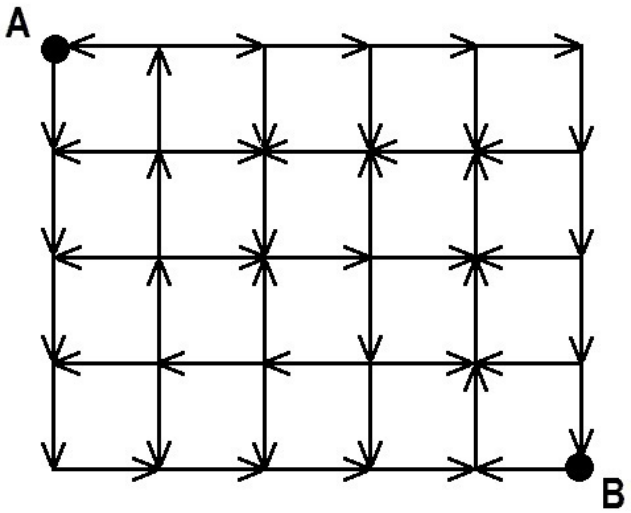
Первый отборочный этап

Задачи первого этапа. Информатика

Первая попытка. Задачи 8–11 класса

Задача I.1.1.1. Город перекрестков (20 баллов)



Вы разрабатываете навигатор для одного города. Этот город разбит улицами на квадратные кварталы, причем движение по любому из отрезков улицы в пределах каждого квартала строго одностороннее. С каждого перекрестка можно выехать только в разрешенных знаками направлениях. Требуется по прилагаемой карте города с указанными на ней разрешенными направлениями перемещения проложить самый короткий маршрут из точки A в точку B.

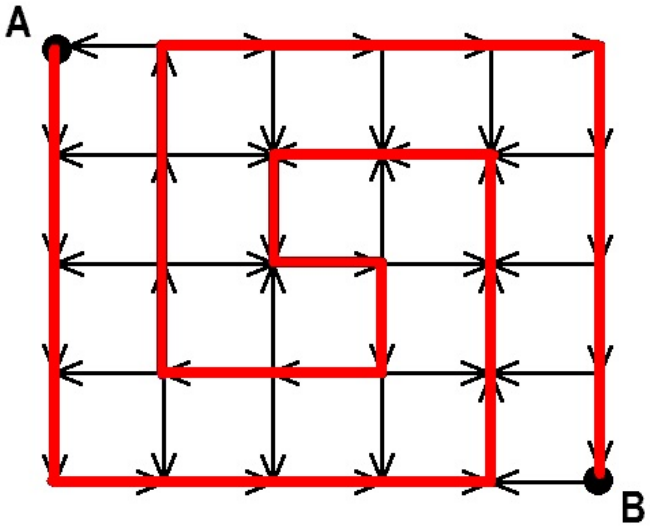
Формат входных данных

На вход подается карта перекрестков города. В первой строке содержатся два числа N — число кварталов с севера на юг и M — число кварталов с запада на восток ($1 \leqslant n \leqslant 50$). Точка A самая северо-западная, точка B самая юго-восточная. Далее в $2 \cdot N + 1$ строках содержится описание разрешенных направлений движения. Улицы города запад-восток описаны в нечетных строках. В каждой такой строке содержится по M символов без пробела, указывающих разрешенное движение на соответствующем участке. В четных строках содержится описание улиц север-юг. В этих строках содержится по $M + 1$ символов, указывающих возможное движение по отрезкам улиц север-юг. Движение на север, юг, запад, восток обозначается буквами n, s, w, e соответственно.

Формат выходных данных

В первую строку вывести число отрезков улиц в самом коротком маршруте из точки A в точку B. Во вторую строку нужно выдать описание этого маршрута в виде последовательности символов n, s, w, e без пробелов. Если кратчайших маршрутов несколько, выдать самый первый среди них по алфавитному порядку. Гарантируется, что из точки A можно попасть в точку B.

Пояснения к примеру



Примеры

Пример №1

Стандартный ввод
4 5 weeee snssss wewww snsnns weeew snnsns wwwew ssssns eeeew

Стандартный вывод
29 sssseeeennnwwseswwnnneeeessss

Пример программы-решения

Ниже представлено решение на языке C++.

```cpp
//Алгоритм Дейкстры

#include<bits/stdc++.h>

#define pb push_back
#define mp make_pair
#define f first
#define s second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

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

int dx[4] = {-1, 0, 1, 0},
    dy[4] = {0, 1, 0, -1};
char z[4] = {'n', 'e', 's', 'w'};

signed main(){

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

   int n, m;
   cin >> n >> m;
   vector<vector<vector<int> > > G(n+1, vector<vector<int> >(m+1));

   string s;
   for0(i, 2*n+1){
    cin >> s;
    if(i%2){
            int tx = i/2;
        for0(j, sz(s))
            if(s[j] == 's')
                G[tx][j].pb(2);
            else
                G[tx+1][j].pb(0);

    }
    else{
        int tx = i/2;
        for0(j, sz(s))
          if(s[j] == 'w')
            G[tx][j+1].pb(3);
          else
            G[tx][j].pb(1);
    }
   }

    string inf;
    inf.resize(6000, '#');
    vector<vector<string> > d(n+1, vector<string>(m+1, inf));
    vector<vector<int> > mark(n+1, vector<int>(m+1, 0));
    d[0][0] = "";

    for0(u, (n+1)*(m+1)){
```

```
61      int tx, ty, mn = 1e9;
62         for0(i, n+1) for0(j, m+1) if(mark[i][j] == 0 && sz(d[i][j]) < mn){
63             tx = i;
64             ty = j;
65             mn = sz(d[i][j]);
66         }
67
68         mark[tx][ty] = 1;
69
70         for0(i, sz(G[tx][ty])){
71             int tdir = G[tx][ty][i];
72             int nx = tx + dx[tdir];
73             int ny = ty + dy[tdir];
74             string ts = d[tx][ty] + z[tdir];
75
76             if(sz(ts) < sz(d[nx][ny]) || (sz(ts) == sz(d[nx][ny]) && ts < d[nx][ny]))
77                 d[nx][ny] = ts;
78         }
79     }
80
81     cout<<sz(d[n][m])<<endl;
82     cout<<d[n][m];
83 }
```

Пример программы-решения

Ниже представлено решение на языке C++.

```
1  //Обход в ширину
2
3  #include<bits/stdc++.h>
4
5  #define pb push_back
6  #define mp make_pair
7  #define x first
8  #define y second
9
10 #define all(x) (x).begin(), (x).end()
11 #define sz(a) (int)(a).size()
12 #define for0(i, n) for (int i = 0; i < (int)(n); i++)
13 #define for1(i, n) for (int i = 1; i <= (int)(n); i++)
14 #define int long long
15
16 using namespace std;
17 typedef pair<int, int> pii;
18 typedef long double ld;
19
20    int n, m;
21 int dx[4] = {0, -1, 1, 0},
22     dy[4] = {1, 0, 0, -1};
23 char z[4] = {'e', 'n', 's', 'w'};
24
25 bool in_sq(int x, int y){
26
27     return (x >= 0 && y >= 0 && x <= 2*n && y <= 2*m);
28 }
29
30 signed main(){
31     ios::sync_with_stdio(0), cin.tie(0), cout.tie(0);
```

```
32
33      string ans[200][200];
34      cin >> n >> m;
35      vector<string> G;
36      string s;
37      for0(i, 2*n+1){
38          cin >> s;
39          string z;
40          if(i%2 == 0){
41              for(auto q : s) z = z + "0" + q;
42                 z += "0";
43          }
44          if(i%2 == 1){
45              for(auto q : s) z = z + q + " ";
46          }
47          G.pb(z);
48      }
49      deque<pii> och;
50      och.pb({0, 0});
51      G[0][0] = '#';
52      ans[0][0] = "";
53
54      while(sz(och) > 0){
55          pii tp = och[0];
56          och.pop_front();
57
58          for0(i, 4){
59              int nx = tp.x + dx[i];
60              int ny = tp.y + dy[i];
61              if(in_sq(nx, ny) && G[nx][ny] == z[i] &&  G[nx + dx[i]][ny + dy[i]] == '0'
                ↪  ){
62                  G[nx + dx[i]][ny + dy[i]] = '#';
63                  ans[nx + dx[i]][ny + dy[i]] = ans[tp.x][tp.y] + z[i];
64                  och.pb({nx + dx[i], ny + dy[i]});
65              }
66          }
67      }
68
69      cout<<sz(ans[2*n][2*m])<<endl;
70      cout<<ans[2*n][2*m];
71  }
```

Задача I.1.1.2. WALL-E и кубики (20 баллов)

Робот WALL-E пытается упорядочить груду кубиков. В зоне его ответственности есть n расположенных друг за другом столбиков, каждый состоит из кубиков, стоящих друг на друге (в i-м столбике содержится h_i кубиков, столбики нумеруются слева направо). Робот действует по следующему алгоритму: он ищет самый левый столбик с номером i, в котором кубиков больше, чем в предыдущем, то есть $h_{i-1} < h_i$. Далее он берет верхний кубик столбика номер i и сбрасывает его на столбик номер $i-1$. Если существует столбик с номером $i-2$ и в нем теперь меньше чем в $i-1$-м столбике, то данный кубик перемещается далее на $i-2$-й столбик. И так до тех пор, пока этот кубик, либо не дойдет до верха первого столбика либо не упрется в более высокий столбик слева. Далее эта последовательность операций повторяется до тех пор, пока есть два рядом стоящих столбика таких, что $h_{i-1} < h_i$. По значениям высот исходных n столбиков вывести высоты упорядоченных n столбиков.

Формат входных данных

В первой строке содержатся число N — количество столбиков. В следующей строке находятся N чисел $h_1, h_2, \ldots, h_N$ через пробел, соответствующие высотам соответствующих столбиков. $1 \leqslant N \leqslant 10^5$, $1 \leqslant h_i \leqslant 2 \cdot 10^9$.

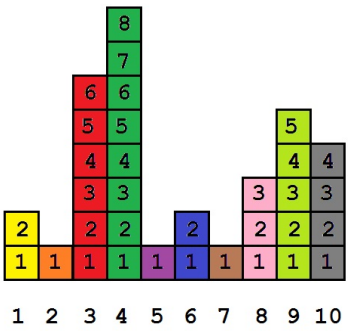
Формат выходных данных

Вывести N чисел через пробел, соответствующих высотам столбиков, после того как WALL-E закончит работу.

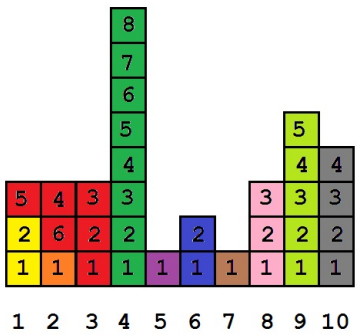
Пояснения к примеру

Следующая серия иллюстраций показывает, куда и какие кубики попали в процессе упорядочивания столбиков:

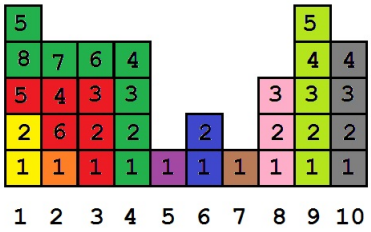
1. Стартовая ситуация



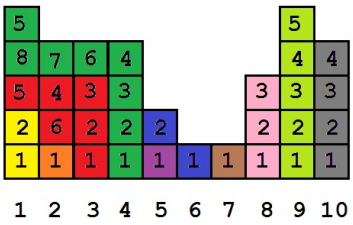
2. Упорядочили столбик 3



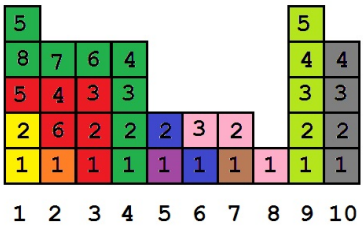
3. Упорядочили столбик 4



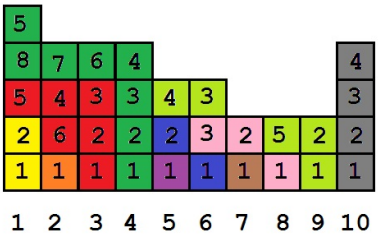
4. Упорядочили столбик 6



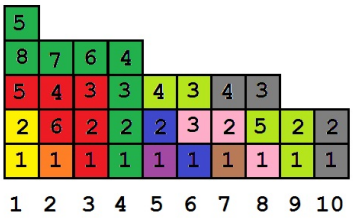
5. Упорядочили столбик 8



6. Упорядочили столбик 9



7. Упорядочили столбик 10, итоговое расположение кубиков



Примеры

Пример №1

Стандартный ввод
10 2 1 6 8 1 2 1 3 5 4
Стандартный вывод
5 4 4 4 3 3 3 3 2 2

Пример программы-решения

Ниже представлено решение на языке C++.

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

#define pb push_back
#define mp make_pair
#define x first
#define y second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

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

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

    int n;
    cin >> n;
    vector<int> h(n);
    for0(i, n) cin >> h[i];

    vector<pii> st;
    st.pb({2e9, 1});
```

```
29
30      for0(i, n){
31
32          if(h[i] < st.back().x) st.pb({h[i], 1});
33          else
34          if(h[i] == st.back().x) st.back().y++;
35          else
36          if(h[i] > st.back().x){
37              int thb = st.back().x;
38              int ost = h[i] - thb;
39              int d = 1;
40              int v = 0;
41              int sump = 0;
42
43              bool ok = 0;
44              while(!ok){
45                  int sp = d * v - sump;
46
47                  if(ost <= sp){
48                          ok = 1;
49                          int tr = ost + sump;
50                          int c = tr / d;
51                          int o = tr % d;
52
53                          if(o > 0){
54                              int thh = thb + c + 1;
55
56                              if(thh ==st.back().x) st.back().y += o;
57                              else st.pb({thh, o});
58                          }
59
60                          int thh = thb + c;
61
62                          if(thh == st.back().x) st.back().y += (d - o);
63                          else st.pb({thh, d - o});
64                  }
65                  else{
66                      d += st.back().y;
67                      v = st[sz(st)-2].x - thb;
68                      sump += (st.back().x - thb) * st.back().y;
69                      st.pop_back();
70                  }
71              }
72          }
73      }
74
75      for(int i = 1; i < sz(st); i++)
76          for0(j, st[i].y) cout<<st[i].x<<' ';
77  }
```

Задача I.1.1.3. Послание внеземного разума (10 баллов)

Сенсация! Известный специалист по UFO профессор Персиков получил послание от внеземного разума. Во всяком случае он так считает. Послание представляет собой непрерывную последовательность из сигналов нескольких типов, каждый из этих типов для определенности можно обозначить малой буквой латинского алфавита. Профессор считает, что доказательством искусственности происхождения сигнала служит его периодичность. При этом период должен быть строго равен «константе

Персикова» — числу P. Такую последовательность назовем P — периодичной.

К сожалению, некоторые сигналы из-за прохождения сквозь бездны пространства (а может даже и времени) были утеряны. Их заменили на знак вопроса. Теперь профессору для дальнейшей работы требуется выяснить, можно ли как-то заменить утерянные сигналы на буквы так, чтобы последовательность стала P — периодичной. Для чистоты эксперимента полученная из космоса последовательность была размещена среди себе подобных. Не подведите, в ваших руках — будущее межгалактических взаимоотношений!

Формат входных данных

В первой строке содержится два числа через пробел: N — общее количество строк, которые вам нужно проверить на P-периодичность и число P — константа Персикова ($1 \leqslant P \leqslant 5 \cdot 10^3$). Далее содержится N непустых строк, состоящих из малых букв латиницы и знаков вопроса. $1 \leqslant N \leqslant 5 \cdot 10^4$, суммарная длина всех строк не превосходит $2 \cdot 10^5$.

Формат выходных данных

Вывести N строк, в каждой из которых вывести либо «YES», если соответствующую строку можно сделать P-периодичной путем замены всех знаков вопроса на некоторые буквы, и «NO» в противном случае. Будем считать последовательность P-периодичной, если для любых двух символов, расстояние между которыми кратно P верно, что они совпадают.

Примеры

Пример №1

Стандартный ввод
8 4 abacabaca abracadabra aa?aaa?aaaaaaaaaa q ??????? q?er?w? q?erw?? q?er?wer?werw?er

Стандартный вывод
YES NO YES YES YES YES NO NO

Пример программы-решения

Ниже представлено решение на языке C++.

```cpp
//Метод карманов
#include<bits/stdc++.h>
#define pb push_back
#define mp make_pair
#define x first
#define y second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

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

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

    int u, p;
    cin >> u >> p;
        while(u--){
            string s;
            cin >> s;
            vector<set<char> > v(p);
            for0(i, sz(s))
              if(s[i] != '?') v[i%p].insert(s[i]);

              bool ok = 1;
            for(auto q : v)
                if(sz(q) > 1)
                    ok = 0;

                 cout<<  ((ok)? "YES\n" : "NO\n");
        }
}
```

Пример программы-решения

Ниже представлено решение на языке C++.

```cpp
//Эффективный перебор
#include<bits/stdc++.h>

#define pb push_back
#define mp make_pair
#define x first
#define y second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
```

```
#define int long long

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

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

    int u, p;
    cin >> u >> p;
        while(u--){
            string s;
            cin >> s;
                bool ok = 1;

                for0(i, p){
                    char c = '#';
                for(int j = 0; i+j < sz(s); j += p) {
                    if(s[i+j] != '?')
                        if(c == '#') c = s[i+j];
                    else
                        if(c != s[i+j]) ok = 0;
                }
                }

                 cout<<  ((ok)? "YES\n" : "NO\n");
        }

}
```

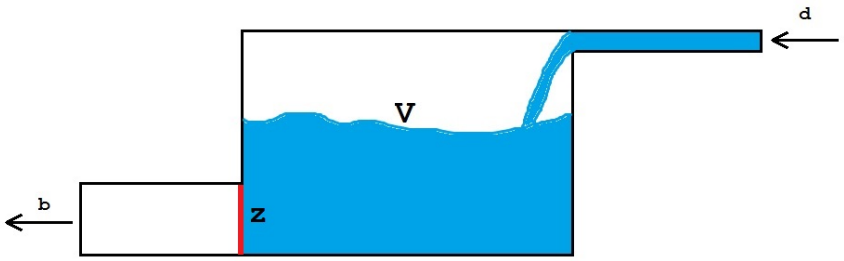
Пример программы-решения

Ниже представлено решение на языке Python3.

```
n, p = map(int, input().split())
lines = []
f = open('text.txt', 'w')
for i in range(n):
    lines.append(input())
for i in range(n):
    line = lines[i]
    q = True
    for j in range(p):
        c = ''
        k = j
        while k < len(line):
            if line[k] != '?':
                if c == '':
                    c = line[k]
                else:
                    if line[k] != c:
                        q = False
                        break
            k += p
        if not q:
            break


```

```
    if q:
        f.write("YES" + '\n')
    else:
        f.write("NO" + '\n')
f.close()
```

Задача I.1.1.4. Водопровод с резервной емкостью (20 баллов)



Рассмотрим следующую модель водопровода. Имеется резервная емкость объемом V литров. По входной трубе постоянно поступает вода со скоростью d литров в минуту. По выходной трубе жидкость выливается со скоростью b литров в минуту. Выполняется условие, что $d < b$. В начальный момент времени выход Z из емкости перекрыт. В тот момент, когда емкость наполняется, выход открывается и вода поступает на выпуск. Так как $d < b$, то в какой-то момент емкость полностью опустеет, и заслонка Z снова закрывается, после чего процесс повторяется снова.

Очевидно, что при этом в работе водопровода случаются паузы, в течение которых происходит заполнение резервной емкости. К ним относится и первая пауза, служащая для первичного наполнения емкости водой. По техническим причинам требуется, чтобы длина каждой такой паузы не превышала величины p. При этом нас интересует суммарное время t, в течение которого выходная труба была открыта. Количество пауз n при этом должно быть минимальным возможным.

В рассматриваемой модели используются сколь угодно малые доли времени и объема. Считается, что заслонка Z срабатывает мгновенно.

По заданным величинам d, b, t, p требуется найти такой целый объем V, при котором число пауз n было минимально возможным для заданного времени подачи воды t, а среди всех объемов, обеспечивающих такое время работы подачи воды и такое количество пауз найти самый маленький.

Формат входных данных

В первой строке содержится четыре целых числа d, b, t, p через пробел: $1 \leqslant d < b \leqslant 2 \cdot 10^9$, $1 \leqslant t \leqslant 15000$, $1 \leqslant p \leqslant 5000$.

Формат выходных данных

Вывести одно целое число — объем V резервной емкости, обеспечивающей необходимые ограничения на подачу воды. Если таких объемов несколько, вывести минимальный.

Пояснения к примеру

Рассмотрим первый пример из условия. Скорость подачи d равна 5 литров в минуту, скорость выпуска b равна 10 литров в минуту, требуется обеспечить суммарную работу выпуска в течение $t = 32$ минут, при этом максимальный размер любой паузы не должен превышать $p = 8$ минут.

Если мы установим объем резервной емкости 40 литров, то:

- ровно за 8 минут (что разрешено условием) она наполнится со скоростью 5 литров в минуту;
- далее пойдет процесс выпуска: 40 литров будут выпущены за 4 минуты со скоростью выпуска 10 литров в минуту, но за это время в емкость попадет $4 \cdot 5 = 20$ литров новой воды. Она будет выпущена за 2 минуты, но за это время поступит $2 \cdot 5 = 10$ литров новой воды, которая будет выпущена за 1 минуту и так далее. Так как доли времени и объема могут быть сколь угодно малыми, получим при подсчете времени работы выпуска следующую сумму: $4 + 2 + 1 + 0,5 + 0,25 + 0,125 + ...$, которая в итоге равна 8. То есть через 8 минут емкость полностью опустеет, и заслонка закроется. Для того, чтобы время работы выпуска было равно 32 минуты, потребуется $32/8 = 4$ таких цикла, а значит и 4 паузы. За меньшее число пауз работу выпуска в течение 32 минут при существующих ограничениях обеспечить не получится.
  Очевидно, что $V = 40$ литров — максимально возможный разрешенный объем, иначе первая же пауза будет больше допустимой. С другой стороны, если мы попробуем уменьшить объем до 39 литров, то получим:
- первоначальное заполнение будет произведено за $39/5 = 7.8$ минуты. Далее процесс выпуска будет работать $39/10 + 39/20 + 39/40 + ... = 7,8$ минуты. Тогда за 4 цикла выпуск будет работать 31.2 минуты и для обеспечения 32 минут работы выпуска потребуется пятая пауза.

Примеры

Пример №1

Стандартный ввод
`5 10 32 8`

Стандартный вывод
`40`

Пример программы-решения

Ниже представлено решение на языке C++.

