
        mask[v[big]] = 1;
        mask[v[small]] = 0;
        v[small]++;
        mask[v[small]] = 1;
    }

    for(auto q : v) cout<<q<<' ';
    cout<<endl;
    return 0;
}
```

Задача III.1.1.3. Проектирование электросети (15 баллов)

Рассмотрим проект модели снабжения мегаполиса электроэнергией. Имеется n источников, генерирующих электроэнергию для нужд мегаполиса и уже запланированная структура линий электропередач, соединяющая эти источники с мегаполисом. Совокупно источники и мегаполис будем называть объектами. Для каждого источника известно, какую мощность он генерирует, кроме того известно, какие объекты будут соединены между собой передающими линиями. Известно, что вся сеть имеет древовидную структуру, то есть она связная и не имеет в своем составе циклов. Считаем, что мегаполис потребит всю мощность, которую получится доставить. Потери мощности при передаче считаем равными нулю.

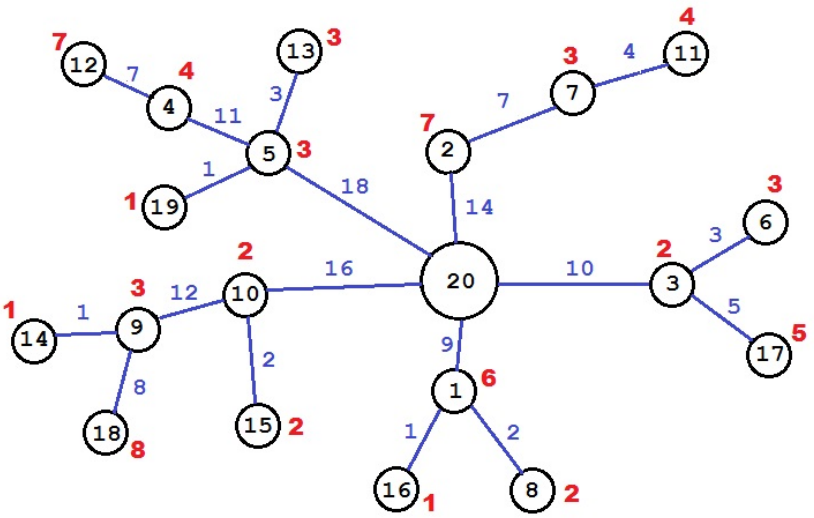
Требуется рассчитать пропускную способность каждой линии в проектируемой сети, то есть наибольшую мощность, которую можно передать по этой линии. При этом нужно указать минимально необходимую пропускную способность, такую, чтобы все имеющиеся генерируемые мощности были доставлены до мегаполиса.

Формат входных данных

В первой строке содержится число n — количество источников электроэнергии. Источники пронумерованы от 1 до n. Мегаполис имеет номер $n+1$. Далее в n следующих строках содержится по два числа a_i, b_j — обозначающих, что объекты номер a_i и b_j по проекту будут соединены линией. В последней строке содержатся n чисел gp_i через пробел, описывающих генерируемую мощность источника номер i. Все числа на входе в пределах от 1 до 1000. Гарантируется, что заданный граф является деревом, то есть связен и не содержит циклов.

Формат выходных данных

Для каждой линии в соответствующей ей отдельной строке указать необходимую минимальную пропускную способность. Порядок линий на выходе должен совпадать с порядком линий на входе.



Примеры

Пример №1

Стандартный ввод
19 12 4 19 5 5 13 4 5 5 20 2 20 20 10 7 2 10 15 10 9 18 9 14 9 1 20 1 8 16 1 20 3 3 6 17 3 7 11 6 7 2 4 3 3 3 2 3 2 4 7 3 1 2 1 5 8 1

Стандартный вывод
7 1 3 11 18 14 16 7 2 12 8 1 9 2 1 10 3 5 4

Пояснение к примеру

На рисунке приведена схема для первого примера. Красным обозначены генерируемые мощности, синим — необходимые минимальные пропускные способности линий. В частности, видно, что восточный блок источников 3, 6 и 17 генерирует суммарную мощность равную 10, а юго-западный подблок 9, 14, 18 требует для себя линии с пропускной способностью 12, и т. д.

Пример программы-решения

Ниже представлено решение на языке C++.

