Заключительный этап

Индивидуальный предметный тур

Информатика. 8–11 класс

Задача III.1.1.1. A plus/minus B (10 баллов)

Вы являетесь членом команды, разрабатывающей интеллектуальный калькулятор. Идея этого калькулятора следующая: пользователь вслух произносит арифметическое выражение, результат которого он хочет узнать, а калькулятор автоматически распознает его речь, вычисляет выражение и формирует голосовое сообщение с результатом вычисления. Часть команды разработчиков уже перевела голосовое сообщение в текстовый вид. Вы ответственный за внутренние вычисления в калькуляторе. На вход вашему блоку подается текстовая запись выражения, на выходе должен быть текстовый ответ для этого выражения. Так как это только прототип, то все вычисления происходят на целых числах в интервале от 0 до 99, а выражение имеет вид либо A plus B либо A minus B.

Формат входных данных

В первой строке содержится количество выражений n, на которые нужно вывести ответ ($1 \leqslant n \leqslant 100$).

В следующих n строках содержатся текстовые записи арифметических выражений. Каждая строка имеет вид либо A plus B либо A minus B, где A и B — текстовая запись двух чисел от 0 до 99 на английском языке. Гарантируется, что результат вычисления каждого такого выражения также число из этого отрезка. Все слова каждого выражения разделены ровно одним пробелом. В записи чисел используются следующие слова:

zero, one, two three, four, five, six, seven, eight, nine,

ten, eleven, twelve, thirteen, fourteen, fifteen, sixteen, seventeen, eighteen, nineteen,

twenty, thirty, forty, fifty, sixty, seventy, eighty, ninety.

Формат выходных данных

Вывести n строк, в каждой ответ на соответствующие выражение в виде текстового сообщения.

Примеры

Пример №1

Стандартный ввод
4 twelve plus forty nine seventy two minus seventeen one minus one twenty three plus sixty eight

Стандартный вывод
sixty one fifty five zero ninety one

Пример программы-решения

Ниже представлено решение на языке C++.

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

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

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

    string S19[20] = {"zero", "one", "two", "three", "four", "five", "six", "seven", "eight", "nine",
                      "ten", "eleven", "twelve", "thirteen", "fourteen", "fifteen", "sixteen", "seventeen", "eighteen", "nineteen"};
    string S90[9] = {"ten", "twenty", "thirty", "forty", "fifty", "sixty", "seventy", "eighty", "ninety"};

    map<string, int> T;

    for(int i = 0; i < 20; i++)
        T[S19[i]] = i;

    for(int i = 0; i < 9; i++)
        T[S90[i]] = (i+1) * 10;

    vector<string> itotxt(100);
    for(auto q : T)
        itotxt[q.second] = q.first;

    int u;
    cin >> u;
    string vs;
    getline(cin, vs);
    while(u--){
        string s;
        getline(cin, s);
```

```
34
35          stringstream ss;
36          ss << s;
37
38          string ds, op;
39          bool frst = 1;
40          int a = 0, b = 0, r;
41          while(ss >> ds){
42              if(ds == "plus" || ds == "minus"){
43                  frst = 0;
44                  op = ds;
45              }
46              else
47                  if(frst)
48                      a += T[ds];
49                  else
50                      b += T[ds];
51          }
52
53          if(op == "plus")
54              r = a + b;
55          else
56              r = a - b;
57
58          if(r >= 20) {
59                  cout<<itotxt[r - r%10];
60                  if(r%10 > 0)
61                      cout<<' '<<itotxt[r%10];
62                  cout<<endl;
63          }
64          else cout<<itotxt[r]<<endl;
65      }
66  }
```

Задача III.1.1.2. Развлечение ИИ (15 баллов)

Один ИИ в целях самосовершенствования генерирует случайные последовательности натуральных чисел и затем для каждой находит самую большую перестановку, которая является непрерывным подотрезком сгенерированной последовательности.

Последовательность A является подотрезком последовательности B, если A может быть получена из B удалением нескольких (возможно, ни одного) элементов из начала и нескольких (возможно, ни одного) элементов из конца.

Перестановкой длины n называется последовательность, содержащая числа от 1 до n в произвольном порядке. Каждое число должно входить в нее ровно один раз.

Требуется проверить результат ИИ. По исходной последовательности нужно найти размер самой большой перестановки, являющейся непрерывным подотрезком заданной последовательности натуральных чисел.

Формат входных данных

В первой строке записано число n — количество чисел в исходной последовательности, $1 \leqslant n \leqslant 1000$. В следующей строке находятся n чисел через пробел — элементы последовательности. Все они в пределах от 1 до n.

Формат выходных данных

Вывести одно число — ответ на задачу. Если ни одной перестановки нет, вывести 0.

Пример №1

Стандартный ввод
19 5 2 6 3 5 1 4 2 1 3 6 2 5 8 1 2 5 4 3
Стандартный вывод
6

Пояснения к примеру

В предложенной последовательности встречаются перестановки 4 2 1 3 и 1 2 5 4 3, но самой большой является перестановка 6 3 5 1 4 2 или 2 6 3 5 1 4 длины 6.

Пример программы-решения

Ниже представлено решение на языке C++.

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

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

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

    int n, ans = 0;
    cin >> n;
    vector<int> v(n);
    for(int i = 0; i < n; i++) cin >> v[i];
    vector<int> mask(n+1, 0);

    for(int i = 0; i < n; i++){
            int t = 0, mx = 0;
        for(int j = i; j < n; j++){
            if(mask[v[j]] != 0) break;
            else{
                t++;
                mask[v[j]] = 1;
                mx = max(mx, v[j]);
                if(mx == t) ans = max(ans, mx);
            }
        }
        for(int j = i; j < n; j++)
            if(mask[v[j]] != 0) mask[v[j]] = 0; else break;
    }
     cout << ans;
}
```