```
1  //Бинарный поиск
2  #include<bits/stdc++.h>
3
4  #define pb push_back
5  #define mp make_pair
6  #define x first
7  #define y second
8
9  #define all(x) (x).begin(), (x).end()
10 #define sz(a) (int)(a).size()
```

```
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

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


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

  int d, b, t, p;

  cin >> d >> b >> t >> p;

  int y = t * (b-d);
  int z = d * p;

  int a = y / z + (y % z != 0);

  int L = 0, R = d * p;

  while(R - L > 1){
    int M = (R + L) / 2;

    if( M * a >= t * (b-d)) R = M; else L = M;

  }

  cout << R;
}
```

Задача I.1.1.5. Разбиения на различные множители (30 баллов)

Эта задача формулируется предельно кратко.

Дано натуральное число N. Требуется найти число способов представить его в виде произведения попарно различных множителей больших 1.

Формат входных данных

В первой строке содержится одно натуральное число $2 \leqslant N \leqslant 10^{12}$.

Формат выходных данных

Вывести одно число — количество способов представить число N в виде произведения попарно различных множителей больших 1.

Пояснения к примеру

Имеется 7 различных способов представить число 48 в виде произведения (в том числе и вырожденного) попарно различных множителей больших 1: 48, $2 \cdot 24$, $3 \cdot 16$,

$4 \cdot 12$, $6 \cdot 8$, $2 \cdot 3 \cdot 8$, $2 \cdot 4 \cdot 6$.

Примеры

Пример №1

Стандартный ввод
48
Стандартный вывод
7

Пример программы-решения

Ниже представлено решение на языке C++.

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

#define pb push_back
#define mp make_pair
#define x first
#define y second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

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

vector<pii> raz;
set<int> del;
vector<int> rdel;

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

    int n, dn;
    cin >> n;
    dn = n;

    for(int i = 2; i <= 1e6+1 && dn > 1; i++) if(dn % i == 0){
        int t = 0;
            while(dn % i == 0){
                t++;
                dn /= i;
            }
        raz.pb({i, t});
    }

    if(dn > 1) raz.pb({dn, 1});

    del.insert(1);

    for(auto q : raz){
```

```
        for0(i, q.y){
            rdel.resize(0);
            for(auto w : del) rdel.pb(w * q.x);
            for(auto w : rdel) del.insert(w);
        }
    }

    rdel.resize(0);
    for(auto q : del) rdel.pb(q);
    int r = sz(rdel);

    map<int, int> rev;
    for0(i, r) rev[rdel[i]] = i;

    vector<vector<int> > dp(r, vector<int>(r, 0));

    dp[0][0] = 1;
    for(int i = 1; i < r; i++) dp[i][0] += dp[i-1][0];

    for(int j = 1; j < r; j++){
        for(int i = 1; i < r; i++) if(rdel[j] % rdel[i] == 0)
            dp[i][j] = dp[i-1][rev[rdel[j] / rdel[i]]];
        for(int i = 1; i < r; i++) dp[i][j] += dp[i-1][j];

    }
    cout<<dp[r-1][r-1];
}
```

Пример программы-решения

Ниже представлено решение на языке Python3.

```
from bisect import bisect_left

def binary_search(a, x):
    i = bisect_left(a, x)
    if i != len(a) and a[i] == x:
        return i
    else:
        return -1

n = int(input())
dividers = []
for i in range(1, math.ceil(math.sqrt(n))):
    if n % i == 0:
        dividers.append(i)
divs = dividers.copy()
if int(math.sqrt(n)) ** 2 == n:
    dividers.append(int(math.sqrt(n)))
for x in divs[::-1]:
    dividers.append(n // x)
dp = [[0 for i in range(len(dividers))] for j in range(len(dividers))]
dp[0][0] = 1
for a in range(len(dividers) - 1):
    pref_sum = 0
    for i in range(len(dividers) - 1):
        pref_sum += dp[a][i]
        dp[a][i] = pref_sum
        if dividers[a + 1] % dividers[i + 1] == 0:
```

```
            bs = binary_search(dividers, dividers[a + 1] // dividers[i + 1])
            if bs != -1:
                dp[a + 1][i + 1] = dp[bs][i]
print(sum(dp[-1]))
```

Вторая попытка. Задачи 8–11 класса

Задача I.1.2.1. Распознавание деталей (15 баллов)

Ваша задача — написать программу для распознавания деталей заданного вида на конвейере. Изображение нужной детали вводится при начале распознавания. Далее вводится изображение части конвейера, на котором могут находиться детали различных видов. Требуется распознать и выделить все изображения заданной детали.

Формат входных данных

Сначала приведено изображение искомой детали. Оно имеет размер 5×5. В 5 строках содержится по 5 символов «.» и «#», где решетки соответствуют детали, а точки — фону. Гарантируется, что изображение детали является 4-х связной фигурой. После изображения детали идет изображение текущего состояния конвейера. Это изображение имеет размер 10×20. В последующих 10 строках содержится по 20 символов. Каждая деталь на конвейере 4-х связная и имеет свой цвет, обозначенный малой буквой латинского алфавита. Разные детали обозначены разными буквами. Таким образом изображение конвейера может содержать буквы от «a» до «z» и символ «.», по прежнему обозначающий фон. На конвейере находится не более 26 деталей. Следует учитывать, что требуемые детали могут быть повернуты на угол кратный 90 градусов и/или лежать обратной стороной.

Формат выходных данных

Требуется вывести изображение конвейера, на котором все вхождения заданной детали выделены соответствующими буквами, но переведенными в верхний регистр, остальные символы должны остаться без изменений.

Примеры

Пример №1

Стандартный ввод
....##. ...## ...#. ....a..... ....aaac.....gg... ....a.ccc.....gg... ....b.c....d..... ....bbb...ddd..... .....b.....dhh..... ....f.....hh...e... ...ff.....h.eee... ....ff.....ee.. .....mmmmmmmmmmm

Стандартный вывод
.....A..... ....AAAC.....gg... ....A.CCC.....gg... ....B.C....d..... ....BBB...ddd..... .....B.....dHH..... ....F.....HH...e... ...FF.....H.eee... ....FF.....ee.. .....mmmmmmmmmmm

Пример программы-решения

Ниже представлено решение на языке C++.

```
1  //Сравнение с шаблоном, моделирование
2
3  #include<bits/stdc++.h>
4
5  #define pb push_back
6  #define mp make_pair
7  #define x first
8  #define y second
9
10 #define all(x) (x).begin(), (x).end()
11 #define sz(a) (int)(a).size()
12 #define for0(i, n) for (int i = 0; i < (int)(n); i++)
13 #define for1(i, n) for (int i = 1; i <= (int)(n); i++)
14 #define int long long
15
16 using namespace std;
17 typedef pair<int, int> pii;
18 typedef long double ld;
19 typedef vector<string> vs;
```

```
20
21  vs knv;
22  vector<vs > obr(8);
23  string s;
24  set<char> ans;
25
26  vs r_90(vs &a){
27      vs r = a;
28      for0(i, sz(a))
29          for0(j, sz(a[i])) r[j][sz(a) - i - 1] = a[i][j];
30      return r;
31  }
32
33  vs mirr(vs &a){
34      vs r = a;
35      for0(i, sz(r)) reverse(all(r[i]));
36      return r;
37  }
38
39  vs cut(int x, int y, char z){
40      vs r;
41      for0(i, 5) r.pb(knv[x+i].substr(y, 5));
42          for0(i, 5) for0(j, 5) if(r[i][j] != z) r[i][j] = '.'; else r[i][j] = '#';
43
44      return r;
45  }
46
47  int toko(vs &a, char z){
48      int r = 0;
49      for(auto q : a) for(auto w : q) r += (w == z);
50      return r;
51  }
52
53  signed main(){
54      ios::sync_with_stdio(0), cin.tie(0), cout.tie(0);
55
56      for0(i, 5){
57          cin >> s;
58          obr[0].pb(s);
59      }
60
61      for1(i, 3) obr[i] = r_90(obr[i-1]);
62
63      obr[4] = mirr(obr[0]);
64
65      for(int i = 5; i < 8; i++) obr[i] = r_90(obr[i-1]);
66
67      int ko = toko(obr[0], '#');
68
69      string z;
70      for0(i, 30) z += ".";
71      for0(i, 5) knv.pb(z);
72      for0(i, 10) {
73          cin >> s;
74          for0(j, 5) s = "." + s;
75          for0(j, 5) s += ".";
76          knv.pb(s);
77      }
78      for0(i, 5) knv.pb(z);
79
```

```
80      map<char, int> msk;
81      for(auto q : knv)
82          for(auto w : q) msk[w]++;
83
84      for(auto q : msk) if(q.y == ko){
85              bool ok = 0;
86          for0(i, 14) for0(j, 24) {
87              vs d = cut(i, j, q.x);
88
89                  if(toko(d, '#') == ko){
90                      bool ok1;
91                      for0(j, 8){
92                          if(d == obr[j]){
93                                  ans.insert(q.x);
94                          }
95                      }
96                  }
97          }
98      }
99
100     for(int i = 5; i < 15; i++){
101         for(int j = 5; j < 25; j++)
102         if(ans.find(knv[i][j]) != ans.end()) cout<<(char)(knv[i][j] - 'a' + 'A');
103         else cout<<knv[i][j];
104         cout<<endl;
105     }
106 }
```

Задача I.1.2.2. Мерцающие звезды (22 баллов)

Современных звездных путешественников очень трудно удивить. Однако фирма Amazing Star Travel хочет предложить нечто новое: наблюдения за мерцающими звездами. Это очень эффектное явление, возникающее в тот момент, когда мощную звезду заслоняет планета. Для этого разработан маршрут между двумя точками A и B. Специалисты фирмы выделили N наиболее ярких звезд в видимой части космоса и отметили M крупных планет. Осталось подсчитать, сколько раз за время путешествия по отрезку AB путешественники насладятся видом мерцающей звезды.

Формат входных данных

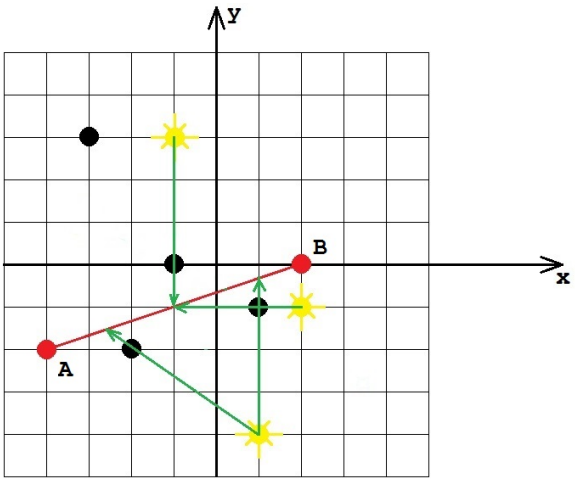
В первой строке содержится четыре целых числа через пробел XA, YA, XB, YB — координаты точек A и B. Во второй строке содержатся числа N и M, разделенные пробелом ($0 \leqslant N, M \leqslant 100$) — количество звезд и количество планет соответственно. В каждой из следующих N строк содержатся координаты очередной звезды. Далее в каждой из следующих M строк содержатся координаты очередной планеты. Все координаты целые, по модулю не превосходят 1000. Гарантируется, что никакие три точки из всех вышеперечисленных не находятся на одной прямой.

Формат выходных данных

В ответе нужно выдать одно число — количество случаев, когда при движении по отрезку из точки A в точку B какая-либо звезда будет заслонена от наблюдателя планетой. Если какие-либо две звезды мерцают одновременно, то это считается как

два независимых случая. Все упомянутые объекты считаем материальными точками, для упрощения вычислений все рассматриваем на плоскости. Помимо этого, согласно теории относительности, путешествие с точки зрения внешнего наблюдателя, совершается мгновенно, то есть положение звезд и планет за время путешествия не изменяется, однако для путешественников оно достаточно длительное, чтобы насладиться захватывающими видами.

Пояснения к примеру



Примеры

Пример №1

Стандартный ввод
-4 -2 2 0 3 4 -1 3 2 -1 1 -4 -3 3 -1 0 -2 -2 1 -1

Стандартный вывод
4

Пример программы-решения

Ниже представлено решение на языке C++.

```
1  //Целочисленно, сравнение площадей
2  #include<bits/stdc++.h>
3
4  #define pb push_back
5  #define mp make_pair
6  #define x first
```

```
#define y second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

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

int sqt2(pii a, pii b, pii c){
    pii v1 = {b.x - a.x, b.y - a.y};
    pii v2 = {c.x - a.x, c.y - a.y};

    return abs(v1.x*v2.y - v1.y*v2.x);
}

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

    pii A, B;
    cin >> A.x >> A.y >> B.x >> B.y;

    int n, m;
    cin >> n >> m;
    vector<pii> s(n), p(m);
    for0(i, n) cin >> s[i].x >> s[i].y;
    for0(i, m) cin >> p[i].x >> p[i].y;

    int ans = 0;

    for0(i, n){
        int S = sqt2(A, B, s[i]);
        for0(j, m){
           int s1 = sqt2(A, B, p[j]);
           int s2 = sqt2(A, p[j], s[i]);
           int s3 = sqt2(B, p[j], s[i]);
           if(s1 + s2 + s3 == S) ans++;
        }
    }
    cout << ans;
}
```

Пример программы-решения

Ниже представлено решение на языке C++.

```
//вещественно, бинпоиском точку пересечения
#include<bits/stdc++.h>

#define pb push_back
#define mp make_pair
#define x first
#define y second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
```

```cpp
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

using namespace std;
typedef long double ld;
typedef pair<ld, ld> pld;

ld eps = 1e-7;

struct line{
    ld A, B, C;
};

ld dist(pld a, pld b){
    return sqrt((a.x-b.x)*(a.x-b.x)+(a.y-b.y)*(a.y-b.y));
}

line L(pld a1, pld a2){
    ld A = a2.y - a1.y;
    ld B = a1.x - a2.x;
    ld C = a2.x*a1.y - a1.x*a2.y;

    line r = {A, B, C};
    return r;
}

int sgn(ld x){
    if(x < -eps) return -1;
    if(x > eps) return 1;
    return 0;
}

pld tp(pld A, pld B, pld s, pld p){
    line AB = L(A, B);

    ld L = 0, R = 1e6;

    pld v = {p.x - s.x, p.y - s.y};
    pld ab = {B.x - A.x, B.y - A.y};
    ld Fs = AB.A * (s.x + L * ab.x) + AB.B * (s.y + L * ab.y) + AB.C;

    if(abs(v.x*ab.y - v.y*ab.x) < eps) return{1e6, 1e6};

    while(R - L > eps){
        ld M = (L+R)/2.0;

        pld m = {s.x + M*v.x, s.y + M*v.y};

        ld F = AB.A * m.x + AB.B * m.y + AB.C;

        if(sgn(F) == sgn(Fs)) L = M; else R = M;
    }

    return {s.x + L*v.x, s.y + L*v.y};
}

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

```

```
71      pld A, B;
72      cin >> A.x >> A.y >> B.x >> B.y;
73
74      int n, m;
75      cin >> n >> m;
76      vector<pld> s(n), p(m);
77      for0(i, n) cin >> s[i].x >> s[i].y;
78      for0(i, m) cin >> p[i].x >> p[i].y;
79
80      int ans = 0;
81
82      for0(i, n)
83      for0(j, m){
84          line AB = L(A, B);
85          ld tos = AB.A * s[i].x + AB.B * s[i].y + AB.C;
86          ld top = AB.A * p[j].x + AB.B * p[j].y + AB.C;
87
88          if(sgn(tos) == sgn(top)){
89
90              pld w = tp(A, B, s[i], p[j]);
91
92              if(abs(w.x) < 1e6 && abs(w.y) < 1e6){
93
94                  if(abs(dist(A, w) + dist(w, B) - dist(A, B)) < eps) {
95                          ans++;
96                  }
97              }
98
99          }
100     }
101
102     cout << ans;
103 }
```

Задача I.1.2.3. Послание внеземного разума 2 (23 баллов)

Профессор Персиков снова получил послание внеземного разума. Он по-прежнему считает, что доказательством этого является его периодичность. При этом период должен быть равен «константе Персикова» — числу P. К сожалению, послание не совсем периодичное. Если сказать более точно, то оно совсем не периодичное. Однако, это никак не останавливает исследователя космических глубин. Он говорит, что некоторые сигналы были неправильно откалиброваны, отфильтрованы и интерпретированы. По-прежнему мы будем считать, что все сигналы отображаются малыми буквами латинского алфавита, но в данной задаче все они распознаны и знаков вопроса нет. Тем не менее профессор, согласно своей теории, может заявить, что все вхождения такой-то буквы интерпретированы неверно и их **все** следует заменить на вхождения какой-то другой (одной и той же) буквы. Более строго: пусть на позициях $p_{i_1}, p_{i_2}, \ldots p_{i_k}$ и только на них в последовательности находится одна и та же буква. Профессор может выбрать любую другую букву (как встречающуюся в слове, так и не встречающуюся) и поставить ее во всех этих позициях. Например, в слове *qqzbbacabadaba* он может выбрать все вхождения буквы b и заменить их на букву a, получив слово *qqzaaacaaadaaa* (это считается одной заменой, независимо от числа вхождений). Очевидно, что таким образом любое послание можно сделать P-периодическим, но профессор заинтересован сделать как можно меньше таких замен. При этом он хочет получить лексикографически минимальное послание.

Формат входных данных

В первой строке содержится число P — константа Персикова ($1 \leqslant P \leqslant 10^5$). В следующей строке содержится непустая последовательность, состоящая из малых букв латиницы. Длина этой строки не превосходит $2 \cdot 10^2$.

Формат выходных данных

Вывести строку, которая получается из исходной путем минимального числа операций замены вхождений всех букв одного вида на вхождения какой-то (одной и той же) другой буквы. Итоговая строка должна быть P-периодической, то есть любые две ее буквы, расстояние между которыми кратно P должны совпадать. Среди всех таких строк вывести лексикографически минимальную (первую в алфавитном порядке).

Примеры

Пример №1

Стандартный ввод
4 qqzbbacabadaba

Стандартный вывод
aacaaacaaacaaa

Пример программы-решения

Ниже представлено решение на языке C++.

```
1  //граф на буквах, обход в глубину
2  #include<bits/stdc++.h>
3
4  #define pb push_back
5  #define mp make_pair
6  #define x first
7  #define y second
8
9  #define all(x) (x).begin(), (x).end()
10 #define sz(a) (int)(a).size()
11 #define for0(i, n) for (int i = 0; i < (int)(n); i++)
12 #define for1(i, n) for (int i = 1; i <= (int)(n); i++)
13 #define int long long
14
15 using namespace std;
16 typedef long double ld;
17 typedef pair<int, int> pii;
18
19 map<char, set<char> > G;
20 map<char, int> rzb;
21 int num = 0;
22 vector<set<char> > toans;
23 set<char> emp;
24
25 void dfs(char a){
26     toans[rzb[a]].insert(a);
```

```
    for(auto to : G[a]) if(rzb[to] == 0) {
        rzb[to] = rzb[a];
        dfs(to);
    }
}

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

    int p;
    cin >> p;
    string s;
    cin >> s;

    vector<set<char> > op(p);

    for0(i, sz(s))  op[i%p].insert(s[i]);

    for0(i, p)
        for(auto q : op[i])
        for(auto w : op[i]) {
            G[q].insert(w);
            G[w].insert(q);
        }

        toans.pb(emp);
        for(auto q : G)
        if(rzb[q.x] == 0){
            num++;
            toans.pb(emp);
            rzb[q.x] = num;
            dfs(q.x);
        }

    for0(i, sz(s)) s[i] = *(toans[rzb[s[i]]].begin());
    cout<<s;
}
```

Задача I.1.2.4. Проект «Ровные дороги» (10 баллов)

При проектировании новой автодороги было принято решение сделать ее не более чем из двух абсолютно горизонтальных участков. Будущую трассу разбили на n равных по длине малых отрезков. Будем считать, что в пределах одного малого отрезка местность имеет одну и ту же высоту h_i. При этом в целях эффективной трансформации местности требуется для выравнивания использовать исключительно грунт с этой же трассы. Это означает, что можно с некоторого малого отрезка высоты h_i взять некоторое количество грунта d так, что высота этого участка станет $h_i - d > 0$. Далее эти d единиц грунта обязательно нужно поместить на другой малый отрезок высоты h_j так, что его высота станет $h_j + d$. Перемещать грунт можно только в пределах одного из двух выбранных участков, то есть отрезки номер i и номер j должны принадлежать одному и тому же горизонтальному после выравнивания участку. Технические требования таковы, что эти два выровненных участка должны иметь целочисленную высоту над уровнем моря. Таким образом, для начала требуется определить сколькими способами можно переместить грунт с возвышенностей в низины так, что образуется не более чем два горизонтальных участка трассы, каждый из которых имеет целочисленную высоту.