```
#include <bits/stdc++.h>

using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef long double ld;

int n, a, b;
vector<vector<pii> > G;
vector<int> gp, ans;

void dfs(int a, int p){
    for(auto q : G[a])
        if(q.first != p){
            dfs(q.first, a);
            ans[q.second] = gp[q.first];
            gp[a] += gp[q.first];
        }
}

int main(){
```

```
    ios::sync_with_stdio(0), cin.tie(0), cout.tie(0);

    cin >> n;
    n++;
    G.resize(n+2);
    gp.resize(n+1);
    ans.resize(n-1);
    for(int i = 0; i < n-1; i++){
        cin >> a >> b;
        G[a].push_back({b, i});
        G[b].push_back({a, i});
    }

    for(int i = 1; i < n; i++)
        cin >> gp[i];

    dfs(n, n);

    for(int i = 0; i < ans.size(); i++)
        cout << ans[i] <<endl;

    return 0;
}
```

Пример программы-решения

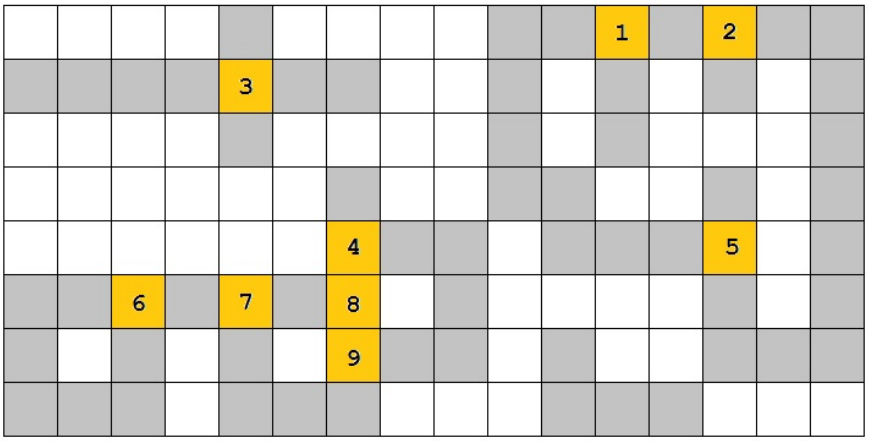
Ниже представлено решение на языке Python.

```
import sys
def dfs(u, par):
    global getId, a, dp, ans
    for elem in a[u]:
        if elem != par:
            dfs(elem, u)
            dp[u] += dp[elem]
            ans[getId[u][elem]] = dp[elem]
sys.setrecursionlimit(100000)
n = int(input())
n += 1
a = []
getId = []
ans = [0] * (n + 1)
dp = [0] * (n + 1)
for i in range(n + 1):
    a.append([])
    add = []
    for j in range(n + 1):
        add.append(-1)
    getId.append(add)
for i in range(n - 1):
    sd = input().split()
    u = int(sd[0])
    v = int(sd[1])
    a[u].append(v)
    a[v].append(u)
    getId[u][v] = i + 1
    getId[v][u] = i + 1
oo = list(map(int, input().split()))
for i in range(len(oo)):
```

```
32      dp[i + 1] = oo[i]
33  dfs(n, -1)
34  for i in range(1, n):
35      print(ans[i])
```

Задача III.1.1.4. Матрица кратчайших расстояний (25 баллов)

Рассмотрим уже известную модель дорожной сети, получаемую при анализе снимка местности. Изображение состоит только из точек двух типов. Точка, отмеченная 1, изображает дорожное полотно, все остальное изображается точками 0. При этом ширина дороги всегда ровно 1 точка. Нигде на изображении нет квадрата 2×2, состоящего из точек 1. Будем считать, что в точках, являющихся пересечением трех или четырех дорог находятся населенные пункты. Пронумеруем эти населенные пункты от 1 до K, сверху вниз и слева направо (см. пример, $K = 9$).



В данном случае нужно решить классическую задачу теории графов для не самого классического способа задания этих графов — при помощи упрощенного рисунка. А именно, нужно для каждой пары населенных пунктов определить кратчайшее расстояние между ними при движении по дорогам, или выяснить, что по дорогам попасть из одного в другой не получится. Будем считать, что перемещаться можно только из одной клетки помеченной 1 в соседнюю по стороне, так же помеченную 1, при этом длина такого перемещения равна 1.

Формат входных данных

Первая строка содержит два числа N и M через пробел — размер изображения. Далее в N строках содержится по M символов 0 или 1 без пробелов между ними, соответствующих изображению. Нигде на изображении нет квадрата 2×2, состоящего из точек 1. N и M находятся в пределах от 5 до 50. Гарантируется, что есть хотя бы один населенный пункт.

Формат выходных данных

Вывести K строк, по K чисел в каждой — матрицу кратчайших расстояний. В i-й строке на j-й позиции должно находиться кратчайшее расстояние между пунктами номер i и номер j. Обратите внимание на формат вывода: каждое расстояние

требуется выводить в формате пять знаков, недостающие символы нужно заменять пробелами. Если из одного пункта попасть в другой нельзя, нужно выводить в таком случае -1.

Примеры

Пример №1

Стандартный ввод
8 16 0000100001111111 1111111001010101 0000100001010001 0000001001100101 0000001110111101 1111111010000101 1010101110100111 1110111000111000

Стандартный вывод
0 2 -1 -1 10 -1 -1 -1 -1 2 0 -1 -1 12 -1 -1 -1 -1 -1 -1 0 -1 -1 -1 -1 -1 -1 -1 -1 -1 0 -1 5 3 1 2 10 12 -1 -1 0 -1 -1 -1 -1 -1 -1 -1 5 -1 0 2 4 5 -1 -1 -1 3 -1 2 0 2 3 -1 -1 -1 1 -1 4 2 0 1 -1 -1 -1 2 -1 5 3 1 0

Пример программы-решения

Ниже представлено решение на языке C++.