Пример программы-решения

Ниже представлено решение на языке Python.

```
n = int(input())
a = input().split()
ans = 0
for i in range(n):
    a[i] = int(a[i])
for len in range(1, n + 1):
    upd = []
    for i in range(n + 1):
        upd.append(0)
    l = 0
    r = len
    bal = 0
    upper = 0
    for i in range(len):
        if upd[a[i]] == 0:
            bal +=1
        upd[a[i]] += 1
        if a[i] > len:
            upper += 1
    if bal == len and upper == 0:
        ans = len
    for i in range(len, n):
        if upd[a[i]] == 0:
            bal += 1
        if a[i] > len:
            upper += 1
        upd[a[i]] += 1
        if upd[a[i - len]] == 1:
            bal -= 1
        if a[i - len] > len:
            upper -= 1

        upd[a[i - len]] -= 1
        if bal == len and upper == 0:
            ans = len
print(ans)
```

Задача III.1.1.3. Диалог двух ИИ (20 баллов)

Два независимых ИИ ведут диалог. Его запись представляет собой непрерывную строку, в которую без пробелов записаны слова диалога. Известен набор слов, используемых ими при диалоге. В какой-то момент в диалог вмешивается третий ИИ (возможно в самый последний момент, когда запись диалога уже закончена), и диалог первых двух нарушается. Требуется по записи беседы и набору слов, доступному первым двум ИИ определить какое максимальное количество слов могли сказать первые два ИИ до того момента, пока не вмешался третий.

Формат входных данных

В первой строке записана непустая строка, содержащая полную запись беседы трех ИИ, длина этой строки не превосходит 1000. Во второй строке находится число

n — количество слов, используемых первыми двумя ИИ ($1 \leqslant n \leqslant 100$). В следующих n строках содержатся слова, при помощи которых первые два ИИ ведут диалог. И первая строка и слова записаны только символами «0» и «1». Длина каждого слова не превосходит 50, слова могут повторяться.

Формат выходных данных

Вывести одно число — максимальное количество слов, которое могли сказать друг другу первые два ИИ до того момента, пока не вмешался третий.

Примеры

Пример №1

Стандартный ввод
110110110111111011010010 6 110 11011 01111 1101101 11 0110

Стандартный вывод
7

Пояснения к примеру

Начало строки из примера можно многими способами представить в виде последовательности слов (в скобках указано окончание, сказанное третьим участником беседы):

$110 + 11011 + 01111 + 1101101 + (0010)$ — здесь было сказано 4 слова;

$11 + 0110 + 11011 + 11 + 11011 + (010010)$ — здесь было сказано 5 слов;

$110 + 110 + 110 + 11 + 11 + 1101101 + (0010)$ — здесь было сказано 6 слов;

$110 + 110 + 110 + 11 + 11 + 110 + 110 + (10010)$ — здесь было сказано 7 слов, и это максимальное возможное количество.

Пример программы-решения

Ниже представлено решение на языке C++.