Формат входных данных

В первой строке содержится число n — количество малых отрезков, на которые разбили трассу, $2 \leqslant n \leqslant 10^2$. Во второй строке указаны высоты h_i этих отрезков через пробел в порядке слева направо, $1 \leqslant h_i \leqslant 2 \cdot 10^4$.

Формат выходных данных

Вывести количество способов достичь требуемых показателей. Если у двух разбиений трассы в результате выравнивания конечные результаты совпадают, то они считаются одинаковыми.

Пояснения к примеру

Пример №1

Можно указать следующие разбиения не более чем на два участка, после выравнивания в пределах которых образуются целые высоты: [10 4] и [2 7 5 8 6 6 15], результат выравнивания: [7 7] и [7 7 7 7 7 7 7]

[10 4 2 7 5 8] и [6 6 15], результат выравнивания: [6 6 6 6 6 6] и [9 9 9]

[10 4 2 7 5 8 6 6] и [15], результат выравнивания: [6 6 6 6 6 6 6 6] и [15]

[10 4 2 7 5 8 6 6 15], результат выравнивания: [7 7 7 7 7 7 7 7 7].

Результаты выравнивания в первом и последнем способе совпадают, таким образом общий ответ равен 3.

Пример №2

Во втором примере получатся следующие варианты разбиения и выравнивания: [3] [5 2 7 6 4 5 8 1 7], результат выравнивания: [3] и [5 5 5 5 5 5 5 5 5]

[3 5] [2 7 6 4 5 8 1 7], результат выравнивания: [4 4] и [5 5 5 5 5 5 5 5]

[3 5 2 7 6 4 5 8] [1 7], результат выравнивания: [5 5 5 5 5 5 5 5] и [4 4]

Примеры

Пример №1

Стандартный ввод
9 10 4 2 7 5 8 6 6 15
Стандартный вывод
3

Пример №2

Стандартный ввод
10 3 5 2 7 6 4 5 8 1 7
Стандартный вывод
3

Пример программы-решения

Ниже представлено решение на языке C++.

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

#define pb push_back
#define mp make_pair
#define x first
#define y second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

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

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

    int n;
    cin >> n;
    vector<int> v(n);
    for0(i, n) cin >> v[i];

    vector<int> suml(n), sumr(n);

    int sum = 0;
    for0(i, n){
        sum += v[i];
        suml[i] = sum;
    }

    sum = 0;
    for(int i = n-1; i >= 0; i--){
        sum += v[i];
        sumr[i] = sum;
    }

     int ans = 0, o1 = 0;
    if(sum % n == 0){
        ans++;
        o1++;
    }

    for(int i = 0; i < n-1; i++){
        if(suml[i] % (i+1) == 0 && sumr[i+1] %(n - i - 1) == 0){
            if(o1 == 1){
                if(suml[i] / (i+1) != sum / n) {
                        ans++;
                }
            }
            else{
                ans++;
            }
        }
```

```
57      }
58      cout << ans;
59  }
```

Задача I.1.2.5. Прогулка по мостам (30 баллов)

Возможно, вы знаете историю про то, как Эйлер гулял по мостам Кенигсберга. Допустим теперь, что Эйлер попал на некий архипелаг, между некоторыми островами которого имеются мосты. Мосты построены таким образом, что между любыми двумя островами архипелага можно построить путь по этим мостам, причем этот путь единственный. Все острова пронумерованы от 1 до n. Эйлер находится на острове номер 1 и желает совершить следующую прогулку: с острова номер 1 он хочет дойти до острова номер 2, далее до острова номер 3 и так далее до острова номер n. С него он хочет вернуться на остров номер 1. Он прекрасно понимает, что при этом ему придется посещать некоторые мосты по многу раз. Осталось выяснить, сколько раз суммарно он пройдет по мостам во время своей прогулки.

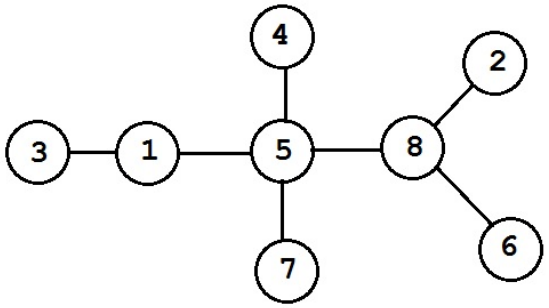
Формат входных данных

В первой строке содержится число n — количество островов в архипелаге, $2 \leqslant n \leqslant 10^5$. В следующих $n-1$ строках содержатся по два числа a_i и b_i через пробел — номера островов, соединенных мостом, $1 \leqslant a_i, b_i \leqslant n, a_i \neq b_i$. Гарантируется, что между любыми двумя островами архипелага можно построить путь по этим мостам, причем этот путь единственный.

Формат выходных данных

Вывести длину описанного пути, то есть суммарное количество пересечений мостов, которые придется совершить.

Пояснения к примеру



На рисунке вы видите расположение мостов в архипелаге из примера. Прогулка Эйлера будет проходить следующим образом:

1 — 5 — 8 — 2 : 3 моста;

2 — 8 — 5 — 1 — 3 : 4 моста;

3 — 1 — 5 — 4 : 3 моста;

4 — 5 : 1 мост;

5 — 8 — 6 : 2 моста;

6 — 8 — 5 — 7 : 3 моста;

7 — 5 — 8 : 2 моста;

8 — 5 — 1 : 2 моста.

Итого он пройдет через $3 + 4 + 3 + 1 + 2 + 3 + 2 + 2 = 20$ мостов.

Примеры

Пример №1

Стандартный ввод
8 5 8 1 3 8 6 7 5 2 8 1 5 4 5
Стандартный вывод
20

Пример программы-решения

Ниже представлено решение на языке C++.

```
//lca двоичным подъемом, обход в глубину
#include<bits/stdc++.h>

#define pb push_back
#define mp make_pair
#define x first
#define y second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

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

int n, a, b, timer;
vector<vector<int> > G;
const int N = 1e5 + 228;
const int LG = 18;
vector<int> dep, in, out;
```

```
24
25  int up[LG][N];
26
27  void dfs(int a, int p) {
28          in[a] = timer++;
29          up[0][a] = p;
30          for (int i = 1; i < LG; i++) {
31                  up[i][a] = up[i - 1][up[i - 1][a]];
32          }
33          for (int to : G[a]) {
34                  if (to != p) {
35                          dep[to] = dep[a] + 1;
36                          dfs(to, a);
37                  }
38          }
39          out[a] = timer;
40  }
41
42  bool upper(int u, int v) {
43          return in[u] <= in[v] && out[v] <= out[u];
44  }
45
46  int lca(int u, int v) {
47          if (in[u] > in[v]) {
48                  swap(u, v);
49          }
50          if (upper(u, v)) {
51                  return u;
52          }
53          for (int i = LG - 1; i >= 0; i--) {
54                  if (!upper(up[i][u], v)) {
55                          u = up[i][u];
56                  }
57          }
58          return up[0][u];
59  }
60
61  int dist(int u, int v) {
62          int t = lca(u, v);
63          return dep[u] + dep[v] - 2 * dep[t];
64  }
65
66  signed main(){
67      ios::sync_with_stdio(0), cin.tie(0), cout.tie(0);
68
69      cin >> n;
70      G.resize(n);
71      for0(i, n-1){
72          cin >> a >> b;
73          a--;
74          b--;
75          G[a].pb(b);
76          G[b].pb(a);
77      }
78      dep.resize(n, 0);
79          in.resize(n);
80          out.resize(n);
81          dfs(0, 0);
82
83          int ans = 0;
```

```
84          for0(i, n){
85          ans += dist(i, (i+1)%n);
86          }
87
88          cout<<ans;
89  }
```

Пример программы-решения

Ниже представлено решение на языке C++.

```
1   // lca двоичным подъемом, обход в ширину без глубокой рекурсии
2   #include <bits/stdc++.h>
3
4   #define pb push_back
5   #define mp make_pair
6   #define x first
7   #define y second
8
9   #define all(x) (x).begin(), (x).end()
10  #define sz(a) (int)(a).size()
11  #define for0(i, n) for (int i = 0; i < (int)(n); i++)
12  #define for1(i, n) for (int i = 1; i <= (int)(n); i++)
13  #define int long long
14
15  using namespace std;
16  typedef long double ld;
17  typedef pair<int, int> pii;
18
19  int n, a, b, timer;
20  vector<vector<int> > G;
21  const int N = 1e5 + 228;
22  const int LG = 18;
23  vector<int> dep, in, out;
24
25  int up[LG][N];
26
27  bool upper(int u, int v) {
28          return in[u] <= in[v] && out[v] <= out[u];
29  }
30
31  int lca(int u, int v) {
32          if (in[u] > in[v]) {
33                  swap(u, v);
34          }
35          if (upper(u, v)) {
36                  return u;
37          }
38          for (int i = LG - 1; i >= 0; i--) {
39                  if (!upper(up[i][u], v)) {
40                          u = up[i][u];
41                  }
42          }
43
44          return up[0][u];
45  }
46
47  int dist(int u, int v) {
48          int t = lca(u, v);
```

```
49          return dep[u] + dep[v] - 2 * dep[t];
50  }
51
52  signed main(){
53      ios::sync_with_stdio(0), cin.tie(0), cout.tie(0);
54
55      cin >> n;
56      G.resize(n);
57      for0(i, n-1){
58          cin >> a >> b;
59          a--;
60          b--;
61          G[a].pb(b);
62          G[b].pb(a);
63      }
64      dep.resize(n, 0);
65          in.resize(n);
66          out.resize(n, 0);
67
68          deque<int> och;
69          vector<int> par(n), d(n, 1), oo;
70          och.pb(0);
71          oo.pb(0);
72          dep[0] = 0;
73          par[0] = 0;
74          while(sz(och) > 0){
75        int tb = och[0];
76        och.pop_front();
77        for(auto to : G[tb])  if(to != par[tb]){
78          dep[to] = dep[tb] + 1;
79          par[to] = tb;
80
81                  up[0][to] = tb;
82          for (int i = 1; i < LG; i++) {
83                  up[i][to] = up[i - 1][up[i - 1][to]];
84          }
85          och.pb(to);
86          oo.pb(to);
87        }
88
89          }
90          for(int i = sz(oo)-1; i >= 1; i--)
91          d[par[oo[i]]] += d[oo[i]];
92
93      in[0] = 0;
94      for(int i = 1; i < sz(oo); i++){
95          in[oo[i]] = out[par[oo[i]]]+1;
96          out[oo[i]] = in[oo[i]] + 1;
97          out[par[oo[i]]] += 2*d[oo[i]];
98      }
99
100         int ans = 0;
101         for0(i, n){
102         ans += dist(i, (i+1)%n);
103         }
104         cout<<ans;
105 }
```

Третья попытка. Задачи 8–11 класса

Задача I.1.3.1. Послание Аресибо (15 баллов)

Имеется прямоугольное изображение, разбитое на единичные квадратики, размер этого изображения $n \times m$, $(5 \leqslant n, m)$. Каждый его квадратик либо черный либо белый. Известно, что на этом изображении нарисована черным цветом на белом фоне одна четырехсвязная фигура. Фигура называется четырехсвязной, если между любыми двумя ее клетками можно построить путь по клеткам этой фигуры, в котором любые две рядом стоящие клетки являются соседними в изображении либо по горизонтали либо по вертикали. Далее изображение разбили на строки и соединили их в одну большую строку без пробелов и разделителей. Длина этой строки $n \cdot m$. После этого ее отправили в направлении шарового звездного скопления M13, находящегося на расстоянии 25000 световых лет в созвездии Геркулеса. Вы обитатель M13 и перед вами поставили задачу восстановить изображение, исходя из информации о его четырехсвязности. Гарантируется, что решение единственно.

Формат входных данных

На вход подается принятая строка. Она состоит из знаков «.» и «#». Точки соответствуют фону, а решетки — изображенной фигуре. Длина строки не превосходит 5184.

Формат выходных данных

Требуется вывести изображение в исходном виде, то есть разбить его на строки одинаковой длины и расположить их друг под другом.

Примеры

Пример №1

Стандартный ввод
...........###...#..#.#...#..###...#...#....#########..####..#. .####..#..#..#..#.##..##..

Стандартный вывод
....###... #..#.#... #..###... #...#.... ######### ..####..# ..####..# ..#..#..# .##..##..

Пример программы-решения

Ниже представлено решение на языке C++.

```cpp
//dfs, нахождение 4-связной области

#include<bits/stdc++.h>

#define pb push_back
#define mp make_pair
#define x first
#define y second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

using namespace std;
typedef pair<int, int> pii;
typedef long double ld;
typedef vector<string> vs;

int k;
int dx[4] = {-1, 0, 1, 0};
int dy[4] = {0, 1, 0, -1};

bool in_tr(int x, int y, int h, int w){
    return (x >= 0 && y >= 0 && x < h && y < w);
}

void dfs(vector<string> &v, int x, int y){
    v[x][y] = '#';
    k++;
    for0(i, 4){
        int nx = x + dx[i];
        int ny = y + dy[i];

        if(in_tr(nx, ny, sz(v), sz(v[0])) && v[nx][ny] == '$')
            dfs(v, nx, ny);
    }
}

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

    string iss;
    cin >> iss;

    int n = sz(iss);
    for0(i, n) if(iss[i] == '#') iss[i] = '$';
    int kk = 0;
    for(auto q : iss)
        kk += (q == '$');

    for(int i = 5; i < n; i++)
    if(n % i == 0 && n/i >= 5){
        string s = iss;
        vector<string> v;
        for0(j, n/i){
            v.pb(s.substr(0, i));
          if(sz(s) > i)  s = s.substr(i);
        }
```

```
61
62          k = 0;
63          int sx, sy;
64          for0(i, sz(v))
65              for0(j, sz(v[i]))
66                  if(v[i][j] == '$'){
67                      sx = i;
68                      sy = j;
69                  }
70
71          dfs(v, sx, sy);
72
73          if(k == kk)
74          for0(i, sz(v)) cout<<v[i]<<endl;
75      }
76  }
```

Задача I.1.3.2. Гравитационный параллакс (10 баллов)

Специалисты фирмы Amazing Star Travel очень ответственно относятся к безопасности своих клиентов. Прежде чем маршрут из точки A в точку B будет одобрен для увлекательных путешествий, он проходит всестороннюю экспертизу. Помимо очевидных опасностей, таких как наличие на маршруте вредных космических излучений, активность пиратов и прогнозирования вспышек сверхновых, в числе прочих анализируется и множество других, более скучных. Современные исследования показали, что на пролетающий на околосветовых скоростях по отрезку AB космический корабль негативное влияние может оказать гравитационное поле звезд и планет. С удалением звезды или планеты от отрезка, это влияние сначала возрастает, а потом скачкообразно падает до нуля. Особенно опасно, если звезда и планета расположены по разные стороны прямой AB. Тогда их воздействие мультиплицируется, то есть перемножается. Особо следует отметить, что эффект мультиплицирования возникает только в паре звезда-планета, а, например, для пар звезда-звезда или планета-планета не обнаружен. Кроме того, этот эффект не наблюдается, если звезда и планета расположены по одну сторону от прямой AB. Осталось сказать, что можно считать эффект гравитационного воздействия равным площади треугольника ABC, где A и B — начало и конец маршрута, а C — место расположения звезды или планеты. Вас просят оценить опасность заданного отрезка AB, то есть найти максимальную величину гравитационного воздействия на отрезок среди всех пар звезда-планета, расположенных по разные стороны прямой AB. Не стоит забывать и о простом воздействии отдельных объектов, которое может оказаться даже больше, чем воздействие пары.

Формат входных данных

В первой строке содержится четыре целых числа через пробел XA, YA, XB, YB — координаты точек A и B. Во второй строке содержатся числа N и M, разделенные пробелом ($0 \leqslant N, M \leqslant 100$) — количество звезд и количество планет соответственно. В каждой из следующих N строк содержатся координаты очередной звезды. Далее в каждой из следующих M строк содержатся координаты очередной планеты. Указанные звезды и планеты, и только они могут оказать ненулевое гравитационное влияние на отрезок AB. Все координаты целые, по модулю не превосходят 1000.

Гарантируется, что никакие три точки из всех вышеперечисленных не находятся на одной прямой.

Формат выходных данных

В ответе нужно выдать ответ на задачу — одно вещественное число округленное ровно до двух знаков после точки. Разделитель между целой и дробной частями ответа — точка.

Пояснения к примеру

Итоговое наибольшее воздействие на отрезок AB оказывают выделенные треугольниками звезда и планета. Воздействие звезды равно площади треугольника ABC, которая равна 11, воздействие планеты равно площади треугольника ABD, которая равна 14. Так как они находятся по разные стороны прямой AB, то их общее воздействие равно произведению $11 \cdot 14 = 154,00$

Примеры

Пример №1

Стандартный ввод
-4 -2 2 0 3 4 -2 4 2 -1 1 -4 -3 3 -1 0 -2 -2 1 -1

Стандартный вывод
154.00

Пример программы-решения

Ниже представлено решение на языке C++.

```
1  //разделение - знаки векторного произведения, площадь - модуль векторного
2  // произведения
3  #include<bits/stdc++.h>
4
5  #define pb push_back
6  #define mp make_pair
7  #define x first
8  #define y second
9
10 #define all(x) (x).begin(), (x).end()
11 #define sz(a) (int)(a).size()
12 #define for0(i, n) for (int i = 0; i < (int)(n); i++)
13 #define for1(i, n) for (int i = 1; i <= (int)(n); i++)
14 #define int long long
15
```

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

int vprod(pii a, pii b){
    return a.x*b.y - b.x*a.y;
}

int sgn_vprod(pii a, pii b){
    int vp = vprod(a, b);
    if(vp < 0) return -1;
    if(vp > 0) return 1;
    return 0;
}

int sqt2(pii a, pii b, pii c){
    pii v1 = {b.x - a.x, b.y - a.y};
    pii v2 = {c.x - a.x, c.y - a.y};

    return abs(vprod(v1, v2));
}

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

    pii A, B;
    cin >> A.x >> A.y >> B.x >> B.y;
    pii AB = {B.x - A.x, B.y - A.y};

    int n, m;
    cin >> n >> m;
    vector<pii> s(n), p(m);
    for0(i, n) cin >> s[i].x >> s[i].y;
    for0(i, m) cin >> p[i].x >> p[i].y;

    double ans = 0;

    for0(i, n)
    ans = max(ans, sqt2(A, B, s[i]) / 2.0);

    for0(j, m)
    ans = max(ans, sqt2(A, B, p[j]) / 2.0);

    for0(i, n){
        pii Bs = {s[i].x - B.x, s[i].y - B.y};

        for0(j, m){
            pii Bp = {p[j].x - B.x, p[j].y - B.y};

            if(sgn_vprod(AB, Bs) != sgn_vprod(AB, Bp)){
                ans = max(ans, sqt2(A, B, s[i]) * sqt2(A, B, p[j]) / 4.0);
            }
        }
    }

    cout.precision(2);
    cout<<fixed<<ans;
}
```

Пример программы-решения

Ниже представлено решение на языке C++.

```
//разделение - знаки подстановки в общее уравнение прямой, площадь --- формула
// Герона
#include<bits/stdc++.h>

#define pb push_back
#define mp make_pair
#define x first
#define y second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

using namespace std;
typedef long double ld;
typedef pair<double, double> pii;
pii A, B;

struct line{
    double A, B, C;
};

line L(pii a, pii b){
    line r;
    r.A = b.y - a.y;
    r.B = a.x - b.x;
    r.C = b.x*a.y - a.x*b.y;

    return r;
}

double dist(pii a, pii b){
    return sqrt((a.x - b.x)*(a.x - b.x)+(a.y - b.y)*(a.y - b.y));
}


double sqt2(pii a, pii b, pii c){
    double p = (dist(a, b) + dist(b, c) + dist(a,c)) / 2.0;
    return 2*(sqrt(p * (p-dist(a,b)) * (p-dist(b,c)) * (p-dist(a,c)) ));
}

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

    cin >> A.x >> A.y >> B.x >> B.y;

    int n, m;
    cin >> n >> m;
    vector<pii> s(n), p(m);
    for0(i, n) cin >> s[i].x >> s[i].y;
    for0(i, m) cin >> p[i].x >> p[i].y;

    double ans = 0;
    line AB = L(A, B);
```

```
57
58      for0(i, n)
59      ans = max(ans, sqt2(A, B, s[i]) / 2.0);
60
61      for0(j, m)
62      ans = max(ans, sqt2(A, B, p[j]) / 2.0);
63
64      for0(i, n){
65          pii Bs = {s[i].x - B.x, s[i].y - B.y};
66          double stoAB = AB.A*s[i].x + AB.B*s[i].y + AB.C;
67
68          for0(j, m){
69              pii Bp = {p[j].x - B.x, p[j].y - B.y};
70
71              double ptoAB = AB.A*p[j].x + AB.B*p[j].y + AB.C;
72
73
74              if(stoAB < 0 && ptoAB > 0 || stoAB > 0 && ptoAB < 0){
75                  ans = max(ans, sqt2(A, B, s[i]) * sqt2(A, B, p[j]) / 4.0);
76              }
77          }
78      }
79
80      cout.precision(2);
81      cout<<fixed<<ans;
82  }
```

Задача I.1.3.3. Послание внеземного разума 3 (25 баллов)

Профессор Персиков снова на первых полосах новостных агрегаторов! Он сделал очередное гениальное предположение по поводу новой последовательности сигналов из глубин космоса.