```
#include <bits/stdc++.h>

using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef long double ld;

int n, m;
vector<string> S;
vector<vector<int> > M, H;
int dx[4] = {0, -1, 0, 1};
int dy[4] = {-1, 0, 1, 0};

bool intr(int x, int y){
    return (x >= 0 && y >= 0 && x < n && y < m);
}

bool np(int x, int y){
    int r = 0;
    for(int i = 0; i < 4; i++){
        int nx = x + dx[i];
```

```
22          int ny = y + dy[i];
23          if(intr(nx, ny))
24              r += S[nx][ny] - '0';
25      }
26
27      return (r >= 3 && S[x][y] == '1');
28  }
29
30  void dfs(int x, int y, int d, int p){
31      if(M[x][y] == 0)
32          M[x][y] = 1;
33      for(int i = 0; i < 4; i++){
34          int nx = x + dx[i];
35          int ny = y + dy[i];
36
37          if(intr(nx, ny) && S[nx][ny] == '1'){
38              if(M[nx][ny] == 0)
39                  dfs(nx, ny, d+1, p);
40              else
41                  if(M[nx][ny] < 0 && -M[nx][ny] != p){
42                     int np = -M[nx][ny];
43                      H[np][p] = min(H[np][p], d+1);
44                      H[p][np] = min(H[p][np], d+1);
45                  }
46          }
47      }
48  }
49
50  int main(){
51      ios::sync_with_stdio(0), cin.tie(0), cout.tie(0);
52
53      cin >> n >> m;
54      S.resize(n);
55      for(int i = 0; i < n; i++)
56          cin >> S[i];
57
58      M.resize(n, vector<int> (m, 0));
59
60      int K = 0;
61      for(int i = 0; i < n; i++)
62          for(int j = 0; j < m; j++)
63          if(np(i, j)){
64              K--;
65              M[i][j] = K;
66          }
67
68          H.resize(-K+1, vector<int>(-K+1, 1e9));
69          for(int i = 1; i <= -K; i++)
70              H[i][i] = 0;
71
72      for(int i = 0; i < n; i++)
73          for(int j = 0; j < m; j++)
74              if(M[i][j] < 0)
75                  dfs(i, j, 0, -M[i][j]);
76
77
78      for (int k = 1; k <= -K; k++)
79          for (int i = 1; i <= -K; i++)
80              for (int j = 1; j <= -K; j++)
81                  H[i][j] = min (H[i][j], H[i][k] + H[k][j]);
```

```
82
83          for(int i = 1; i <= -K; i++)
84              for(int j = 1; j <= -K; j++)
85                          if(H[i][j] == 1e9)
86                              H[i][j] = -1;
87
88          for(int i = 1; i <= -K; i++){
89              for(int j = 1; j <= -K; j++)
90              cout<<setw(5)<<H[i][j];
91              cout<<endl;
92          }
93      return 0;
94  }
```

Пример программы-решения

Ниже представлено решение на языке Python.