- во-первых, говорит профессор, для подтверждения искусственности происхождения сигнала достаточно, чтобы он был периодическим с периодом, **не превосходящим известную** «константу Персикова» P.
- во-вторых, согласно закону диффузного рассеивания информации в некогерентных пространствах при нелинейно возрастающем коэффициенте Заальшютца – Персикова, качество сигнала падает при росте времени его передачи, что означает, что некоторое окончание последовательности можно отбросить как недостоверно опознанное.

Таким образом, все что осталось профессору — отбросить несколько подряд идущих сигналов из конца последовательности так, чтобы она стала периодической с периодом, не превосходящим P. Как обычно, профессор заинтересован удалить как можно меньше сигналов.

Формат входных данных

В первой строке содержится число P — константа Персикова ($1 \leqslant P \leqslant 10^5$). В следующей строке содержится непустая последовательность, состоящая из малых букв латиницы — послание из космоса. Длина этой строки не превосходит $2 \cdot 10^5$.

Формат выходных данных

Вывести одно число — минимальное количество символов, которые нужно удалить из конца последовательности, чтобы она стала периодической с периодом T, не превосходящим P. Последовательность имеет период T, если для любых двух ее символов, расстояние между которыми кратно T верно, что они совпадают.

Пояснения к примеру

Рассмотрим исходную последовательность **abcabcaabcabcabab**. Если из нее ничего не удалять, то ее наименьший период будет равен 15. Если удалить из ее конца одну букву, то период **abcabcaabcabcaba** не изменится, если же удалить две буквы, период **abcabcaabcabcab** станет равен 10. Если удалить три буквы, период abcabcaabcabca равен 7. И этого достаточно для первого теста. Чтобы период не превосходил 3, придется удалить из конца 10 символов и получить **abcabca** с периодом 3. Ну а период 2 можно получить только у начала **ab**, для этого придется удалить 15 символов.

Примеры

Пример №1

Стандартный ввод
8 abcabcaabcabcabab
Стандартный вывод
3

Пример программы-решения

Ниже представлено решение на языке C++.

```
1  //префикс-функция, метод Кнута-Морриса-Пратта
2
3  #include<bits/stdc++.h>
4
5  #define pb push_back
6  #define mp make_pair
7  #define x first
8  #define y second
9
10 #define all(x) (x).begin(), (x).end()
11 #define sz(a) (int)(a).size()
12 #define for0(i, n) for (int i = 0; i < (int)(n); i++)
13 #define for1(i, n) for (int i = 1; i <= (int)(n); i++)
14 #define int long long
15
16 using namespace std;
17 typedef pair<int, int> pii;
18 typedef long double ld;
19 typedef vector<string> vs;
20
21 vector<int> pf (string s) {
22         int n = sz(s);
```

```
        vector<int> pi(n);
        for (int i=1; i<n; ++i) {
                int j = pi[i-1];
                while (j > 0 && s[i] != s[j])
                        j = pi[j-1];
                if (s[i] == s[j])  ++j;
                pi[i] = j;
        }
        return pi;
}

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

    int p;
    string s;

    cin >> p >> s;

    vector<int> kmp = pf(s);

    for(int i = sz(kmp)-1; i >= 0; i--)
        if(i - kmp[i] + 1 <= p)
        return cout<<sz(s)-1 - i, 0;
}
```

Задача I.1.3.4. Проект «Ровные дороги» 2 (30 баллов)

При проектировании новой автодороги было принято решение сделать ее не более чем из двух абсолютно горизонтальных участков. Будущую трассу разбили на n равных по длине малых отрезков. Будем считать, что в пределах одного малого отрезка местность имеет одну и ту же высоту h_i. При этом в целях эффективной трансформации местности требуется для выравнивания использовать исключительно грунт с этой же трассы. Это означает, что можно с некоторого малого отрезка высоты h_i взять некоторое количество грунта d так, что высота этого участка станет $h_i - d > 0$. Далее эти d единиц грунта обязательно нужно переместить на другой малый отрезок высоты h_j так, что его высота станет $h_j + d$. Перемещать грунт можно только в пределах одного из двух выбранных участков, то есть отрезки номер i и номер j должны принадлежать одному и тому же горизонтальному после выравнивания участку. В данной версии задачи высоты выравниваемых участков могут быть любыми положительными, в том числе и не целыми числами. Следующим важным вопросом при строительстве являются трудозатраты. По этой причине требуется выбрать такое разбиение трассы ровно на два непустых участка, чтобы суммарный объем перемещенного грунта был минимально возможным.

Формат входных данных

В первой строке содержится число n — количество малых отрезков, на которые разбили трассу, $2 \leqslant n \leqslant 2 \cdot 10^5$. Во второй строке указаны высоты h_i этих отрезков через пробел в порядке слева направо, $1 \leqslant h_i \leqslant 2 \cdot 10^5$.

Формат выходных данных

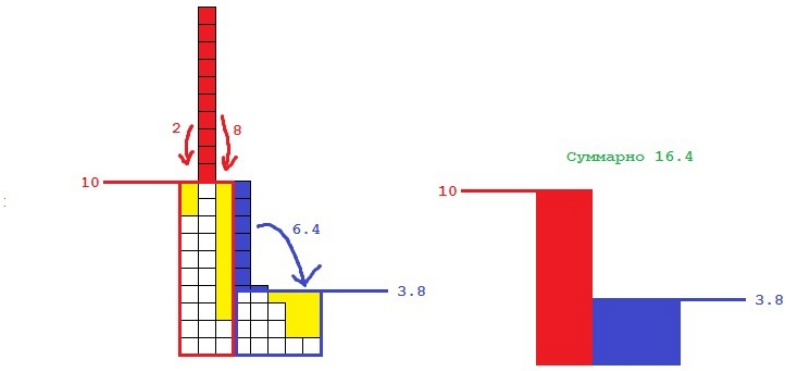
Вывести два ненулевых числа a и b через пробел. Их сумма должна равняться n. Отрезки с номерами с первого по a-й включительно будут принадлежать первому выровненному участку, отрезки с номерами от $a+1$ до n будут принадлежать второму выровненному участку. При этом суммарный объем грунта, перемещенного для такого выравнивания, должен быть минимальным среди всех возможных разбиений трассы на два участка. Если минимальных вариантов несколько вывести тот, у которого число a меньше.

Пояснения к примеру

На следующих рисунках приведены рассуждения для трех вариантов разбиения трассы на два участка,

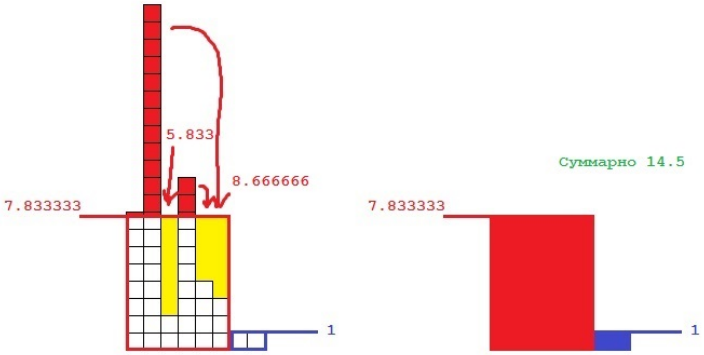
$a = 3, b = 5$:

```
a = 3     b = 5
8 20 2 10 4 3 1 1
(8 + 20 + 2) / 3 = 10
на левом  участке будет перемещено 10 единиц

(10 + 4 + 3 + 1 + 1) / 5 = 3.8
на правом участке будет перемещено 6.4 единиц

Всего будет перемещено 10 + 6.4 = 16.4 единиц
```



$a = 6, b = 2$:

```
a = 6       b = 2
8 20 2 10 4 3 1 1
(8 + 20 + 2 + 10 + 4 + 3) / 6 = 7.8333333333
на левом участке будет перемещено 14.5 единиц
(1 + 1) / 2 = 1
на правом участке будет перемещено 0 единиц
Всего будет перемещено 14.5 + 0 = 14.5 единиц
```



$a = 4, b = 4$:

a = 4 b = 4

8 20 2 10 4 3 1 1

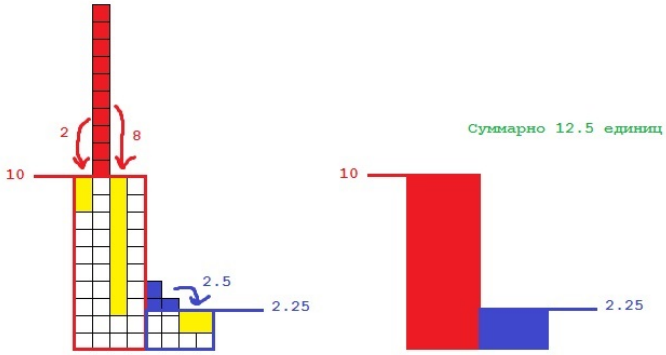
(8 + 20 + 2 + 10) / 4 = 10

на левом участке будет перемещено 10 единиц

(4 + 3 + 1 + 1) / 4 = 2.25

на правом участке будет перемещено 2.5 единиц

Всего будет перемещено 10 + 2.5 = 12.5 единиц



Если перебрать все варианты разбиения трассы на две части, самым оптимальным будет разбиение 4 4 с результатом 12.5.

Примеры

Пример №1

Стандартный ввод
8 8 20 2 10 4 3 1 1
Стандартный вывод
4 4

Пример программы-решения

Ниже представлено решение на языке C++.

```
1  //два дерева отрезков на сумму
2
3  #include<bits/stdc++.h>
4
5  #define pb push_back
6  #define mp make_pair
7  #define x first
8  #define y second
9
10 #define all(x) (x).begin(), (x).end()
```

```cpp
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

using namespace std;
typedef pair<int, int> pii;
typedef long double ld;
typedef vector<string> vs;

int n, ans = 0, N = 262144;
vector <int> kl, su;
double eps = 1e-7;

int get(int l, int r, vector<int> &t){
    int res=0;
    for(l+=N, r+=N; l<=r; l=(l+1)>>1, r=(r-1)>>1){
        if(l & 1)
            res += t[l];
        if(!(r & 1))
            res += t[r];
    }
    return res;
}
void upd(int pos, int val, vector<int> &t)
{
    pos += N;
    t[pos] += val;
    for(pos >>= 1; pos; pos >>= 1)
    {
        t[pos] = t[pos * 2] + t[pos * 2 + 1];
    }
}

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

    cout.precision(3);

    cin >> n;
    vector<int> h(n);
    vector<double> ld(n), rd(n), lsa(n), rsa(n);

    for0(i, n)
        cin >> h[i];

    kl.resize(2*N, 0);
    su.resize(2*N, 0);
    double sum = 0;
    for0(i, n){
        sum += h[i];
        lsa[i] = sum / (i+1);
    }

    for0(i, n){
        upd(h[i], 1, kl);
        upd(h[i], h[i], su);

        int gr = lsa[i] + 1 + eps;
        int k = get(gr, N-1, kl);
```

```
71          double sb = get(gr, N-1, su);
72
73          ld[i] = sb - k * lsa[i];
74      }
75
76      kl.resize(0);
77      su.resize(0);
78      kl.resize(2*N, 0);
79      su.resize(2*N, 0);
80      sum = 0;
81      for(int i = n-1; i >= 0; i--){
82          sum += h[i];
83          rsa[i] = sum / (n - i);
84      }
85
86      for(int i = n-1; i >= 0; i--){
87          upd(h[i], 1, kl);
88          upd(h[i], h[i], su);
89
90          int gr = rsa[i] + 1 + eps;
91          int k = get(gr, N-1, kl);
92          double sb = get(gr, N-1, su);
93
94          rd[i] = sb - k * rsa[i];
95      }
96
97      double mn = 1e18;
98      int a, b;
99      for(int i = 1; i < n; i++)
100         if(ld[i-1] + rd[i] < mn) {
101             mn = ld[i-1] + rd[i];
102             a = i;
103             b = n - i;
104         }
105     cout<<a<<' '<<b;
106 }
```

Задача I.1.3.5. Прогулка по мостам 2 (20 баллов)

Возможно вы знаете историю про то, как Эйлер гулял по мостам Кенигсберга. Допустим теперь, что Эйлер попал на некий архипелаг, между некоторыми островами которого имеются мосты. Мосты построены таким образом, что между любыми двумя островами архипелага можно построить путь по этим мостам, причем этот путь единственный. Все острова пронумерованы от 1 до n. Эйлер находится на острове номер 1 и желает совершить следующую прогулку: с острова номер 1 он хочет дойти до острова номер 2, далее до острова номер 3 и так далее до острова номер n. С него он хочет вернуться на остров номер 1. Он прекрасно понимает, что при этом ему придется посещать некоторые мосты по многу раз. У каждого моста есть смотритель. Одного из них заинтересовало сколько раз по его мосту пройдет Эйлер.

Формат входных данных

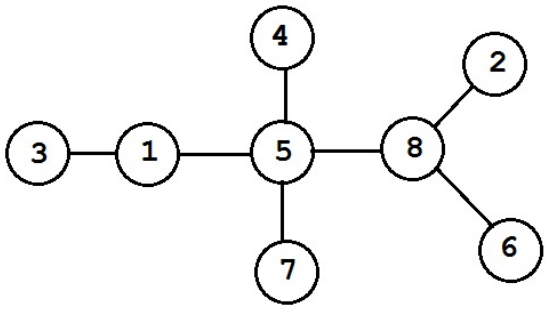
В первой строке содержится два числа через пробел: n — количество островов в архипелаге, $2 \leqslant n \leqslant 10^5$ и b — номер моста, для которого нужно узнать, сколько раз по нему проходит указанный маршрут.В следующих $n - 1$ строках содер-

жатся по два числа a_i и b_i через пробел — номера островов, соединенных мостом, $1 \leqslant a_i, b_i \leqslant n, a_i \neq b_i$. Мосты нумеруются в порядке появления во входных данных, начиная с 1. Гарантируется, что между любыми двумя островами архипелага можно построить путь по этим мостам, причем этот путь единственный.

Формат выходных данных

Вывести одно число — ответ на задачу.

Пояснения к примеру



На рисунке вы видите расположение мостов в архипелаге из примера. Прогулка Эйлера будет проходить следующим образом:

1 — 5 — 8 — **2** — 8 — 5 — 1 — **3** — 1 — 5 — **4** — **5** — 8 — **6** — 8 — 5 — **7** — 5 — **8** — 5 — **1**.

Интересующий нас мост номер 1 соединяет острова 5 и 8. Маршрут проходит по нему 6 раз.

Примеры

Пример №1

Стандартный ввод
8 1 5 8 1 3 8 6 7 5 2 8 1 5 4 5
Стандартный вывод
6

Пример программы-решения

Ниже представлено решение на языке C++.

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

#define pb push_back
#define mp make_pair
#define x first
#define y second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

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

int n, z, a, b, q1, q2;
vector<vector<int> > G;
vector<set<int> > rz(2);

void dfs(int a, int p, int dol){
    rz[dol].insert(a);
    for(auto to : G[a]) if(to != p) dfs(to, a, dol);
}

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

    cin >> n >> z;
    z--;
    G.resize(n);
    for0(i, n-1){
        cin >> a >> b;
        a--;
        b--;
       if(i != z) {
        G[a].pb(b);
        G[b].pb(a);
       }
       else{
        q1 = a;
        q2 = b;
       }
    }

    dfs(q1, q1, 0);
    dfs(q2, q2, 1);

    vector<int> rzb;
    for(auto q : rz[0]) rzb.pb(q);

    int ans = 0;
    for0(i, sz(rzb)-1){
        if(rzb[i]+1 != rzb[i+1]) ans++;
    }
    if(!(rzb.back() == n-1 && rzb[0] == 0)) ans++;
    cout<<2*ans;
}
```

Пример программы-решения

Ниже представлено решение на языке C++.

```cpp
//обход в ширину, без глубокой рекурсии
#include<bits/stdc++.h>

#define pb push_back
#define mp make_pair
#define x first
#define y second

#define all(x) (x).begin(), (x).end()
#define sz(a) (int)(a).size()
#define for0(i, n) for (int i = 0; i < (int)(n); i++)
#define for1(i, n) for (int i = 1; i <= (int)(n); i++)
#define int long long

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

int n, z, a, b, q1, q2;
vector<vector<int> > G;
vector<int> rz;
int mn = 1e9, aa, bb;

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

    cin >> n >> z;
    z--;

    G.resize(n+1);
    for0(i, n-1){
        cin >> a >> b;

        if(i != z){
        G[a].pb(b);
        G[b].pb(a);
        }
        else{
            q1 = a;
            q2 = b;
        }
    }

    vector<int> och;
    vector<int> mark(n+1, -1);
    och.pb(q1);
    mark[q1] = q1;
    int b = 0;

    while(b < sz(och)){
            int v = och[b];
        for(auto to : G[v]) if(mark[to] == -1){
            och.pb(to);
            mark[to] = v;
        }
        b++;
```

```
    }

    sort(all(och));

       int ans = 0;
    for0(i, sz(och)-1){
        if(och[i]+1 != och[i+1]) ans++;
    }
    if(!(och.back() == n && och[0] == 1)) ans++;

    cout<<2*ans;
}
```

Четвертая попытка. Задачи 8–11 класса

Задача I.1.4.1. Скобки (10 баллов)

Задана строка, в которой могут быть встречены 3 типа скобок: фигурные, квадратные и круглые. Помимо скобок в строке встречаются и другие последовательности символов. Вложенность скобок может быть произвольной. Необходимо проверить корректность скобочной записи: каждой открывающей скобке должна соответствовать следующая за ней закрывающая скобка того же типа на том же уровне вложенности, не должно быть открывающей или закрывающей скобки без пары.

Формат входных данных

Строка, содержащая произвольный набор символов (в т. ч. и без скобок).

Формат выходных данных

- Слово «correct», если запись корректна или не содержит скобок.
- Слово «incorrect», если запись не корректна.

Примеры

Пример №1

Стандартный ввод
(this [is] test)
Стандартный вывод
correct

Пример программы-решения

Ниже представлено решение на языке Python3.

```
def balancedBrackets(expr):
    stack = []
    for char in expr:
        if char in ['(','{','[']:
```

```
            stack.append(char)
        elif char in [')','}', ']']:
            if(len(stack) == 0):
                return False
            top = stack.pop()
            if(top == '(' and char != ')'):
                return ""
            if(top == '{' and char != '}' ):
                return False
            if(top == '[' and char != ']'):
                return False
        else:
            continue
    if(len(stack) > 0):
        return False
    return True

expr = input()
print("correct") if balancedBrackets(expr) else print("incorrect")
```

Задача I.1.4.2. Рисунок (15 баллов)

Задан рисунок, состоящий из пронумерованных точек и линий между ними. Напишите программу, которая скажет, можно ли нарисовать этот рисунок (провести ручку по всем точкам и линиям), не отрывая руки и при этом не проводя одну линию дважды?

Формат входных данных

Первая строка содержит число N — количество точек (число от 0 до 1000 включительно).

Вторая строка содержит число M — количество линий (число от 0 до 1000 включительно).

Далее идет M строк, каждая из которых содержит пары номеров точек, соединенных линиями (нумерация точек с 1, между парой точек может быть проведено несколько линий).

Формат выходных данных

- «Yes», если рисунок можно нарисовать в соответствии с условием.
- «No», если нет.

Примеры

Пример №1

Стандартный ввод
3 2 1 2 2 3

Стандартный вывод
Yes

Пример программы-решения

Ниже представлено решение на языке Python3.

```
def task1():
    vert_num = int(input())
    edge_num = int(input())
    verts = [{"watched": False} for _ in range(vert_num)]
    edges = []
    for _ in range(edge_num):
        edges.append(tuple(int(i)-1 for i in input().split()))

    if vert_num == 0:
        return 'Yes'
    line = [0, 0]
    verts[0]['watched'] = True
    flag = True
    while flag:
        flag = False
        for edge in edges:
            if edge[0] in line or edge[1] in line:
                if edge[0] in line:
                    verts[edge[1]]['watched'] = True
                    if edge[0] == line[0]:
                        line[0] = edge[1]
                    else:
                        line[1] = edge[1]
                else:
                    verts[edge[0]]['watched'] = True
                    if edge[1] == line[0]:
                        line[0] = edge[0]
                    else:
                        line[1] = edge[0]
                edges.remove(edge)
                flag = True
    for v in verts:
        if not v['watched']:
            return 'No'
    if len(edges):
        return 'No'
    return 'Yes'

print(task1())
```

Задача I.1.4.3. Часовой механизм (20 баллов)

К вам обратился владелец часовой мастерской, которая делает часы с прозрачным корпусом. В мастерской есть одна проблема: дизайнеры придумывают то, как будут выглядеть часы, но не задумываются о том, как шестеренки будут крутиться. Поэтому, когда дизайн получают мастера, им приходится проверять работоспособность нарисованного дизайнерами механизма — проверять не заклинит ли механизм от сцепки двух крутящихся в одном направлении шестеренок.

Напишите программу, которая будет делать эту работу за мастеров.

Формат входных данных

Первая строка содержит количество шестеренок N (число от 1 до 1000 включительно).

Вторая строка содержит количество связей между шестеренками M (число от 0 до 1000 включительно), которые сцеплены между собой (если одна из них крутится по часовой стрелке, вторая должна крутиться против часовой и наоборот).

Далее идет M строк с парами чисел, являющихся номерами сцепленных между собой шестеренок (нумерация начинается с 1).

Формат выходных данных

Программа должна вывести одно из 2 слов: «good», если механизм работоспособен или «bad», если механизм заклинит.

Примеры

Пример №1

Стандартный ввод
4 4 1 2 2 3 3 4 4 1

Стандартный вывод
good

Пример программы-решения

Ниже представлено решение на языке Python3.

```
def get_unwatched(verts):
    i = 0
    for vert in verts:
        if not vert['watched']:
            vert['orient'] = 1
            return vert

```

```
def task1():
    vert_num = int(input())
    edges = int(input())
    verts = [{"edges": [], "orient": 0, "watched": False} for _ in range(vert_num)]
    for _ in range(edges):
        a, b = (int(i)-1 for i in input().split())
        verts[a]["edges"].append(b)
        verts[b]["edges"].append(a)

    stack = []
    while vert_num > 0:
        if len(stack) == 0:
            stack.append(get_unwatched(verts))
        vert = stack.pop()
        vert_num -= 1
        vert["watched"] = True
        for next_v in vert['edges']:
            if not verts[next_v]["watched"] and verts[next_v] not in stack:
                stack.append(verts[next_v])
            if verts[next_v]["orient"] != 0 and verts[next_v]["orient"] +
            ↪  vert['orient'] != 3:
                return 'bad'
            elif verts[next_v]["orient"] == 0:
                verts[next_v]["orient"] = 3 - vert['orient']
    return 'good'

print(task1())
```

Задача I.1.4.4. Парковка (20 баллов)

На парковке у бизнес-центра N контрольно-пропускных пунктов(КПП). Известно, что каждый день через каждый день работающие в центре сотрудники въезжают на парковку M раз. Каждый раз сотрудник въезжает в определенное время через один заранее известный КПП и выезжает в определенное время через другой (не обязательно совпадающий с первым). На въезде сотрудник получает пропуск, который он должен отдать на выезде. Один и тот же пропуск может быть использован разными сотрудниками, если они находятся на территории парковки в разное время.

Определите минимальное количество пропусков, которое должно быть суммарно на всех КПП к началу дня, чтобы хватило всем сотрудникам.

Если время въезда и выезда разных сотрудников на одном КПП совпадает, считать, что они могут воспользоваться одним пропуском

Формат входных данных

Первая строка содержит число N (число от 0 до 1000 включительно).

Вторая строка содержит число M (число от 0 до 1000 включительно).

Далее следует M строк, содержащих следующую информацию:

Время въезда (число от 6 до 22 включительно).

Время выезда (число от 7 до 23 включительно).

Номер КПП при въезде (нумерация с 1).

Номер КПП при выезде (нумерация с 1).

Формат выходных данных

Программа должна вывести количество пропусков необходимое суммарно на всех КПП к началу дня.

Примеры

Пример №1

Стандартный ввод
2 2 6 12 1 2 12 22 2 2

Стандартный вывод
1

Пример программы-решения

Ниже представлено решение на языке Python3.

```
def solve():
    required = 0
    kpps_count = int(input())
    kpps = [0] * kpps_count
    n_notes = int(input())
    visitors = []
    for _ in range(n_notes):
        entry_hour, out_hour,in_kpp,out_kpp = [int(readed) for readed in
        ↪  input().split()]
        visitors.append((entry_hour,in_kpp - 1,True))
        visitors.append((out_hour,out_kpp - 1,False))

    visitors.sort(key = lambda visitor: (visitor[0],visitor[2]))

    for visitor in visitors:
        if visitor[2]:
            if not kpps[visitor[1]]:
                required += 1
            else:
                kpps[visitor[1]] -= 1
        else:
            kpps[visitor[1]] += 1
    print(required)

solve()
```

Задача I.1.4.5. Обучаем Терминатора (35 баллов)

Перед отправкой Терминатора T-800 в прошлое для спасения Джона Коннора (события 2 части) обнаружилось, что при анализе текстовых документов, OCR-модуль машины допускает ошибки при чтении символов в записи моделей терми-

наторов. Времени на повторное обучение нейронной сети нет, поэтому было принято решение написать hot-fix на символы «T», «0», «1», «8» и «-». При чтении Терминатор каждый символ переводит в матрицу 10 на 10 точек, где 1 означает наличие заполнения, а 0 — отсутствие.

Символы распознаются следующим образом:

- «T» — два прямоугольника лежащих друг на друге, левая граница верхнего прямоугольника левее нижнего, правая граница верхнего прямоугольника правее нижнего.
- «0» — заполненный прямоугольник с прямоугольным вырезом внутри, границы выреза не лежат на сторонах внешнего прямоугольника.
- «8» — заполненный прямоугольник с двумя прямоугольными вырезами внутри, границы вырезов не лежат на сторонах внешнего прямоугольника, границы вырезов не пересекаются, нижняя граница одного выреза выше другого.
- «1» — заполненный прямоугольник, ширина прямоугольника строго меньше его длины.
- «-» — заполненный прямоугольник, ширина прямоугольника строго больше его длины.

Необходимо, чтоб остальные комбинации интерпретировались символом «X».

Напишите программу для решения поставленной задачи.

Формат входных данных

10 строк состоящих из 10 символов «0» или «1».

Формат выходных данных

Один из символов «T», «0», «1», «8», «-» или «X».

Примеры

Пример №1

Стандартный ввод
0000000000 0001110000 0001010000 0001010000 0001110000 0001110000 0001010000 0001010000 0001110000 0000000000

Стандартный вывод
8

Пример программы-решения

Ниже представлено решение на языке Python3.

```python
def get_borders(l, sym=1):
    a, b = None, None
    for i in range(len(l)):
        if sym in l[i]:
            a = i
            break
    for i in range(len(l)-1, -1, -1):
        if sym in l[i]:
            b = i
            break
    return a, b

def get_lr(l, sym=1):
    left, right = None, None
    for i in range(len(l)):
        if l[i] == sym:
            left = i
            break
    for i in range(len(l)-1, -1, -1):
        if l[i] == sym:
            right = i
            break
    return left, right

def get_rectangle(l, b):
    result = []
    for i in range(b[0], b[2]+1):
        tmp = []
        for j in range(b[1], b[3]+1):
            tmp.append(l[i][j])
        result.append(tmp)
    return result

def is_rectangle(l, sym=1):
    up, bottom = get_borders(l, sym)
    if up is None:
        return None
    left, right = get_lr(l[up], sym)
    for i in range(up, bottom+1):
        if (left, right) != get_lr(l[i], sym):
            return None
    return (up, left, bottom, right)

def task5():
    rows = []
    for _ in range(10):
        rows.append([int(i) for i in input()])
    a = is_rectangle(rows)
    if a:
        tmp = get_rectangle(rows, a)
        if is_rectangle(tmp, 0):
            return '0'
        for i in range(len(tmp)):
            if is_rectangle(tmp[:i], 0) and is_rectangle(tmp[i:], 0):
                return '8'
        if a[2]-a[0] > a[3]-a[1]:
```

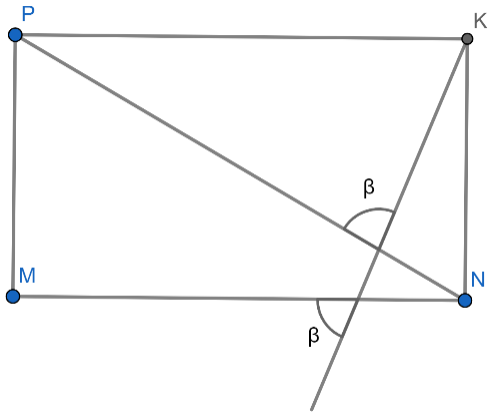
```
            return '1'
        if a[2]-a[0] < a[3]-a[1]:
            return '-'
    else:
        up, bottom = get_borders(rows)
        for i in range(up, bottom+1):
            a = is_rectangle(rows[:i])
            b = is_rectangle(rows[i:])
            if a and b and a[1] < b[1] and a[3] > b[3]:
                return 'T'
    return 'X'