```
1   import queue
2   dx = [-1, 0, 1, 0]
3   dy = [0, 1, 0, -1]
4   n, m = list(map(int, input().split()))
5   a = [['0'] * (m + 2) for i in range(n + 2)]
6   check = [[-1] * (m + 2) for i in range(n + 2)]
7   for i in range(1, n + 1):
8       om = input()
9       for j in range(m):
10          a[i][j + 1] = om[j]
11  yk = 1
12  ff = []
13  for i in range(1, n + 1):
14      for j in range(1, m + 1):
15          if a[i][j] == '1':
16              cc = ord(a[i - 1][j]) + ord(a[i + 1][j]) + ord(a[i][j - 1]) + ord(a[i][j +
                ↪  1]) - ord('0') - ord('0') - ord('0') - ord('0')
17              if cc >= 3:
18                  check[i][j] = yk
19                  ff.append([i, j])
20                  yk += 1
21  for k in range(len(ff)):
22      Q = queue.Queue()
23      Q.put(ff[k])
24      dist = [[1000000000] * (m + 2) for i in range(n + 2)]
25      dist[ff[k][0]][ff[k][1]] = 0
26      while Q.empty() == False:
27          u = Q.get()
28          for i in range(4):
29              if (u[0] + dy[i] >= 1 and u[0] + dy[i] <= n and u[1] + dx[i] >= 1 and u[1]
                ↪  + dx[i] <= m and a[u[0] + dy[i]][u[1] + dx[i]] == '1' and dist[u[0] +
                ↪  dy[i]][u[1] + dx[i]] > dist[u[0]][u[1]] + 1):
30                      dist[u[0] + dy[i]][u[1] + dx[i]] = dist[u[0]][u[1]] + 1
31                      Q.put([u[0] + dy[i], u[1] + dx[i]])
32      for i in range(len(ff)):
33          if dist[ff[i][0]][ff[i][1]] == 1000000000:
34              st = repr(-1).rjust(5)
35              print(st,end='',flush = True)
36          else:
37              st=repr(dist[ff[i][0]][ff[i][1]]).rjust(5)
```

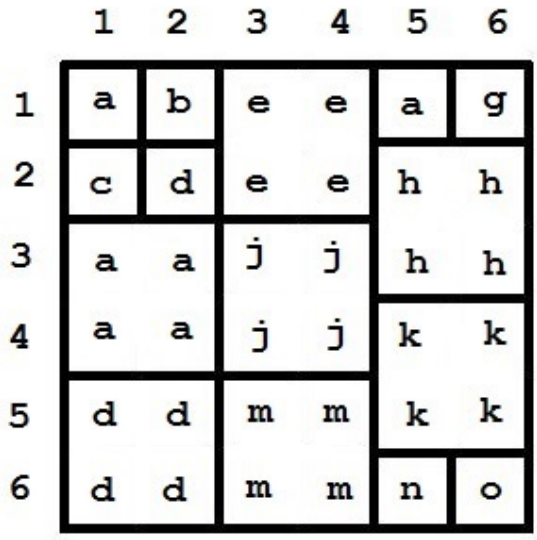
```
38              print(st,end='',flush = True)
39      print()
```

Задача III.1.1.5. Агломерация (33 баллов)

Агломерация — это «компактное скопление населенных пунктов, главным образом городов, местами срастающихся, объединенных в сложную многокомпонентную динамическую систему с интенсивными производственными, транспортными и культурными связями». Изначально отдельно образовавшиеся поселения со временем разрастаются и сливаются в одно многоуровневое образование — городскую агломерацию. При этом процесс слияния может иметь рекурсивную структуру, когда несколько агломераций сливаются в одну агломерацию более высокого уровня.

В этой задаче нужно построить обработку снимка городской агломерации, находящую ее уровень. Идея очевидна — если в агломерацию входит как часть агломерация уровня $n-1$ и нет частей-агломераций уровня n, то сама эта рассматриваемая агломерация имеет уровень n.

Но не все так просто — в данной задаче уточним понятие агломерации следующим образом: под агломерацией будем понимать любую квадратную область плоскости, целиком занятую отдельными поселениями, причем любое поселение, имеющееся в этом квадрате, не должно выходить за его пределы. Любой отдельный населенный пункт назовем агломерацией первого уровня. На снимке такой населенный пункт имеет вид квадрата, все ячейки которого обозначаются одним и тем же символом. Из низкоуровневых агломераций путем слияния получаются более высокоуровневые. Для заданного снимка квадратной области, целиком заполненной населенными кварталами, требуется определить его уровень.



Рассмотрим пример. Каждый единичный квадрат обозначен своим символом и является жилым кварталом. Если несколько рядом стоящих кварталов обозначены одним символом, они образуют населенный пункт, целиком занимающий квадратную область. Каждый такой населенный пункт образует агломерацию первого уровня. Всего таких на рисунке 15. Агломераций второго уровня на рисунке две, укажем их левый верхний и правый нижний углы: (1, 1) —- (2, 2) и (3, 1) —- (6, 4). Агломераций

третьего уровня одна, ее координаты (1, 1) — (4, 4). Весь квадрат в примере является агломерацией уровня четыре.

Отдельно следует отметить, что разные населенные пункты могут быть обозначены одним символом, при условии, что они не имеют общую границу ненулевой длины.

Формат входных данных

В первой строке находится число n — размер стороны квадрата заданной агломерации ($1 \leqslant n \leqslant 100$). В следующих n строках содержится по n маленьких букв из множества a ... z без пробелов, описывающих населенные пункты в составе агломерации. Буквы одного вида, являющиеся соседними по стороне всегда образуют квадраты. Некоторые пары различных квадратов могут быть обозначены одинаковыми буквами, но при этом данные пары квадратов не имеют общих границ ненулевой длины.

Формат выходных данных

Вывести одно число — уровень представленной во входных данных агломерации.

Примеры

Пример №1

Стандартный ввод
6 abeeag cdeehh aajjhh aajjkk ddmmkk ddmmno
Стандартный вывод
4

Пример программы-решения

Ниже представлено решение на языке C++.

```
#include <bits/stdc++.h>

using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef long double ld;

int n;
vector<vector<int> > u, d, l ,r;

bool sq(int rx, int ry, int D){
```

```
12      int lx = rx - D + 1;
13      int ly = ry - D + 1;
14      if(r[lx][ly] >= D && d[lx][ly] >= D && u[rx][ry] >= D && l[rx][ry] >= D)
15          return 1;
16      return 0;
17  }
18
19  int main(){
20      ios::sync_with_stdio(0), cin.tie(0), cout.tie(0);
21
22      string s;
23      cin >> n;
24      vector<string> v(n+2, string(n+2, '#'));
25      for(int i = 1; i <= n; i++){
26          cin >> s;
27          s = "#" + s + "#";
28          v[i] = s;
29      }
30
31      u.resize(n+2, vector<int>(n+2, 0));
32      for(int i = 1; i <= n; i++)
33      for(int j = 1; j <= n; j++){
34          u[i][j] = (v[i][j] != v[i][j+1]);
35          u[i][j] += u[i-1][j] * (u[i][j] != 0);
36      }
37
38      l.resize(n+2, vector<int>(n+2, 0));
39      for(int j = 1; j <= n; j++)
40          for(int i = 1; i <= n; i++){
41              l[i][j] = (v[i][j] != v[i+1][j]);
42              l[i][j] += l[i][j-1] * (l[i][j] != 0);
43          }
44
45      d.resize(n+2, vector<int>(n+2, 0));
46      for(int i = n; i >= 1; i--)
47          for(int j = n; j >= 1; j--){
48              d[i][j] = (v[i][j] != v[i][j-1]);
49              d[i][j] += d[i+1][j] * (d[i][j] != 0);
50          }
51
52      r.resize(n+2, vector<int>(n+2, 0));
53      for(int j = n; j >= 1; j--)
54          for(int i = n; i >= 1; i--){
55              r[i][j] = (v[i][j] != v[i-1][j]);
56              r[i][j] += r[i][j+1] * (r[i][j] != 0);
57          }
58
59      vector<vector<int> > dp(n+2, vector<int>(n+2, 0));
60
61      for(int u = 1; u <= n; u++)
62          for(int i = n; i >= 1; i--)
63              for(int j = n; j >= 1; j--)
64                  if(i - u + 1 >= 1 && j - u + 1 >= 1){
65                      dp[i][j] = max({dp[i][j], dp[i][j-1], dp[i-1][j], dp[i-1][j-1]});
66                      if(sq(i, j, u)) dp[i][j]++;
67                  }
68
69      cout << dp[n][n]<<endl;
70
71      return 0;
```

```
}
```

Пример программы-решения

Ниже представлено решение на языке Python.

```python
n = int(input())
a = [['#'] * (n + 2) for i in range(n + 2)]
for i in range(1, n + 1):
    sd = input()
    for j in range(1, n + 1):
        a[i][j] = sd[j - 1]
ans = []
k = [[0] * (n + 2) for i in range(n + 2)]
for i in range(n + 2):
    ans.append(k)
for sz in range(1, n + 1):
    for i in range(1, n - sz + 2):
        for j in range(1, n - sz + 2):
            y1 = i
            x1 = j
            x2 = j + sz - 1
            y2 = i + sz - 1
            xx1 = x1
            xx2 = x2
            ch = 1
            while x1 <= x2:
                if (a[y1 - 1][x1] == a[y1][x1] or a[y2][x1] == a[y2 + 1][x1]):
                    ch = 0
                    break
                x1 += 1