print(task5())
```

Задачи первого этапа. Математика

Первая попытка. Задачи 8–9 класса

Задача I.2.1.1. (20 баллов)

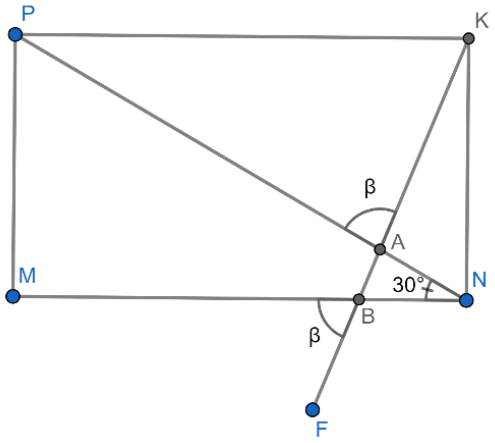
Сторона PM прямоугольника $MNKP$ равна 6, диагональ PN равна 12. Найдите величину угла β. Ответ запишите в градусах.



Решение

В прямоугольном треугольнике PMN катет PM вдвое меньше гипотенузы PN поэтому $\angle PNM = 30°$.

В треугольнике ABN углы A и B равны β как вертикальные с углами KAP и MCF соответственно, а угол N равен 30°. По теореме о сумме углов треугольника получаем: $2\beta + 30° = 180°$, откуда $\beta = 75°$.



Ответ: 75.

Задача I.2.1.2. (20 баллов)

Пираты Меткий Джек, Капитан Флинт и Длинноногий Бенни, приплыв на остров, стали обмениваться добычей. Сначала Джек дал Флинту и Бенни столько слитков золота, сколько было у каждого из них. Затем Флинт дал Джеку и Бенни столько, сколько стало у каждого из них. И наконец, Бенни дал Джеку и Флинту столько, сколько у каждого из них к этому моменту имелось. В результате у всех оказалось по 80 слитков. Сколько слитков было у каждого в начале? В ответе напишите одно число, состоящее из первоначальных количеств, без пробелов в порядке: Меткий Джек, Капитан Флинт, Длинноногий Бенни.

Решение

Будем рассматривать тройку количеств слитков (Джек; Флинт; Бенни). После всех обменов эта тройка (80; 80; 80). При последнем обмене два первых числа удвоились. Значит, тройка была (40; 40; 160). Она получилась при обмене, когда удвоились первое и третье число. Значит, была (20; 140; 80). Эта тройка возникла при удвоении второго и третьего числа. Получаем первоначальную тройку (130; 70; 40).

Ответ: 1307040.

Задача I.2.1.3. (20 баллов)

На кружок по робототехнике ходят ученики 6, 7, 8 и 9 классов (их возраст 12, 13, 14 и 15 лет соответственно). Известно, что шестиклассников меньше, чем семиклассников, а семиклассников меньше, чем восьмиклассников. При этом девятиклассников в кружке столько же, сколько семи- и восьмиклассников вместе взятых, а количества шести- и семимиклассников нечетны. На одном из занятий кружка руководитель с удивлением обнаружил, что средний возраст ребят — целое число лет. Сколько учащихся посещают занятия кружка, если их меньше 30?

Решение

Пусть a, b, c, d — количество шести-, семи- восьми- и девятиклассников соответственно, n лет — средний возраст ребят, тогда:

$$\frac{12a + 13b + 14c + 15d}{a + b + c + d} = n$$

или $(12 - n)a + (13 - n)b + (14 - n)c + (15 - n)d = 0$. С учетом условия $d = b + c$ получаем $(12 - n)a + (28 - 2n)b + (29 - 2n)c = 0$. Перепишем уравнение в виде $(69 - 5n)a + (57 - 4a)(b - a) + (29 - 2n)(c - b) = 0$. Поскольку $a < b < c$, при $n \leqslant 13$ левая часть уравнения положительна, а при $n \geqslant 13$ — отрицательна. Поэтому единственное возможное значение $n = 13$, следовательно, $c = 2a$.

По условию a и b нечетны. Рассмотрим возможные значения a:

a. Случай $a = 1$ невозможен, т. к. тогда $c = 2$ и условие $a < b < c$ невыполнимо.
b. Если $a = 3$, то $c = 6$ Единственное возможное нечетное значение $b = 5$. При этом $d = 11$, и общее количество ребят равно 25. (Легко убедиться, что все условия задачи выполнены).
c. Если $a \geqslant 5$, то $b \geqslant 7$, $c \geqslant 10$, $d = b + c \geqslant 17$. Общее количество участников кружка $a + b + c + d \geqslant 39$, и условия задачи не выполняются.

Таким образом, единственное решение задачи соответствует случаю b.

Ответ: 25.

Задача I.2.1.4. (20 баллов)

У Ани есть квадратный трехчлен $A(x) = x^2 + 3x - 6$, а у Бори — $B(x) = x^2 - 11x + 22$. Каждый из ребят загадал натуральное число, a и b соответственно. Оказалось, что $A(a) = B(b)$. Найдите наибольшее возможное значение $|a - b|$.

Решение

По условию $a^2 + 3a - 6 = b^2 - 11b + 22$ или $a^2 + b^2 + 3a + 11b - 28 = 0$. Раскладывая левую часть на множители, получаем: $(a + b - 4)(a - b + 7) = 0$. Отсюда $a + b = 4$ или $a - b = -7$. В первом случае имеем три пары чисел $(1; 3)$ $(2; 2)$ $(3; 1)$, при этом $|a - b|$ принимает значения 0 или 2. Во втором случае $|a - b| = 7$, и это наибольшее возможное значение.

Ответ: 7.

Задача I.2.1.5. (20 баллов)

У Саши есть сундучок с карточками, на которых записаны все четырехзначные числа, не содержащие в своей записи цифры 0 и 1 (на каждой карточке записано одно число). Сергей случайным образом достает одну карточку. Какова вероятность, что на этой карточке написано число, в котором цифры различны и расположены в порядке убывания, начиная с разряда тысяч?

Ответ запишите в виде десятичной дроби с тремя верными цифрами после запятой.

Решение

Всего карточек у Саши $8^4 = 4096$. Количество чисел, в которых цифры расположены в порядке убывания C_8^4, так как каждой неупорядоченной группе различных цифр соответствует ровно одно число, в котором эти цифры расположены в порядке убывания, их количество 70. Таким образом, искомая вероятность равна $\frac{70}{4096} = 0,0178984375$.

Ответ: 0,017.

Первая попытка. Задачи 10–11 класса

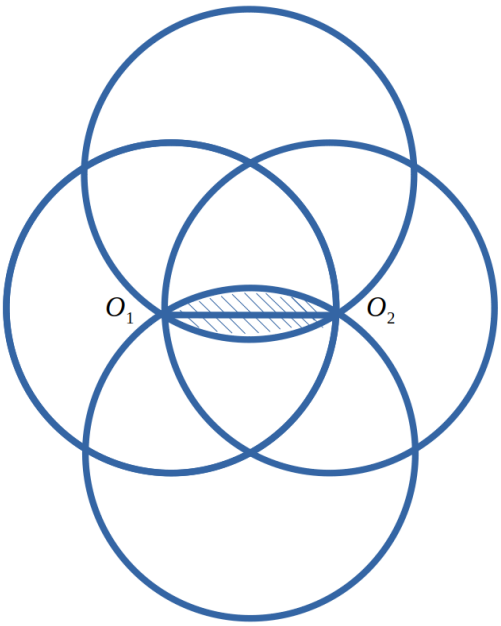
Задача I.2.2.1. (15 баллов)

Точность плоттеров (графических построителей) определяется минимально возможным значением приращения координаты. Гоша сконструировал плоттер из доступных ему деталей и решил определять его точность с помощью следующего приема: построителем строятся две окружности одинакового радиуса так, что каждая проходит через центр другой, а две другие окружности того же радиуса имеют центры в точках пересечения первых двух окружностей. Далее с помощью специального алгоритма, который Гоше передал приятель, в двух последовательных измерениях оценивается площадь области, общей для всех четырех кругов, при этом во втором измерении перо построителя перемещается как раз на искомое минимально возможное значение. Какова точность графопостроителя в миллиметрах, если в первом измерении радиус R окружностей был равен 6 см, а площадь заданной области во втором измерении — 669,78 мм^2. В алгоритме число $\pi = 3,14$, а иррациональные числа и любой результат деления округляются до сотой доли.

- 1 мм.
- 1,5 мм.
- 0,5 мм.
- 2 мм.

Решение

Каждая из двух последних окружностей проходит через центры первых двух (см. рисунок к задаче), поэтому длина из общей хорды будет равна $O_1O_2 = R$.



Искомая площадь будет равна удвоенной площади сегмента с центральным углом 60°, т. е.:b

$$S = 2\left(\frac{\pi R^2}{6} - \frac{R^2\sqrt{3}}{4}\right) = \frac{R^2(2\pi - 3\sqrt{3})}{6}.$$

Подставим в полученное выражение для площади известные значения переменных, учтя, что для второго измерения вместо R нужно писать $R + \Delta R$, и найдем решение:

$$\frac{(R+\Delta R)^2(2\pi - 3\sqrt{3})}{6} = 669,78$$

$$(60+\Delta R)^2 \cdot 0,18 = 669,78$$

$$(60+\Delta R)^2 = 61^2$$

$$\Delta R = 1$$

Ответ: 1 мм.

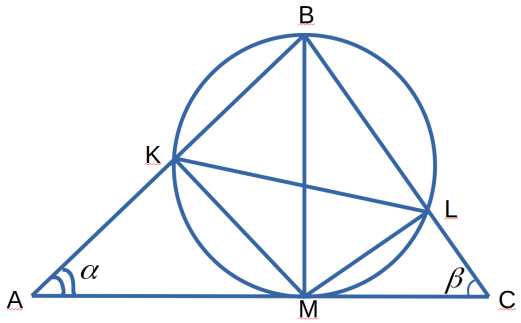
Задача I.2.2.2. (20 баллов)

Программист разработал алгоритм по оценке площади плоских фигур, а также фигур, вписанных в окружность, при этом в качестве результата программа выводила отношение площадей двух выбранных фигур. На этапе тестирования для анализа эффективности на вход алгоритма был треугольник ABC с углами $\angle A = \alpha$ и $\angle B = \beta$, а также высотой BM, на которой была построена окружность. Данная окружность пересекает сторону AB в точке K, а сторону BC — в точке L. Верно ли сработал алгоритм, если при вычислении отношения площади треугольника KLM к площади треугольника ABC для углов $\alpha = 37^o$ и $\beta = 60°$ ответом алгоритма было число 0.194587? Если алгоритм верно произвел вычисление, то до какого знака после запятой (включительно)?

1. Алгоритм выполнил вычисления некорректно.
2. Алгоритм выполнил вычисления корректно с точностью до 1 знака после запятой.
3. Алгоритм выполнил вычисления корректно с точностью до 2 знака после запятой.
4. Алгоритм выполнил вычисления корректно с точностью до 3 знака после запятой.
5. Алгоритм выполнил вычисления корректно с точностью до 4 знака после запятой.

Решение

По условию, $BM \perp AC$, BM — диаметр окружности, $\angle BAC = \alpha$, $\angle BCA = \beta$, K и L — точки пересечения окружности со сторонами AB и BC (см. рис.); требуется найти $S_{\Delta KLM} : S_{\Delta ABC}$.



Пусть $BM = h$; тогда $AM = h\operatorname{ctg}\alpha$, $MC = h\operatorname{ctg}\beta$. Имеем:

$$S_{\Delta ABC} = \frac{1}{2}BM \cdot AC = \frac{1}{2}BM \cdot (AM + MC) = \frac{1}{2}h(h\operatorname{ctg}\alpha + h\operatorname{ctg}\beta) =$$

$$= \frac{1}{2}h^2(\operatorname{ctg}\alpha + \operatorname{ctg}\beta) = \frac{1}{2}h^2\frac{\sin(\alpha+\beta)}{\sin\alpha\sin\beta}$$

Так как $\angle ABM = 90° - \alpha$, $\angle MBC = 90° - \beta$, а также $\angle BKM = \angle BLM = 90^o$ (вписанные углы, опирающиеся на диаметр), то $\angle BMK = \alpha$, $\angle BML = \beta$. Тогда из ΔBKM и ΔBLM находим $KM = h\cos\alpha$, $ML = h\cos\beta$. Значит:

$S_{\Delta KLM} = \frac{1}{2}KM \cdot ML\sin(\alpha+\beta) = \frac{h^2}{2}\cos\alpha\cos\beta\sin(\alpha+\beta)$

Окончательно получим:

$$S_{\Delta KLM} : S_{\Delta ABC} = \frac{0.5h^2\cos\alpha\cos\beta\sin(\alpha+\beta)\sin\alpha\sin\beta}{0.5h^2\sin(\alpha+\beta)} = \frac{1}{4}\sin 2\alpha\sin 2\beta$$

Подставляя данные в условии задачи значения углов α и β, получим окончательно, что $S_{\Delta KLM} : S_{\Delta ABC} = 0,19459461$. Можно заключить, что в данном случае алгоритм сработал корректно, а точность вычислений — до 4-го знака после запятой включительно.

Ответ: 5.

Задача I.2.2.3. (15 баллов)

Путешествуя по джунглям, леди $T.$ забрела на поляну, на которой обитают 15 пауков-птицеедов и 25 древесных лягушек. Какова вероятность того, что после прогулки по этой поляне леди $T.$ снимет с себя двух лягушек и трех пауков? Записать ответ с точностью до тысячных.

Решение

Всего на поляне было $15 + 25 = 40$ насекомых. Общее количество вариантов выбора 5 насекомых из 40 равно:

$$n = C_{40}^{5} = \frac{40!}{5! \cdot 35!} = 12 \cdot 37 \cdot 38 \cdot 39.$$

Количество вариантов, когда возникает событие, благоприятствующее условию задачи и снятию с себя 3 пауков и 2 лягушек, равно:

$$m = C_{15}^{3} \cdot C_{25}^{2} = \frac{15!}{3! \cdot 12!} \cdot \frac{25!}{2! \cdot 23!} = 2 \cdot 13 \cdot 14 \cdot 15 \cdot 25.$$

Тогда вероятность будет равна:

$$p = \frac{m}{n} = \frac{2 \cdot 13 \cdot 14 \cdot 15 \cdot 25}{12 \cdot 37 \cdot 38 \cdot 39} = \frac{5 \cdot 7 \cdot 25}{3 \cdot 37 \cdot 38} = 0,207.$$

Ответ: 0,207.

Задача I.2.2.4. (20 баллов)

Найти количество положительных значений параметра a, при которых неотрицательные значения x, удовлетворяющие уравнению:

$$\cos(21ax + 16x) = \cos(10x - 11ax)$$

И расположенные в порядке возрастания, образуют арифметическую прогрессию.

- Таких решений нет.
- 1 значение параметра a.
- 2 значения параметра a.
- 3 значения параметра a.
- 4 значения параметра a.
- 5 значений параметра a.
- 6 значений параметра a.
- 7 значений параметра a.
- 8 значений параметра a.

Решение

Преобразуем исходное уравнение:

$$2\sin\left(\frac{21ax+16x+10x-11ax}{2}\right)\cdot\sin\left(\frac{21ax+16x-10x+11ax}{2}\right)=0.$$

Тогда:

$$\left[\begin{array}{l}\sin(5ax+13x)=0,\\ \sin(16ax+3x)=0,\end{array}\right. => \left[\begin{array}{l}5ax+13x=\pi k, k\in N,\\ 16ax+3x=\pi n, n\in N,\end{array}\right. => \left[\begin{array}{l}x=\frac{\pi k}{5a+13}, k\in N,\\ x=\frac{\pi n}{16a+3}, n\in N.\end{array}\right.$$

Положительные значения каждого решения (при $k, n \in N$) образуют арифметическую прогрессию с разностями соответственно:

$$d_1=\frac{\pi}{5a+13}, d_2=\frac{\pi}{16a+3}$$

Из членов этих прогрессий можно составить в порядке возрастания одну прогрессию, только когда а) $\frac{d_1}{d_2}=p$ или б)$\frac{d_2}{d_1}=q$ (при $p, q \in N$).

а)$\frac{\pi}{5a+13}:\frac{\pi}{16a+3}=p => a=\frac{13p-3}{16-5p}$.

Так как по условию $a>0$, то, решая неравенство $\frac{13p-3}{16-5p}>0$, получим $\frac{3}{13}<p<\frac{16}{5}$.

Так как p - натуральное число, то $p=\{1;2;3\}$ и, соответственно, $a=\left\{\frac{10}{11};\frac{23}{6};36\right\}$.

б)$\frac{\pi}{16a+3}:\frac{\pi}{5a+13}=q \Rightarrow a=\frac{13-3q}{16q-5} \Rightarrow \frac{5}{16}<p<\frac{13}{3} \Rightarrow q=\{1;2;3:4\}$ и $a=\left\{\frac{10}{11};\frac{7}{27};\frac{4}{43};\frac{1}{59}\right\}$.

Таким образом, всего при 6 значениях параметра a решения уравнения можно выстроить в арифметическую прогрессию.

Ответ: 6 значений параметра а.

Задача I.2.2.5. (30 баллов)

Вычислить модуль тангенса угла при вершине равнобедренного треугольника с наибольшей площадью, при условии, что дана длина медианы, проведенной к боковой стороне данного треугольника. Записать ответ с точностью до сотых.

Решение

Обозначим за a длину боковой стороны треугольника, l — длину медианы, проведенной к боковой стороне a, α — угол между двумя боковыми сторонами. Очевидно, что площадь треугольника $S=0,5a^2\sin\alpha$. Выразим длину медианы через длины боковых сторон через теорему косинусов:

$$l^2=a^2+\left(\frac{a}{2}\right)^2-2a\frac{a}{2}\cos\alpha=a^2\left(\frac{5}{4}-\cos\alpha\right).$$

Из этого выражения выразим боковую сторону треугольника:

$$a^2 = \frac{4l^2}{5 - 4\cos\alpha},$$

откуда выражение для площади треугольника будет переписана следующим образом:

$$S = \frac{2l^2 \sin\alpha}{5 - 4\cos\alpha} = S(\alpha),$$

где $0 < \alpha < \pi$ по смыслу задачи. Из выражения дл площади видно, что при $\alpha \to 0$ и $\alpha \to \pi$ функция $S(\alpha) \to 0$. При этом очевидно и из смысла площади, и из математической функции, что $S(\alpha) > 0$. Если в интервале $(0; \pi)$ функция $S(\alpha)$ имеет критическую точку, то в ней $S(\alpha)$ принимает наибольшее значение. Найдем производную от функции $S(\alpha)$:

$$S'(\alpha) = 2l^2 \frac{5\cos\alpha - 4}{(5 - 4\cos\alpha)^2} = 0,$$

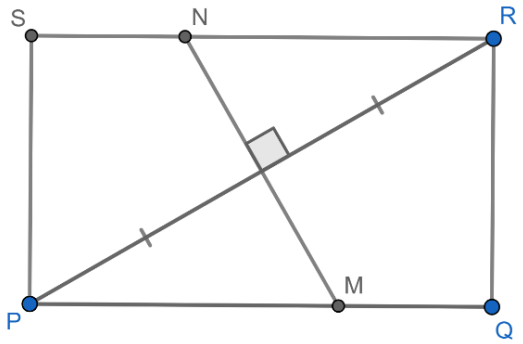
откуда получаем, что наибольшее значение площади будет при $\cos\alpha = 0,8$. Следовательно, $\sin\alpha = 0,6$ и $\tan\alpha = \frac{3}{4}$.

Ответ: 0,75.

Вторая попытка. Задачи 8–9 класса

Задача I.2.3.1. (20 баллов)

Дан прямоугольник $PQRS$ со сторонами $PQ = 16$ и $QR = 12$. Серединный перпендикуляр к PR пересекает стороны PQ и RS в точках M и N соответственно. Найдите длину отрезка MN.



Решение

Пусть Z — середина диагонали PR.

По теореме Пифагора для треугольника PQR: $PQ = \sqrt{PQ^2 + QR^2} = 20$, тогда $PZ = \frac{1}{2}PQ = 10$.

Прямоугольные треугольники RNZ и PMZ равны по катету и прилежащему острому углу: $RZ = PZ$ по условию, а $\angle NRZ = \angle MPZ$ как внутренние накрест лежащие при параллельных прямых SR и PQ и секущей PR. Следовательно, $NZ = MZ = \frac{1}{2}MN$.

Прямоугольные треугольники PZM и PQR подобны по общему острому углу P, поэтому $\frac{PZ}{PQ} = \frac{ZM}{QR}$ откуда $ZM = \frac{PZ \cdot QR}{PQ} = \frac{10 \cdot 12}{16} = \frac{15}{2}$.

Окончательно получаем: $MN = 2 \cdot ZM = 15$.

Ответ: 15.

Задача I.2.3.2. (20 баллов)

Петя поделил натуральное число на 8, 5 и 3 с остатком. Сумма всех остатков оказалась равной 13. Найдите наибольшее возможное трехзначное число, которое делил Петя.

Решение

Остаток от деления меньше делителя, поэтому сумма 13 может получиться, только если остаток от деления на 8 равен 7, от деления на 5 равен 4, при делении на 3 равен 2. Тогда, если к искомому числу прибавить 1, то оно будет делиться на 8, 5 и 3. Поскольку эти числа взаимно простые, то оно будет делиться на их произведение равное 120. Наибольшее такое трехзначное число 960. Искомое число на 1 меньше.

Ответ: 959.

Задача I.2.3.3. (20 баллов)

В начале сентября класс из 25 человек написал сочинение «Как я провел лето». Учитель при проверке выставляет двойную оценку. Например, оценка 4/5 означает, что 4 поставлено за содержание, 5 — за грамотность. Двоек не было! Учеников, имеющих среди своих оценок хотя бы одну 3, было 16, хотя бы одну 4 — 12, хотя бы одну 5 — 5. «Твердых хорошистов» (с оценкой 4/4) было вдвое меньше «упертых троечников» (с оценкой 3/3). Сколько было среди учеников «круглых отличников» (с оценкой 5/5), если известно, что их было нечетное число?

Решение

Пусть x — количество учеников с оценкой 4/4, тогда учеников с оценкой 3/3 — $2x$. Пусть y — количество учеников, получивших 3/4 или 4/3, z — количество учеников с оценками 3/5 или 5/3, u — количество учеников с оценками 4/5 или 5/4, v — количество учеников с оценкой 5/5. Из условия задачи получаем систему уравнений:

$$\begin{cases} 2x + y + z + u + v = 25 \\ 2x + y + z = 16 \\ y + x + u = 12 \\ z + u + v = 5 \end{cases}$$

Выражая все переменные через x получаем:

$$\begin{cases} z = x - 4 \\ y = 20 - 3x \\ u = 2x - 8 \\ v = 17 - 3x \end{cases}$$

Возможны только два значения x: 4 и 5. При допустимом значении x получаем нечетное $v = 5$.

Ответ: 5.

Задача I.2.3.4. (20 баллов)

Сергей вычислил значения приведенного квадратного трехчлена при трех значениях $x = 3$, 14 и 16 и сложил их. Саша вычислил значения другого приведенного квадратного трехчлена при тех же значениях x, сложил их и получил ту же сумму. При каком значении x трехчлены Сергея и Саши равны? (У приведенного многочлена коэффициент при наибольшей степени равен единице).

Решение

Пусть у Саши был многочлен $P(x) = x^2 + ax + b$ а у Сергея $Q(x) = x^2 + cx + d$. По условию:

$$3^2 + 3a + b + 14^2 + 14a + b + 16^2 + 16a + b =$$
$$= 3^2 + 3c + d + 14^2 + 14c + d + 16^2 + 16c + d$$

Отсюда $33a + 3b = 33c + 3d$, то есть:

$$11(a - c) = d - b \tag{I.2.1}$$

Если $a = c$, то и $d = b$, то есть $P(x)$ совпадает с $Q(x)$, что по условию не так. Пусть x_0 — искомое значение, то есть $P(x_0) = Q(x_0)$ или $x_0^2 + ax_0 + b = x_0^2 + cx_0 + d$.

Отсюда, учитывая (I.2.1), получаем $x_0 = \frac{d-b}{a-c} = 11$.

Ответ: 11.

Задача I.2.3.5. (20 баллов)

Если выбрать два произвольных различных числа из множества $\{1, 2, 3, ..., n-1, n\}$, вероятность того, что числа являются последовательными, равна $\frac{1}{31}$. Найдите n.

Решение

Всего способов выбрать два числа из n различных чисел:

$$C_n^2 = \frac{n(n-1)}{2}$$

Количество пар последовательных чисел из n равно $n-1$. Таким образом, данная вероятность:

$$\frac{1}{31} = \frac{n-1}{C_n^2} = \frac{2}{n}$$

Отсюда $n = 62$.

Ответ: 62.

Вторая попытка. Задачи 10–11 класса

Задача I.2.4.1. (15 баллов)

На Земле после внезапного эволюционного изменения зайцы стали плотоядными и начали охотиться на волков, в то время как волки остались плотоядными и не изменили свой рацион.