            while y1 <= y2:
                if (a[y1][xx1] == a[y1][xx1 - 1] or a[y1][xx2] == a[y1][xx2 + 1]):
                    ch = 0
                    break
                y1 += 1;
            ans[sz][i][j] =ch + max(ans[sz - 1][i][j], max(ans[sz - 1][i][j + 1],
            ↪  max(ans[sz - 1][i + 1][j], ans[sz - 1][i + 1][j + 1])))
print(ans[n][1][1])
```

Физика. 8-9 класс

Задача III.1.2.1. Лед в воде (20 баллов)

Высокий стакан имеет форму цилиндра с площадью основания 10 см^2. В нем находится холодная вода при температуре 0 °C. В стакан помещают кусок льда массой 15 г и температурой (-10 °C) и наливают теплую воду массой 10 г и температурой (40 °C). Найти увеличение высоты воды в стакане. Плотность воды 1000 кг/м^3, плотность льда 920 кг/м^3. Удельная теплоемкость воды 4200 Дж/(кг·°C), удельная теплоемкость льда 2100 Дж/(кг·°C), удельная теплота плавления льда 333000 Дж/кг. Ответ округлить до 1 мм.

Решение

Из уравнения теплового баланса следует, что спустя время в стакане будет плавать остаток льда в холодной воде. Так как плавающий лед вытесняет объем воды, равный объему воды из растающего льда, то увеличение высоты воды в стакане равно

$$h = \frac{m_л + m_B}{\rho S} = 25 \text{ мм}.$$

Точность 1 мм.

Ответ: 25 мм $\pm$ 1 мм.

Система оценки

1. Правильное использование силы Архимеда — 5 баллов.
2. Правильное использование уравнения теплового баланса — 5 баллов.
3. Увеличение высоты воды в стакане в общем виде — 5 баллов.
4. Правильное численное значение — 5 баллов.

Максимум 20 баллов.

Задача III.1.2.2. Александровская колонна (20 баллов)

Высота монолитного гранитного ствола Александровской колонны на Дворцовой площади Санкт — Петербурга равна 25,6 м. Ствол немного сужается кверху так, его нижний диаметр равен 3,66 м, а верхний диаметр равен 3,15 м. Масса ствола составляет 612 тонн. Ствол был установлен на пьедестале высотой 10 м над площадью с помощью канатов и блоков. Определить работу силы тяжести при подъеме ствола из горизонтального положения на площади в вертикальное положение на пьедестале. $g = 10$ м/с^2 — ускорение свободного падения. Считать ствол колонны цилиндром с площадью основания S равной средней площади поперечного сечения ствола. Ответ дать в МДж и округлить до целого.

Решение

Площадь основания S равна средней площади поперечного сечения ствола $S = \frac{\pi(d_1^2+d_2^2)}{8} = \pi r^2$ с нижним и верхним диаметрами. Тогда работа силы тяжести равна

$$A = -mh(h_1 + \frac{h}{2} - r) = -mg(h_1 + \frac{h}{2} - \sqrt{\frac{d_1^2 + d_2^2}{8}}) = -130 \text{ МДж}.$$

Ответ: -130 МДж $\pm$ 1 МДж.

Система оценки

1. Правильная начальная высота — 5 баллов.
2. Правильная разность высот — 5 баллов.

3. Работа силы тяжести в общем виде — 5 баллов.
4. Правильное численное значение — 5 баллов.

Максимум 20 баллов.

Задача III.1.2.3. Три насоса (20 баллов)

Воду накачивают из колодца тремя насосами в бак. Расход воды (объем воды в единицу времени) через второй насос в 2 раза больше, чем через первый, а расход воды через третий насос в 2 раза больше, чем через второй. За три часа третий насос накачал воды на 8,0 м^3 больше, чем второй насос. Найти расход воды через первый насос в м^3/ч с точностью до десятых долей.

Решение

Расходы воды через насосы связаны между собой: $Q_2 = 2Q_1, Q_3 = 2Q_2, (Q_3 - Q_2)t = V$. Отсюда

$$Q_1 = \frac{V}{2t} = 1,3 \text{ м}^3/\text{ч}$$

Ответ: $1,3$ м^3/ч $\pm\, 0,1$ м^3/ч.

Система оценки

1. Правильная связь расхода воды с объемом — 5 баллов.
2. Правильная система уравнений расходов для 3 насосов — 5 баллов.
3. Расход воды через первый насос в общем виде — 5 баллов.
4. Правильное численное значение — 5 баллов.

Максимум 20 баллов.

Задача III.1.2.4. Радиатор отопления (20 баллов)

Через радиатор отопления за одну минуту проходит 5 кг жидкого теплоносителя — антифриза. В радиатор антифриз входит с температурой 75 °C, а выходит из радиатора с температурой 60 °C. Найти тепловую мощность радиатора. Плотность антифриза $\rho = 1,1 \cdot 10^3$ кг/м^3, а его удельная теплоемкость равна 3100 Дж/(кг·°C) Ответ дать в кВт и округлить до десятых долей.

Решение

Тепловая мощность равна потери радиатором тепла в единицу времени

$$P = c\frac{m}{t}(T_{\text{вход}} - T_{\text{выход}}) = 3,9 \text{ кВт}$$

Ответ: $3,9$ кВт $\pm\, 0,1$ кВт.

Система оценки

1. Правильное определение тепловой мощности — 5 баллов.
2. Правильное уравнение теплового баланса — 5 баллов.
3. Тепловая мощность радиатора в общем виде — 5 баллов.
4. Правильное численное значение — 5 баллов.

Максимум 20 баллов.

Задача III.1.2.5. Топливо для ядерного реактора (20 баллов)

В ядерном реакторе, имеющем форму цилиндра высотой 4,5 м и радиусом 2,8 м, топливные каналы (стержни) располагаются параллельно оси цилиндра в вершинах квадратной сетки. Расстояние между соседними каналами 30 см, а радиус каждого канала равен 3,5 см и высота равна высоте реактора. Топливные каналы заполнены двуокисью урана с плотностью 11 т/м^3. Считая, что каналы располагаются равномерно по сечению реактора, найти массу полной загрузки реактора ядерным топливом. Ответ дать в т с точностью до целого.

Решение

Число каналов в вершинах квадратной сетки для равномерного распределения $N = \frac{\pi R^2}{a^2}$. Тогда масса топлива равна

$$M = N\rho\pi r^2 H = \frac{\pi^2 R^2 r^2 \rho H}{a^2} = 52 \text{ т}.$$

Ответ: 52 т $\pm$ 5 т.

Система оценки

1. Правильное число топливных каналов — 5 баллов.
2. Правильная масса топлива в одном канале — 5 баллов.
3. Масса полной загрузки реактора в общем виде — 5 баллов.
4. Правильное численное значение — 5 баллов.

Максимум 20 баллов.

Физика. 10-11 класс

Задача III.1.3.1. Луч света в неоднородной среде (20 баллов)

Луч света падает вдоль поверхности соленой воды. Показатель преломления соленой воды плавно увеличивается с глубиной y. Поэтому траектория луча описывается функцией $y = kx^3$, где $k = 1,0$ м$^{-2}$, а координата x отсчитывается от точки

входа луча вдоль поверхности воды. Найти отношение показателя преломления соленой воды на глубине $y = 0,2$ м к показателю преломления на поверхности воды. Ответ округлить до десятых долей.

Решение

Закон преломления $n(0) = n(y)sin\alpha$. Угол α с нормалью к поверхности воды находим из уравнения

$$ctg\alpha = \frac{dy}{dx}.$$

Отсюда

$$\frac{n(y)}{n(0)} = \sqrt{1 + 9(ky^2)^{\frac{2}{3}}} = 1,4$$

Ответ: $1,4 \pm 0,05$.

Система оценки

1. Использование закона преломления — 5 баллов.
2. Уравнение касательной — 5 баллов.
3. Отношение показателей преломления в общем виде — 5 баллов.
4. Правильное численное значение — 5 баллов.

Максимум 20 баллов.

Задача III.1.3.2. Отношение теплоемкостей (20 баллов)

Идеальный одноатомный газ участвует в двух процессах: изобарическом и изохорическом. В обоих процессах газ имеет одинаковые начальные состояния и газу сообщают одинаковые количества теплоты. В изобарическом процессе объем газа увеличивается в 2 раза. Найти отношение теплоемкости газа в изобарическом процессе к теплоемкости в изохорическом процессе. Ответ округлить до сотых долей.

Решение

Изобара

$$Q = \frac{5}{2}p_0V_0.$$

Изохора

$$Q = \frac{3}{2}(p_1 - p_0)V_0.$$

Отсюда $p_1 = \frac{8}{3}p_0$, и $T_1 = \frac{8}{3}T_0$. Тогда

$$\frac{C_p}{C_V} = \frac{(\frac{Q}{2T_0 - T_0})}{(\frac{Q}{T_1 - T_0})} = \frac{5}{3} = 1,67$$

Ответ: $1,67 \pm 0,01$.

Система оценки

1. Использование уравнения Клапейрона — 5 баллов.
2. Правильное определение теплоемкостей — 5 баллов.
3. Отношение теплоемкостей в общем виде — 5 баллов.
4. Правильное численное значение — 5 баллов.

Максимум 20 баллов.

Задача III.1.3.3. Александровская колонна (20 баллов)