При наблюдении за одной особью ученый заметил, что данный заяц имеет обыкновение делать запасы, охотясь за двумя волками сразу. Статистика показывает, что нападение начинает всегда заяц и, напав на каждого из двух волков поочередно, с вероятностью $p_1 = 0,6$ ловит волка. Если волк не становится жертвой зайца, то, независимо от судьбы другого волка, начинает сам охотиться на нападающего зайца и с вероятностью $p_2 = 0,81$ ловит зайца-обидчика.

Найти вероятность того, что в очередной схватке двух волков и одного зайца погибнет одно животное, не важно кто. Ответ записать с точностью до десятитысячной.

- 0,1998.
- 0,3058.
- 0,0144.
- 0,2841.

Решение

Существует три случая, когда погибнет одно животное: либо A_1 — гибнет заяц, а волки остаются целыми, либо A_2 — гибнет первый волк, а второй волк и заяц остаются живыми; либо A_3 — второй волк гибнет, а первый волк и заяц выживают. При этом искомая вероятность $P = P(A_1) + P(A_2) + P(A_3)$.

Определим значения каждого из событий:

$$P(A_1) = (1-p_1)^2 \cdot [1-(1-p_2)^2] = 0,154224;$$

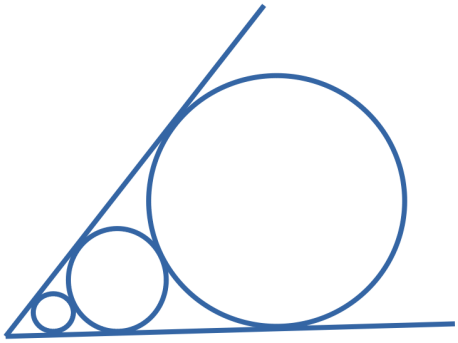
$$P(A_2) = P(A_3) = p_1 \cdot (1-p_1) \cdot (1-p_2) = 0,0456;$$

$$P = (1-p_1)^2 \cdot [1-(1-p_2)^2] + 2p_1 \cdot (1-p_1) \cdot (1-p_2) = 0,2458.$$

Ответ: 0,2458.

Задача I.2.4.2. (15 баллов)

Гигантская сейшельская черепаха Эш решил перебраться поближе к цивилизации и обустроить для своих шестерых черепашат детскую площадку, где они бы могли играть. Для этого на морском дне поблизости от нового дома он нашел две доски и шесть колес разных диаметров от разнообразных агрегатов. Доски он положил так, чтобы двумя концами они упирались друг в друга и образовывали угол между собой в 60°. Внутрь этого угла вложил колеса по возрастанию их радиусов, при этом каждое последующее колесо начиная со второго касается предыдущего и одновременно касается обеих досок. Первое колесо Эш уже заполнил песком, потратив на это 3 минуты. Во сколько раз больше времени ему понадобится, чтобы наносить песок во все 6 колес, если толщина песчаного слоя должна быть во всех колесах одинаковой?



- 63.
- 364.
- 66430.
- 1365.

Решение

Обозначим за v скорость заполнения песком колеса. Тогда, приняв во внимание, что толщина h песчаного слоя во всех колесах одинакова, получим выражение для времени наполнения первого колеса t_1 и всех колес вместе t_{tot}:

$$\begin{cases} t_1 = \frac{hS_1}{v}, \\ t_{tot} = \frac{hS_{tot}}{v}, \end{cases}$$

где S_1 и S_{tot} — площади первого колеса и всех колес вместе, соответственно.

Из этой пары соотношений становится очевидно, что для нахождения отношения времен, необходимо найти отношение площади всех шести колес к площади одного колеса (см. рис. I.2.1):

$$\frac{t_{tot}}{t_1} = \frac{S_{tot}}{S_1},$$

или, очевидно, отношение суммы квадратов радиусов всех окружностей к квадрату радиуса первой окружности. Рассмотрим две соседних окружности, при этом обозначим за R радиус большей окружности.

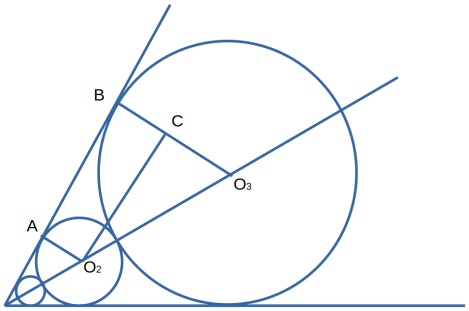


Рис. I.2.1: Чертеж к задаче

Опустим перпендикуляр из центра меньшей окружности на радиус большей окружности, проведенный в точку касания с одной из сторон данного угла. Получим прямоугольный треугольник с гипотенузой $R + r$, катетом $R - r$ и острым углом, равным $30°$, противолежащим этому катету. Тогда $R + r = 2(R - r)$.

Отсюда находим, что $R = 3r$, независимо от того, какая это пара соседних окружностей: первая и вторая, вторая и третья и т. д.. Таким образом, получается, что радиусы окружностей являются геометрической прогрессией со знаменателем, равным, 3. Тогда получится, квадраты радиусов окружностей являются геометрической прогрессией со знаменателем $q = 9$. Тогда сумма квадратов радиусов будет всего лишь суммой 6 членов геометрической прогрессии:

$$\sum_{i=6}^{n} R_i^2 = r_1^2 \frac{q^6 - 1}{q - 1} = 66430 r_1^2.$$

И тогда отношение:

$$\frac{t_{tot}}{t_1} = \frac{S_{tot}}{S_1} = \frac{\pi \sum_{i=6}^{n} R_i^2}{\pi r_1^2} = 66430.$$

Ответ: 66430.

Задача I.2.4.3. (20 баллов)

Найти решение неравенства:

$$\log_{\frac{1}{5}} \log_{11} \left(\frac{\sqrt{x^2+3} + x}{3} \right) < \log_5 \log_{\frac{1}{11}} (\sqrt{x^2+3} - x)$$

Для неправильной дроби с взаимно простыми числителем и знаменателем, являющейся самой левой границей решения, записать в ответ числитель, на единицу больший (например, если решением неравенства является множество решений $(\frac{15}{11}; 5) \bigcup [13; \infty)$, то в ответ нужно будет записать число 16).

Решение

Так как $\sqrt{x^2+3}-x=\frac{3}{\sqrt{x^2+3}+x}$, то:

$$\log_{\frac{1}{11}}(\sqrt{x^2+3}-x)=-\log_{11}(\sqrt{x^2+3}-x)=\log_{11}\left(\frac{\sqrt{x^2+3}+x}{3}\right).$$

С учетом приведенных выше преобразований можно перейти к равносильной системе неравенств:

$$\begin{cases}\frac{\sqrt{x^2+3}+x}{3}>0,\\ \log_{11}\frac{\sqrt{x^2+3}+x}{3}>0,\\ -\log_5\log_{11}\frac{\sqrt{x^2+3}+x}{3}<\log_5\log_{11}\frac{\sqrt{x^2+3}+x}{3},\end{cases}\Rightarrow$$

$$\begin{cases}\frac{\sqrt{x^2+3}+x}{3}>1,\\ \frac{1}{\log_{11}\frac{\sqrt{x^2+3}+x}{3}}<\log_{11}\frac{\sqrt{x^2+3}+x}{3},\end{cases}$$

Воспользовавшись заменой переменных $t=\log_{11}\frac{\sqrt{x^2+3}+x}{3}$, решим нижнее неравенство:

$$\frac{1}{t}<t\Rightarrow\frac{1-t^2}{t}\Rightarrow<\left[\begin{array}{l}-1<t<0,\\ t>1,\end{array}\right.$$

Вернемся к переменной x и запишем итоговую систему неравенств:

$$\begin{cases}\frac{\sqrt{x^2+3}+x}{3}>1,\\ \left[\begin{array}{l}-1<\log_{11}\frac{\sqrt{x^2+3}+x}{3}<0,\\ \log_{11}\frac{\sqrt{x^2+3}+x}{3}>1,\end{array}\right.\end{cases}\Rightarrow\begin{cases}\frac{\sqrt{x^2+3}+x}{3}>1,\\ \log_{11}\frac{\sqrt{x^2+3}+x}{3}>1,\end{cases}\Rightarrow\begin{cases}\frac{\sqrt{x^2+3}+x}{3}>1,\\ \frac{\sqrt{x^2+3}+x}{3}>11,\end{cases}\Rightarrow$$

$$\Rightarrow\frac{\sqrt{x^2+3}+x}{3}>11,\Rightarrow\left[\begin{array}{l}\begin{cases}33-x\geqslant 0,\\ x^2+3>(33-x)^2,\end{cases}\\ \begin{cases}33-x<0,\\ x^2+3\geqslant 0,\end{cases}\end{array}\right.$$

$$\left[\begin{array}{l}\begin{cases}x\leqslant 33,\\ x>\frac{181}{11},\end{cases}\\ x>33\end{array}\right.\quad.$$

Окончательно решение будет выглядеть следующим образом: $x\in(\frac{181}{11},\infty)$. И тогда в ответ запишем число 182.

Ответ: 182.

Задача I.2.4.4. (20 баллов)

Молодой изобретатель решил запустить в космос две ракеты и задал им вектора направления движения: $\vec{v}_1=\left\{4;-8\log_{\frac{1}{2}}t;c\log_{\frac{1}{2}}t\right\}$ и $\vec{v}_2=\left\{5;c;\log_{\frac{1}{2}}t\right\}$, где t — это время. Однако новоявленный Илон Маск забыл значения параметра c, который позволял бы сохранять угол между направлениями запуска после $\frac{1}{16}$ секунды острым, чтобы избежать столкновения ракет в дальнейшем. Помогите отыскать изобретателю набор значений параметра и записать в ответ среднее значение параметра, предполагая, что распределение значения искомого параметра — равномерное. Точность ответа — до тысячной.

Решение

Косинус острого угла положителен, следовательно, скалярное произведение данных векторов $\vec{v}_1 \cdot \vec{v}_2 > 0$ или $20 - 8c\log_{\frac{1}{2}} t + c\log^2_{\frac{1}{2}} t > 0$. Обозначив $\log_{\frac{1}{2}} t = p$, получим $cp^2 - 8cp + 20 > 0$. Причем, так как $x \in [\frac{1}{16}; +\infty)$, неравенство должно выполняться для любых $t \in (0; 4]$.

Геометрическая интерпретация показывает два возможных варианта решения:

а) ветви ее направлены вверх и не пересекает оси t, из-за чего парабола полностью находится над осью t, и при любых значениях переменной t неравенство будет выполняться:

$$\left\{ \begin{array}{l} c > 0, \\ D < 0, \end{array} \right. \Rightarrow \left\{ \begin{array}{l} c > 0, \\ 16c^2 - 20c < 0, \end{array} \right. \Rightarrow \left\{ \begin{array}{l} c > 0, \\ 4c(4c-5) < 0, \end{array} \right. \Rightarrow c \in (0, \tfrac{5}{4}];$$

б) ветви ее направлены вверх, и парабола пересекает ось t, но при $t \in (0; 4]$ ветвь будет находиться выше оси t, а вершина параболы будет находиться левее 4:

$$\left\{ \begin{array}{l} c > 0, \\ D \geqslant 0, \\ f(4) > 0, \\ -\frac{b}{4a} < 4 \end{array} \right. \Rightarrow \left\{ \begin{array}{l} c > 0, \\ 16c^2 - 20c \geqslant 0, \\ f(4) > 0, \\ \frac{8c}{2c} < 4 \end{array} \right. \Rightarrow \left\{ \begin{array}{l} c > 0, \\ 16c^2 - 20c \geqslant 0, \\ f(4) > 0, \\ \frac{4}{<} 4 \end{array} \right. \Rightarrow x \in \emptyset.$$

При $c = 0$ неравенство также выполняется, поэтому мы можем включить значение в промежуток решения $c \in [0, \frac{5}{4}]$. И следовательно, среднее значение параметра будет равняться $\frac{5}{8} = 0,625$.

Ответ: 0,625.

Задача I.2.4.5. (30 баллов)

Потолок в аудитории для проведения пабораторных работ подпирают две металлические абсолютно прямые и очень тонкие балки (длина много больше толщины) из ультрасовременного и прочного сплава. Отношение длин этих балок равно k, а углы, которые балки образуют при этом с полом в аудитории, относятся как 2:3. Найдите наименьшее допустимое значение параметра k и запишите ответ с точностью до сотой.

Решение

Отношение длин отрезков можно найти из выражения для расстояния между параллельными полом и потолком:

$$H = l_1 \sin 3x = l_2 \sin 2x.$$

$$k = \frac{l_2}{l_1} = \frac{\sin 3x}{\sin 2x} = \frac{\sin(2x + x)}{\sin 2x} = \frac{\sin 2x \cos x + \sin x \cos 2x)}{\sin 2x} = 2\cos x - \frac{1}{2\cos x}.$$

Проведем замену $\cos x = t$ и, учитывая, что максимальный угол, который может быть в данной системе, равен $3x = 90 \Rightarrow x \leqslant 30$, следовательно, $t \in [\frac{\sqrt{3}}{2}; 1]$.

Исследование производной функции $k(t) = 2t - \frac{1}{2t}$ ($f'(t) = 2 + \frac{1}{2t^2} > 0$) показывает, что в заданном промежутке значений переменной t функция является растущей и, следовательно, наименьшее значение отношения длин балок k будет соответствовать наименьшему значению параметра t:

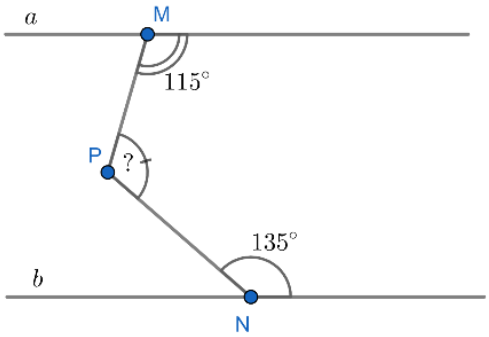
$$k(\frac{\sqrt{3}}{2}) = \sqrt{3} - \frac{1}{\sqrt{3}} = \frac{2\sqrt{3}}{3} = 1,15.$$

Ответ: 1,15.

Третья попытка. Задачи 8–9 класса

Задача I.2.5.1. (20 баллов)

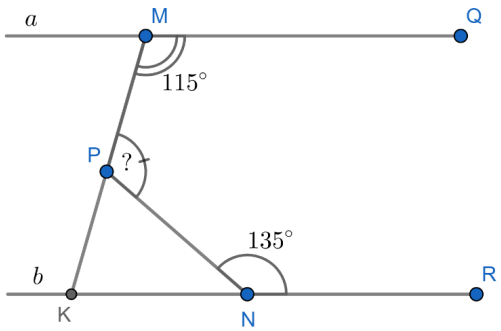
Известно, что прямые a и b параллельны. Найдите величину угла MPN, ответ запишите в градусах.



Решение

Достроим прямую MP до пересечения с прямой b в точке K.

Углы PNR и PNK в сумме дают 180° как смежные, поэтому $\angle PNK = 45°$.



Сумма углов QMP и PKN также составляет 180°, поскольку эти углы — внутренние односторонние при параллельных прямых a и b и секущей MK, отсюда $\angle PKN = 65°$.

По теореме о внешнем угле треугольника:

$$\angle MPN = \angle KNP + \angle NKP = 45^\circ + 65^\circ = 110^\circ.$$

Ответ: 110.

Задача I.2.5.2. (20 баллов)

Сергей поделил натуральное двузначное число на 8 с остатком, а Саша поделил это же число на 5 и получил остаток в 2 раза больше, чем у Сергея. Найдите сумму всех возможных чисел, которые могли делить Сергей и Саша.

Решение

Пусть остаток от деления двузначного числа n у Сергея равен a, тогда остаток Саши — $2a$. Поскольку $2a < 5$ то a может равняться 1 или 2 (случай $a = 0$ невозможен, т. к. второй остаток должен быть больше первого). Разберем эти случаи:

1. $a = 1$. Тогда $n = 8k + 1 = 5m + 2$, где k и m — некоторые натуральные числа. Следовательно, число n имеет вид $40p + 17$ где $p = 0, 1$ или 2. Таким образом, n может принимать значения 17, 57 и 97. Их сумма равна 171.
2. $a = 2$. Тогда $n = 8k + 2 = 5m + 4$, где k и m — некоторые натуральные числа. Следовательно, число n имеет вид $40p + 34$ где $p = 0$ или 1. Таким образом, n может принимать значения 34 и 74. Их сумма равна 108.

Сумма всевозможных значений из обоих случаев составляет $171 + 108 = 279$.

Ответ: 279.

Задача I.2.5.3. (20 баллов)

Спортивный магазин провел акцию «Не имей сто рублей, а имей сто друзей». Она заключалась в следующем: если покупатель, который приобрел велосипед «Дружок», привел 5 друзей, которые приобретали такой же велосипед, то приведшему деньги возвращались. За время проведения акции 25 покупателей пришли сами, остальных привели друзья. Некоторые привели ровно по 5 новых покупателей, а остальные 217 не привели ни одного. Сколько участников акции ездят на велосипеде бесплатно?

Решение

Каждый из x «счастливчиков» привел по 5 друзей. Тогда «приведенных» клиентов было $5x$, а еще 25 пришли сами, значит, всего покупателей было $25 + 5x$. С другой стороны, x человек привели новых клиентов, а 217 — не привели, то есть всего участников акции было $x + 217$. Итак, $25 + 5x = x + 217$ откуда $x = 48$.

Ответ: 48.

Задача I.2.5.4. (20 баллов)

Каждый из графиков функций $y = x^2 + ax + b$ и $y = x^2 + bx + a$ пересекает ось Ox в двух различных точках, а график $y = (x^2 + ax + b)(x^2 + bx + a)$ имеет с осью Ox ровно три общие точки. Найдите наименьшее возможное значение суммы всех координат этих трех точек.

Решение

Координаты по оси Oy точек пересечения с осью Ox равны 0, поэтому достаточно рассмотреть абсциссы точек пересечения, то есть корни многочленов. Из условия следует, что трехчлены $x^2 + ax + b$ и $x^2 + bx + a$ имеют общий корень x_0, а также отличные от него корни x_1 и x_2 соответственно; отсюда, в частности, следует, что $a \neq b$. Общий корень является также корнем разности этих трехчленов, то есть $(a - b)(x_0 - 1) = 0$. Таким образом, $x_0 = 1$. Подставляя этот корень в любой из трехчленов, получаем $1 + a + b = 0$. По теореме Виета $x_0 + x_1 = -a$, $x_0 + x_2 = -b$, откуда $x_0 + x_1 + x_2 = -x_0 - a - b = -(1 + a + b) = 0$.

Ответ: 0.

Задача I.2.5.5. (20 баллов)

Три семьи с детьми (папа, мама и ребенок) отправились в поход. Сколькими способами они могут так построиться для безопасного прохождения болотистого участка, чтобы направляющим и замыкающим были взрослые, и двое детей не шли друг за другом?

Решение

Расставим сначала взрослых. Они могут стать друг за другом 6! способами. При каждом способе есть 5 мест между двумя взрослыми, на которые можно ставить детей. Занумеруем детей как угодно, результат не зависит от того, кого из них мы раньше поставим на место. Место для первого ребенка можно выбрать любым из 5 способов, для второго — любым из 4 оставшихся, для третьего — из 3. Всего $6! \cdot 5 \cdot 4 \cdot 3 = 43200$ способов построения.

Ответ: 43200.

Третья попытка. Задачи 10–11 класса

Задача I.2.6.1. (15 баллов)

Каким должен быть наибольший объем сока, заливаемого в сосуд в виде конуса, если сумма длины образующей и радиуса основания этого сосуда равны 15 см?

Ответ в см^3 записать с точностью до тысячной.

Решение

Обозначим образующую конуса как l, радиус — R, высоту — h. Согласно условию задачи и выражению, связывающему эти три параметра конуса, составим систему уравнений с целью выразить радиус конуса через его высоту:

$$\left\{\begin{array}{l} l+R=15, \\ l^2-R^2=h^2, \end{array}\right. \Rightarrow \left\{\begin{array}{l} l+R=15, \\ (l+R)\cdot(l-R)=h^2, \end{array}\right. \Rightarrow \left\{\begin{array}{l} l+R=15, \\ 15(l-R)=h^2, \end{array}\right.$$

$$\Rightarrow \left\{\begin{array}{l} l+R=15, \\ l-R=\frac{h^2}{15}, \end{array}\right. \text{ откуда } R=\tfrac{225-h^2}{30}.$$

Подставив это выражение для объема конуса, можно получить следующее выражение:

$$V=\frac{1}{3}\pi R^2 h=\frac{\pi}{2700}(h^5-450h^3+50625h).$$

С помощью производной исследуем промежутки знакопостоянства функции объема, памятуя, что $0<h<15$:

$$V'=\frac{\pi}{2700}(5h^4-1350h^2+50625)=0.$$

В пределе $0<h<15$ есть только одна критическая точка $h=3\sqrt{5}$, левее которой производная положительная, а правее — отрицательна. Это говорит о том, что в данной точке есть экстремум — максимум. Тогда $R=\frac{225-h^2}{30}=6$ и объем будет равен $36\sqrt{5}\pi$ или, округленно, $252,893$.

Ответ: 252,893.

Задача I.2.6.2. (15 баллов)

Незадачливый пользователь во время очередной вылазки в интернет нечаянно загрузил себе на компьютер 14 вирусов: 8 рекламных и 6 троянских. Через два часа после окончания интернет-серфинга автоматически должна запуститься антивирусная проверка, которая моментально удаляет все вредоносные программы. Однако за час до вылазки в интернет этот активный пользователь покопался в открытой части кода антивирусника, вследствие чего программа стала удалять случайным образом только некоторое количество вредоносного ПО. Какова вероятность того, что в результате автоматического запуска будут удалены 7 рекламных вирусов и 2 троянских? Ответ записать с точностью до сотой.

Решение

Общее количество вариантов выбора 9 вирусов из 14 равно

$$n=C_{14}^9=\frac{14!}{5!\cdot 9!}=11\cdot 13\cdot 14.$$

Количество вариантов, когда возникает событие, благоприятствующее условию

задачи и удалению с себя 7 рекламных вирусов и 2 троянских, равно:

$$m = C_8^7 \cdot C_6^2 = \frac{8!}{7! \cdot 1!} \cdot \frac{6!}{2! \cdot 4!} = 3 \cdot 5 \cdot 8.$$

Тогда вероятность будет равна:

$$p = \frac{m}{n} = \frac{3 \cdot 5 \cdot 8}{11 \cdot 13 \cdot 14} = \frac{60}{1001} = 0,06.$$

Ответ: 0,06.

Задача I.2.6.3. (30 баллов)

На окружность нанесли 2020 меток и поставили около каждой из них натуральные числа от 1 до 2020. После этого начали последовательно вычеркивать каждое второе число, т. е. числа $2, 4, 6, 8, ..., 2018, 2020, 3, 7$ и т. д. до тех пор, пока не осталось одно единственное число. Назовите оставшееся число.

Решение

Задача представляет собой классический вариант задачи Флавия с $n = 2020$ элементами. В общем случае, если $n = 2^k + m = 2^{10} + 996$, где $m < 2^k$, то останется число $f(n) = 2m + 1 = 2 \cdot 996 + 1 = 1993$.

Ответ: 1993.

Задача I.2.6.4. (20 баллов)

Баба Яга в Крещенскую ночь решила погадать на жениха. Но поскольку вся нечисть в это время занята в других гаданиях, Баба Яга решила привлечь Кощея Бессмертного и Лешего. Гадать она решила так: ровно в полночь Кощей начнет рисовать концентрические окружности с центрами в некоторой точке и радиусами, образующими бесконечную геометрическую прогрессию. За выбор координат центра окружностей и параметров прогрессии ответственным она назначила Лешего. Если прямая $3x - 4y + 13 = 0$ коснется одной из заданных окружностей, то баба Яга выйдет замуж, причем через количество веков, равное номеру окружности. Ответьте, выйдет ли замуж Баба Яга, и если да, то через сколько веков. Леший указал центр концентрических окружностей в точке $(5; 3)$, а в прогрессии первый член $\frac{1}{5}$ и знаменатель прогрессии, равный 2.

Если гадание предскажет, что сказочной героине не суждено найти жениха, то в ответ запишите -1.

Решение

Уравнения окружностей:

$$(x - 5)^2 + (y - 3)^2 = R_n^2,$$

где — $R_n^2 = \frac{1}{5} \cdot 2^{n-1}$. Найдем общие точки окружности и прямой:

$$\begin{cases} x^2 + 10x + 25 + y^2 - 6x + 9 = R^2, \\ y = \frac{1}{4}(3x + 13), \end{cases}$$

$$\Rightarrow \begin{cases} y = \frac{1}{4}(3x + 13), \\ x^2 + 10x + 25 + \frac{1}{6}(9x^2 + 78x + 169) - \frac{6}{4}(3x + 13) = R^2, \end{cases}$$

$\Rightarrow 25x^2 - 154x + 401 - 16R^2 = 0$

Чтобы общая точка была одна, необходимо $D = 0$:

$23716 - 100(401 - 16R^2) = 0$ или $1600R^2 = 16384 \Rightarrow R = \frac{16}{5}$.

$\frac{16}{5} = \frac{1}{5} \cdot 2^{n-1} \Rightarrow n = 5$.

Таким образом, можно сказать, что Бабе яге посчастливится выйти замуж через 5 веков.

Ответ: 5.

Задача I.2.6.5. (20 баллов)

Найти произведение всех пар чисел $(x; y)$ для каждой из которых выполняются одновременно два условия:

$$\begin{cases} 2^{|x^2-2x-3|-\log_2 3} = 3^{-y-4}, \\ 4|y| - |y-1| + (y+3)^2 \leqslant 8 \end{cases}$$

Например, если решением задачи будут две пары чисел $(\frac{1}{2}; 3)$ и $(-1; 2)$, то в ответ надо будет записать следующее число: $\frac{1}{2} \cdot 3 \cdot (-1) \cdot 2 = -3$.

Решение

Оценим левую и правую части уравнения системы:

$x^2 - 2x - 3 = (x-1)^2 - 4 \geqslant 4 \Rightarrow |x^2 - 2x - 3| \leqslant 4 \Rightarrow$

$1 \leqslant 2^{|x^2-2x-3|} \leqslant 16 \Rightarrow \frac{1}{3} \leqslant \frac{1}{3} \cdot 2^{|x^2-2x-3|} \leqslant \frac{16}{3}$.

Решение неравенства системы сводится к решению совокупности трех систем.

$$\left[\begin{array}{l} \begin{cases} y \leqslant 0, \\ y^2 + 3y \leqslant 0, \end{cases} \\ \begin{cases} 0 \leqslant y \leqslant 1, \\ y^2 + 11y \leqslant 0, \end{cases} \\ \begin{cases} y \geqslant 1, \\ y^2 + 9y + 2 \leqslant 0, \end{cases} \end{array}\right. \Rightarrow \left[\begin{array}{l} \begin{cases} y \leqslant 0, \\ -3 \leqslant y \leqslant 0, \end{cases} \\ \begin{cases} 0 \leqslant y \leqslant 1, \\ -11 \leqslant y \leqslant 0, \end{cases} \\ \begin{cases} y \geqslant 1, \\ \frac{-9-\sqrt{73}}{2} \leqslant y \leqslant \frac{-9+\sqrt{73}}{2}, \end{cases} \end{array}\right. \Rightarrow \left[\begin{array}{l} \begin{cases} -3 \leqslant y \leqslant 0, \end{cases} \\ \begin{cases} \emptyset \end{cases} \\ \begin{cases} \emptyset \end{cases} \end{array}\right.$$

$\Rightarrow y \in [-3; 0]$

Таким образом правая часть уравнения допускает оценку:

$$\frac{1}{81} \leqslant 3^{-y-4} \leqslant \frac{1}{3}$$

Так как наименьшее значение левой части уравнения системы совпадает с наибольшим значением правой части системы, то имеет место:

$$\left\{\begin{array}{l} 3^{-y-4}=\frac{1}{3}, \\ 2^{|x^2-2x-3|-\log_2 3}=\frac{1}{3}, \end{array}\right. \Rightarrow \left\{\begin{array}{l} -y-4=-1, \\ |x^2-2x-3|=0, \end{array}\right. \Rightarrow \left\{\begin{array}{l} y=-3, \\ \left[\begin{array}{l} x=-1, \\ x=3, \end{array}\right. \end{array}\right. \Rightarrow \left[\begin{array}{l} \left\{\begin{array}{l} y=-3. \\ x=-1. \end{array}\right. \\ \left\{\begin{array}{l} y=-3. \\ x=3. \end{array}\right. \end{array}\right.$$

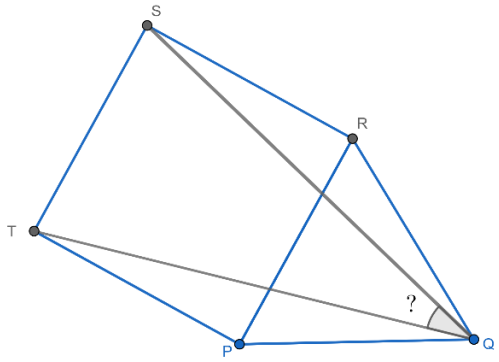
Таким образом произведение всех пар чисел будет равняться -27.

Ответ: -27.

Четвертая попытка. Задачи 8–9 класса

Задача I.2.7.1. (20 баллов)

На стороне PR равностороннего треугольника PQR во внешнюю сторону построен квадрат $PRST$. Найдите величину угла SQT, ответ запишите в градусах.



Решение

Углы TPQ и SRQ равны $90^\circ + 60^\circ = 150^\circ$. Треугольники PQT и SRQ равнобедренные, поэтому $\angle PQT = \angle RQS = \frac{180^\circ - 150^\circ}{2} = 15^\circ$.

Окончательно получаем: $\angle SQT = \angle PQR - \angle RQT - \angle RQS = 60^\circ - 2 \cdot 15^\circ = 30^\circ$.

Ответ: 30.

Задача I.2.7.2. (20 баллов)

Граф Калиостро поделил натуральное двузначное число на 9 с остатком, а Маргадон поделил это же число на 7 и получил остаток в 3 раза больше, чем у Калиостро. Найдите сумму всех возможных чисел, которые могли делить граф Калиостро и Маргадон.

Решение

Пусть остаток от деления двузначного числа n у Калиостро равен a, тогда остаток Маргадона — $3a$. Поскольку $3a < 7$, то а может быть 1 или 2 (случай невозможен, т. к. второй остаток должен быть больше первого). Разберем эти случаи:

1. $a = 1$. Тогда $n = 9k + 1 = 7m + 3$, где k и m — некоторые натуральные числа. Тогда число n имеет вид $63p + 10$, $p = 0, 1$. $n = 10, 73$. Их сумма 83.
2. $a = 2$. Тогда $n = 9k + 2 = 7m + 6$, где k и m — некоторые натуральные числа. Тогда число n имеет вид $63p + 20$, $p = 0, 1$. $n = 20, 83$. Их сумма 103.

Сумма чисел из обоих случаев равна 186.

Ответ: 186.

Задача I.2.7.3. (20 баллов)

У Маши были три корзинки с конфетами. Сначала она переложила из первой корзинки во вторую и третью столько конфет, сколько в каждой из них было изначально. После этого из второй корзинки Маша переложила в третью и первую такое количество конфет, которое в каждой из них имелось на тот момент. Наконец, она переложила из третьей корзинки в первую и вторую столько конфет, сколько там оказалось к этому времени. В результате всех перекладываний в первой корзинке оказалось 160 конфет, во второй 120, а в третьей — 40. Какое количество конфет изначально было во **второй** корзинке?

Решение

Будем записывать количества конфет тройками чисел (первая корзинка; вторая корзинка; третья корзинка). Начнем решать задачу с конца, тогда после третьей операции была тройка $(160; 120; 40)$. Эта тройка образовалась при удвоении числа конфет в первой и второй корзинках, поэтому после второй операции была тройка $(80; 60; 180)$. В свою очередь, эта тройка образовалась при удвоении числа конфет в первой и третьей корзинках; следовательно, после первой операции было $(40; 190; 90)$. Наконец, эта тройка получилась в результате удвоения количества конфет во второй и третьей корзинках. Таким образом, изначально была тройка $(180; 95; 45)$ и во второй корзинке было 95 конфет.

Ответ: 95.

Задача I.2.7.4. (20 баллов)

Буратино вычислил значения приведенного квадратного трехчлена при двух значениях: $x = 2$ и $x = 72$ и сложил их. Пьеро вычислил значения другого приведенного квадратного трехчлена при тех же значениях, сложил их и получил ту же сумму. При каком значении x трехчлены Буратино и Пьеро равны?

У приведенного многочлена коэффициент при наибольшей степени равен единице.

Решение

Пусть у Буратино был многочлен $P(x) = x^2 + ax + b$, а у Пьеро — $Q(x) = x^2 + cx + d$.

По условию $2^2 + 2a + b + 72^2 + 72a + b = 2^2 + 2c + d + 72^2 + 72c + d$.

Откуда $74a + 2b = 74c + 2d$, то есть:

$$37(a - c) = d - b \tag{I.2.2}$$

Если $a = c$, то и $d = b$, то есть $P(x)$ совпадает с $Q(x)$, что по условию не так.

Пусть x_0 — искомое значения, то есть $P(x_0) = Q(x_0)$.

Получаем, что $x_0^2 + ax_0 + b = x_0^2 + cx_0 + d$. Откуда $x_0 = \frac{d-b}{a-c} = 37$, учитывая (I.2.2).

Ответ: 37.

Задача I.2.7.5. (20 баллов)

Если выбрать три произвольных различных числа из множества натуральных чисел $\{1, 2, 3, ..., n-1, n\}$, вероятность того, что числа являются последовательными, равна $\frac{1}{222}$. Найдите n.

Решение

Всего способов выбрать три числа из n различных чисел:

$$C_n^3 = \frac{n(n-1)(n-2)}{6}$$

Количество троек последовательных чисел из n равно $n - 2$. Таким образом, данная вероятность:

$$\frac{1}{222} = \frac{n-2}{C_n^3} = \frac{6}{n(n-1)}.$$

То есть $n = 37$.

Ответ: 37.

Четвертая попытка. Задачи 10–11 класса

Задача I.2.8.1. (15 баллов)

Найти наибольшее значение выражения:

$$\frac{x^2 + 7xy}{x + y} + \frac{y^2 + 7yz}{y + z} + \frac{z^2 + 7zx}{z + x}$$

для положительных значений переменных, сумма которых равняется единице.

Решение

Воспользуемся оценкой, которая выполняется для всех положительных переменных a и b:

$$\frac{2ab}{a+b} \leqslant \frac{a+b}{2}.$$

Преобразуем первое слагаемое в сумме:

$$\frac{x^2+7xy}{x+y} = \frac{x^2+xy+6xy}{x+y} = x + \frac{6xy}{x+y} = x + 3\frac{2xy}{x+y} \leqslant x + 3\frac{x+y}{2} = \frac{5}{2}x + \frac{3}{2}y.$$

По аналогии с предыдущим слагаемым получим:

$$\frac{y^2+7yz}{y+z} = \frac{5}{2}y + \frac{3}{2}z,$$

$$\frac{z^2+7zx}{z+x} = \frac{5}{2}z + \frac{3}{2}x,$$

и тогда сумма полученных слагаемых будет равняться $4(x+y+z) = 4$.

Ответ: 4.

Задача I.2.8.2. (15 баллов)

Согласно правилам некоторой олимпиады по математике, участники, вышедшие в финальный тур, могу получить любое количество призов из призового фонда. Т.е. один финалист, обладающий достаточными знаниями и навыками решения задач, может выиграть и 2, и 4, и, даже, все призы из призового фонда. Сколькими способами будет распределить награды между финалистами, если призовой фонд олимпиады по математике состоит из 6 ноутбуков, а в финал вышли 8 участников? Во скольких из всех способов восьмой участник получит не менее двух ноутбуков? Ответ записать через запятую, без пробелов.

Решение

1. Каждый ноутбук можно разыграть восемью способами, поэтому всего имеется $8^6 = 262144$ способов.
2. Чтобы определить, в скольких случаях 8-ому участнику достанется не менее двух ноутбуков, найдем, сколькими способами можно получить восьмому участнику один ноутбук и не получить ни одного. Если восьмому участнику не досталось ни одного ноутбука, то остальным участникам досталось каждому по 1 ноутбуку, что оценивается в 7^6 способов; если же один из ноутбуков все-таки достался восьмому участнику, то остальным участникам досталось 5 ноутбуков, что можно оценить в $6 \cdot 7^5$ способов распределения наград. В остальных $8^6 - 7^6 - 6 \cdot 7^5$ случаях восьмому участнику достанется не менее двух ноутбуков.

Ответ: 262144,43653.

Задача I.2.8.3. (20 баллов)

Внутри правильного тетраэдра с ребром 10 летает пчела. Каким должен быть ее наименьший замкнутый путь, если пчела должна побывать на всех гранях тетраэдра?

Ответ округлить до тысячной.

Решение

Очевидно, что траектория движения пчелы представляет собой замкнутый пространственный четырехугольника $KLMN$, вершины которого расположены на гранях тетраэдра $ABCD$ (см. чертеж (I.2.2а)).

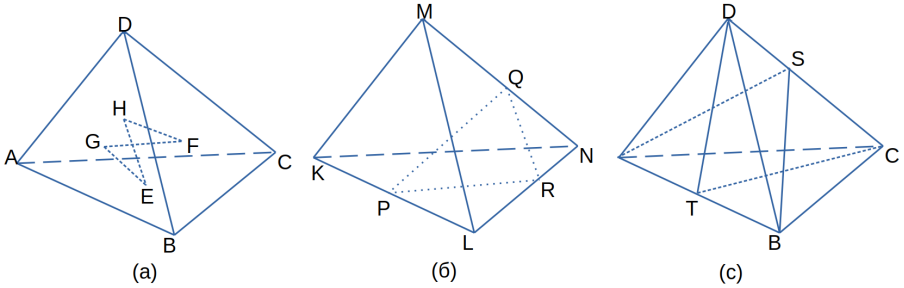


Рис. I.2.2: Чертеж к задаче

Рассмотрим тетраэдр $ABCD$. Пусть пчела побывала на каждой из граней тетраэдра и вернулась в исходную точку. Без ограничения общности можно считать, что она сначала побывала на грани ABC, потом — на грани BCD, затем — на DAB, и, наконец, на ACD. Обозначим соответствующие точки на гранях, в которых побывала муха, через E, F, G и H. Ясно, что минимальное расстояние, которое пчела могла пролететь, равно периметру пространственного четырехугольника $EFGH$.

1. Проведем через DC плоскость, перпендикулярную AB (плоскость симметрии тетраэдра $ABCD$) и рассмотрим четырехугольник $E_1F_1G_1H_1$, симметричный $EFGH$ относительно этой плоскости. (Вершины E_1 и G_1 останутся на тех же гранях, что E и G соответственно, F_1 попадет на одну грань с H, а H_1 — на одну грань с F.) Периметры четырехугольников $EFGH$ и $E_1F_1G_1H_1$ равны.
   Лемма. Рассмотрим любой пространственный четырехугольник $KLMN$ (см. чертеж (I.2.2б)). Пусть P и Q — середины сторон KL и MN. Тогда $PQ \leqslant (KN + LM)$.
   Доказательство. Обозначим через R середину диагонали LN. Имеем $PR = \frac{1}{2}KN$, $RQ = \frac{1}{2}LM$. Таким образом, $PQ < PR + RQ\frac{1}{2}(KN + LM)$.
   Лемма доказана.
2. Обозначим через E_2, F_2, G_2 и H_2 середины отрезков EE_1, FH_1, GG_1 и HF_1 соответственно. Вершины этого четырехугольника тоже лежат на гранях тетраэдра, кроме того, периметр четырехугольника $E_2F_2G_2H_2$ не больше периметра $EFGH$. Кроме того, вершины E_2 и G_2 (середины EE_1 и GG_1) будут лежать в плоскости симметрии тетраэдра, проходящей через CD, т. е. на медианах CT и DT граней ABC и ABD.
   Исходя из четырехугольника $E_2F_2G_2H_2$, точно так же построим $E_3F_3G_3H_3$, симметричный ему относительно плоскости симметрии тетраэдра, проходящей

через AB, а затем, взяв середины отрезков, соединяющих вершины этих четырехугольников, лежащих в одной грани, получим $E_4F_4G_4H_4$, все вершины которого лежат в объединении двух плоскостей симметрии тетраэдра $ABCD$, проходящих через CD и AB. Иными словами, вершины E_4 и G_4 лежат на отрезках CT и DT, а вершины F_4 и H_4 — на медианах AS и BS граней ACD и BCD.

При этом периметр $E_4F_4G_4H_4$ не превосходит периметра $EFGH$. Значит, периметр $EFGH$ не меньше, чем $4d$, где d — расстояние между прямыми CT и BS.

Осталось построить путь длины $4d$ и найти d. Пусть E_0 и F_0 — основания общего перпендикуляра к прямым CT и BS, причем E_0 лежит на CT, а F_0 — на BS. Обозначим через G_0 точку, симметричную точке E_0 относительно плоскости ABS. Из симметрии ясно, что F_0G_0 — общий перпендикуляр к прямым BS и DT. Аналогично строится точка H_0, при этом G_0H_0 и H_0E_0 являются общими перпендикулярами соответственно к DT и AS и к AS и CT. Значит, периметр четырехугольника $E_0F_0G_0H_0$ равен $4d$. Заметим, что нужно еще проверить, что основания этих общих перпендикуляров лежат на гранях тетраэдра, а не на их продолжениях, это будет сделано ниже (нам еще нужно вычислить d).

3. Проведем через AB плоскость, перпендикулярную CT и спроецируем на нее наш тетраэдр. Получим треугольник ABD' (см. чертеж I.2.3), в котором $AB = a$, $DT = \sqrt{\frac{2}{3}}a$ (по формуле для длины высоты правильного тетраэдра).

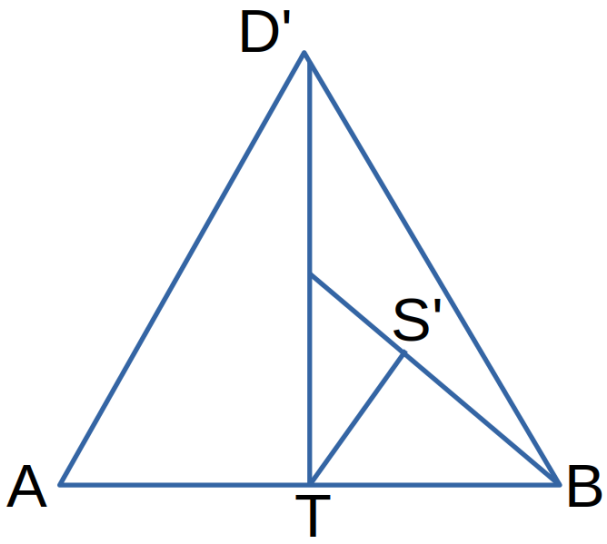


Рис. I.2.3: Чертеж к задаче

Точка S перейдет в S' — середину $D'T$. Искомое расстояние d равно расстоянию от точки T до прямой BS' (поскольку общий перпендикуляр параллелен плоскости проекции). Кроме того, очевидно, что основание перпендикуляра, опущенного из точки T на прямую BS', лежит на отрезке BS', а не на его продолжении, значит, точка F_0 лежит на отрезке BS. Аналогично доказывается, что и остальные вершины четырехугольника лежат на медианах, а не на их продолжениях.

В прямоугольном треугольнике BTS' известны катеты $BT = \frac{a}{2}$, $TS' = \frac{a}{2}\sqrt{\frac{2}{3}}$. Значит, $BS' = \frac{a}{2}\sqrt{\frac{5}{3}}$; $d = \frac{BT \cdot TS'}{BS'} = \frac{a}{\sqrt{10}}$. И тогда искомый периметр равен

$4\frac{a}{\sqrt{10}} = 4\sqrt{10} \approx 12,649.$

Ответ: 12,649.

Задача I.2.8.4. (20 баллов)

В подвале пожилой Дамы хранится 15 корзин с яблоками, причем количество неиспорченных яблок в каждой корзине соответствует номеру этой корзины. Дети, живущие по соседству, решили помочь ей перебрать фрукты, но чтобы работа не была такой скучной, было принято следующее решение: за одно перекладывание добавлять в корзину яблоки можно столько яблок, сколько их уже есть в ней, из любой другой корзины, в которой достаточно для этого яблок. Каким может быть наибольшее количество яблок в одной корзине?

Решение

Всего в корзинах $k = 15(15 + 1)/2 = 120$ яблок. Посмотрим, как происходит операция обратная к досыпанию яблок. Она выглядит так: яблоки в какой-то корзине делят пополам, после чего две половинки или оказываются в отдельных корзинах, или содержимое одной из них добавляют в какую-то не пустую корзину (а другую половинку оставляют в отдельной корзине).

В результате таких операций из корзины с k яблоками можно получить только корзины, в которых $mk/2^S$ яблок при некоторых натуральных m и S (поскольку есть только деление на 2 и сложение). Значит, в таком виде, в частности, должно представляться и число 1, но число 1 не представляется в таком виде, так как k не является степенью двойки.

При четном k в одну корзину нельзя собрать не только k яблок, но и $k-1$ яблок, так как $k-1$ нечетное число, а последняя операция перекладывания могла быть только удвоением.

Таким образом, в одну корзину можно добавить $k - 2 = 118$ яблок. Осталось убедиться, что эти оценки сверху достижимы. Схема алгоритма достижения такова. Имея две корзины с a и b яблоками, где $a+b$ нечетно и $a = 2^m mod(a+b)$ при некотором натуральном m, мы можем получить в этих корзинах после «перекладываний» 1 и $a+b-2$ яблок, затем 2 и $a+b-2$. Далее, воспользовавшись корзинкой с 2 яблоками, «извлечем» еще 2 яблока из остальных корзин и получим в этих двух корзинах 4 и $a+b-2$ яблок и т. д. Таким образом, начав с корзин, в которых 1 и 2 яблока, будем выкладывать из остальных по 2 яблока, пока во всех остальных корзинах в сумме не останется 1 или 0 яблок.

Ответ: 118.

Задача I.2.8.5. (30 баллов)

Астрономическая обсерватория ведет наблюдения за космическими телами, движущимися в некоторой области наблюдаемого космического пространства, и результаты наблюдений за двумя кометами, метеорным потоком и группой метеороидов заносит в карту. Траектории комет представляют из себя две параболы Π_1 и Π_2,

причем траектория второй параболы Π_2 симметрична траектории первой кометы Π_1 с уравнением $y = ax^2$, $a < 0$ относительно точки $N(b; ab^2)$, где $b > 0$. Метеорный поток, движущийся в той же области по прямой, пересекает каждую из траекторий ровно в одной точке: Π_1 — в точке B_1, Π_2 — в точке B_2 так, что угол B_1B_2N — прямой. Группа метеороидов движется по прямой траектории, представляющей из себя касательную к траектории первой параболы, проведенной в точке B_1 и одновременно пересекающей отрезок B_2N в точке L.

Определите отношение параметров $\frac{a}{b}$, если длина отрезка B_1L минимальна, а площадь треугольника B_1B_2N равна $\frac{1}{3}$.

Решение

Известно, что прямые пересекают параболу ровно в одной точке только когда она параллельная оси OY, либо касается этой параболы. Оси парабол параллельны оси OY (см. чертеж I.2.4), поэтому либо $B_1B_2||OY$, либо B_1B_2 — общая касательные к параболам, что невозможно, так как единственная общая касательная данных парабол проходит через точку N. Отсюда $B_1B_2||OY$.

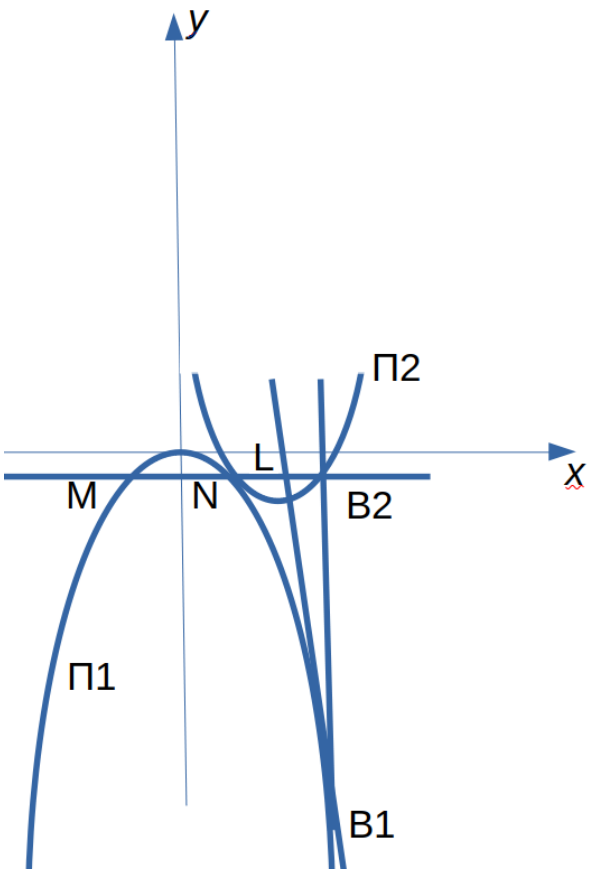


Рис. I.2.4: Чертеж к задаче

Таким образом, $NB_2||OX$, поэтому $MN = 2b_1$ если точка M — точка пересечения NB_2 с Π_1, то и из симметрии парабол следует $NB_2 = 2b$, откуда $B_2(3b; ab)$, $B_1(3b, 9ab^2)$- координаты точек B_2 и $_1$. Тогда касательная B_1L имеет уравнение $y = 6abx - 9ab^2$ и пересекает прямую сNB_2 уравнением $y = ab^2$ в точке $L(\frac{5}{3}b; ab^2)$. Отсюда получаем $NL = \frac{2}{3}b$, $B_2L = \frac{4}{3}b$ т. е. $NL : B_2L = 1 : 2$. Далее из равенства $\frac{1}{3} =$

$S_{B_1B_2N} = \frac{1}{2}NB_2 \cdot B_1B_2 = 8|a|b^2$ следует, что $|a| = \frac{1}{24b^2}$. Поэтому $B_1L^2 = B_1B_2^2 + LB_2^2 = 64a^2b^2 + \frac{16}{9}b^2 = \frac{1}{9}(\frac{1}{b^2} + 16b^2)$. Наименьшее значение этой функции достигается при $b^2 = \frac{1}{4}$, т. е. $b2 = \frac{1}{2}$ и тогда $a = -\frac{1}{3}$. В таком случае отношение параметров $\frac{a}{b} = -1,5$.

Ответ: -1,5.