Высота монолитного гранитного ствола Александровской колонны на Дворцовой площади Санкт — Петербурга равна 25,6 м. Ствол немного сужается кверху так, его нижний диаметр равен 3,66 м, а верхний диаметр равен 3,15 м. Масса ствола составляет 612 тонн. Наверху ствола установлена скульптурная группа массой 37 тонн. Определить укорочение ствола под действием веса самого ствола и веса скульптурной группы. $g = 10$ м/с^2 — ускорение свободного падения. Модуль Юнга гранита равен $E = 5 \cdot 10^{10}$ Па. Модуль Юнга равен отношению

$$E = \frac{(\frac{F}{S})}{\frac{\Delta l}{l}},$$

где F — сжимающая сила, S — средняя площадь поперечного сечения ствола, Δl — укорочение небольшого участка ствола, l — первоначальная длина этого участка. Ответ дать в мм и округлить до десятых долей.

Решение

Вверху гранитный ствол сжат весом скульптурной группы, а внизу добавляется вес самого ствола. Поэтому среднее относительное укорочение всего ствола берем для небольшого участка в середине ствола.

$$\frac{\Delta l}{l} = \frac{(m + \frac{M}{2})g}{SE},$$

где M — масса ствола, m — масса скульптурной группы, $S = \frac{\pi(d_1^2+d_2^2)}{8}$ с нижним и верхним диаметрами. Тогда укорочение ствола высотой h равно:

$$\Delta h = \frac{\Delta l}{l} h = \frac{8(m + \frac{M}{2})gh}{\pi(d_1^2 + d_2^2)E} = 0,2 \text{ мм}$$

Ответ: $0,2$мм $\pm\, 0,05$ мм.

Система оценки

1. Использование уравнения с модулем Юнга — 5 баллов.
2. Правильное среднее относительное укорочение — 5 баллов.
3. Укорочение ствола высотой h в общем виде — 5 баллов.
4. Правильное численное значение — 5 баллов.

Максимум 20 баллов.

Задача III.1.3.4. Поток нейтронов (20 баллов)

При стабильной работе ядерного реактора количество нейтронов, покидающих реактор, равно количеству нейтронов, образующихся в ядерном процессе. Для реактора цилиндрической формы количество покидающих нейтронов в единицу времени пропорционально площади поверхности реактора. При заданном объеме реактора величиной 140 м^3, количество покидающих нейтронов должно быть минимально. Найти радиус основания ядерного реактора с точностью до десятых долей метра.

Решение

Количество покидающих нейтронов пропорционально $(2\pi RH + 2\pi R^2)$, объем цилиндра равен $V = \pi R^2 H$. Отсюда выражаем высоту, подставляем в первое выражение, берем производную по радиусу и приравниваем ее нулю. Получаем

$$R = (\frac{V}{2\pi})^{\frac{1}{3}} = 2,8 \text{ м}$$

Точность 0,05 м.

Ответ: $2,8$ м $\pm\, 0,05$ м.

Система оценки

1. Правильное количество покидающих нейтронов — 5 баллов.
2. Нахождение экстремума функции — 5 баллов.
3. Радиус основания ядерного реактора в общем виде — 5 баллов.
4. Правильное численное значение — 5 баллов.

Максимум 20 баллов.

Задача III.1.3.5. Квадрокоптер (20 баллов)

Квадрокоптер взлетной массой 7,5 кг неподвижно завис на высоте 100 м над поверхностью Земли. Он имеет 4 одинаковых винта с лопастями радиуса 15 см. Найти мощность двигателя одного винта с точностью до 10 Вт. Плотность воздуха 1,0 кг/м3. $g = 10$ м/с^2 — ускорение свободного падения.

Решение

Кинетическая энергия в единицу времени воздушного потока, проходящего через вращающиеся лопасти одного винта равна

$$P_1 = \frac{\Delta m V^2}{2\Delta t},$$

где $\Delta m = \rho S V \Delta t$ — масса воздуха проходящего через площадь $S = \pi r^2$ за время Δt. Отсюда мощность воздушного потока одного винта равна

$$P_1 = \frac{\rho \pi r^2 V^3}{2}.$$

Изменение импульса воздуха от четырех винтов равно силе тяжести, умноженной на время Δt: $4\rho S V^2 \Delta t = Mg\Delta t$. Имеем

$$P_1 = \frac{1}{16}\sqrt{\frac{M^3 g^3}{\rho \pi r^2}} = 150 \text{ Вт}.$$

Ответ: 150 Вт $\pm$ 5 Вт.

Система оценки

1. Кинетическая энергия в единицу времени воздушного потока — 5 баллов.
2. Правильное использование 2 закона Ньютона — 5 баллов.
3. Мощность двигателя одного винта в общем виде — 5 баллов.
4. Правильное численное значение — 5 баллов.

Максимум 20 баллов.