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

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

```

```cpp
vector<int> prefix_function (string s) {
    int n = (int) s.length();
    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;
}

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

    string s;
    cin >> s;
    int n, len, ans = 0;

    s = "#" + s;
    len = s.size();
    cin >> n;
    vector<string> v(n);
    for(int i = 0; i < n; i++) cin >> v[i];

    vector<vector<int> > K(len+1);
    for(int i = 0; i < n; i++){
        string z = v[i] + "@" + s;
        vector<int> tkmp = prefix_function(z);

        for(int j = 0; j < z.size(); j++)
            if(tkmp[j] == v[i].size())
                K[j - v[i].size() + 1 - 1 - v[i].size()].push_back(i);
    }

    vector<int> dp(len, -1);
    dp[0] = 0;

    for(int i = 0; i < len - 1; i++)
    if(dp[i] >= 0)
        for(auto q : K[i+1]){
            int tlen = v[q].size();
            dp[i + tlen] = max(dp[i + tlen], dp[i] + 1);
    }

    for(auto q : dp) ans = max(ans, q);
        cout << ans;
}
```

Пример программы-решения

Ниже представлено решение на языке Python.

```python
s = input()
n = int(input())
a = []
dp = []
for i in range(n):
```

```
    a.append(input())
for i in range(len(s) + 1):
    dp.append(-1000000000)
dp[0] = 0
for i in range(1, len(s) + 1):
    for j in range(n):
        if (i - len(a[j]) >= 0):
            ff = s[i - len(a[j]):i]
            if ff == a[j]:
                dp[i] = max(dp[i], dp[i - len(a[j])] + 1)
ans = 0
for i in range(1, len(s) + 1):
    ans = max(dp[i], ans)
print(ans)
```

Задача III.1.1.4. Круги и окружности (25 баллов)

Классической задачей теории ИИ является задача на распознавание изображений. В данном случае вам предлагается выяснить количество кругов и количество окружностей «нарисованных» на картинке. «Рисовать» будем на прямоугольной таблице, состоящей из единичных квадратов. Если единичный квадрат принадлежит какой-то фигуре, то в нем стоит символ «<#» (решетка), иначе символ «<.» (точка).

Формат входных данных

В первой строке заданы размеры изображения n и m. $30 \leqslant n, m \leqslant 1000$. В следующих n строках, каждая длиной m символов, задается само изображение. Диаметры кругов и окружностей могут быть различными, но не меньше 6. Ни одна из фигур не касается границ изображения и других фигур. Линия любой окружности «непрерывна», то есть из любой ее точки можно попасть в любую другую ее точку перемещаясь по «точкам» окружности (смежными о стороне или углу) как по часовой, так и против часовой стрелки. Рекомендуем внимательно вспомнить определение окружности.

Формат выходных данных

Вывести два числа через пробел – сначала количество кругов, затем количество окружностей, изображенных на рисунке.

Примеры

Пример №1

Стандартный ввод

```
30 40
........................................
...#####.......#####.....#######........
..#######.....#.....#...#########.......
.#########....#.....#..###########......
.#########....#.....#..###########......
.#########....#.....#..###########......
.#########....#.....#..###########......
.#########.....#####...###########......
..#######.............############......
...#####...............############......
.......................#########.......
.....#########...........#######........
....##.......##.........................
...#...........#...........#######......
..#.............#........##.......##....
.##.............##......##.........##...
.#...............#......#...........#...
.#...............#.....#.............#..
.#...............#.....#.............#..
.#...............#.....#.............#..
.#...............#.....#.............#..
.#...............#.....#.............#..
.#...............#.....#.............#..
.##.............##.....#.............#..
..#.............#.......#...........#...
...#...........#........##.........##...
....##.......##..........##.......##....
.....#########.............#######......
........................................
........................................
```

Стандартный вывод

```
2 3
```

Пример программы-решения

Ниже представлено решение на языке C++.

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

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

int n, m;
int dx1[4] = {-1, 0, 1, 0}, dy1[4] = {0, -1, 0, 1};
int dx2[8] = {-1, -1, -1, 0, 1, 1, 1, 0}, dy2[8] = {-1, 0, 1, 1, 1, 0, -1, -1};
```

```
vector<string> T;
    vector<vector<int> > G;

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

void no_dfs1(int x, int y, int c){
    vector<pii> och;
    och.push_back({x, y});
    int b = 0;
    G[x][y] = c;
    while(b < och.size()){
        int tx = och[b].first;
        int ty = och[b].second;
        for(int i = 0; i < 4; i++){
            int nx = och[b].first + dx1[i];
            int ny = och[b].second + dy1[i];
            if(in_tr(nx, ny) && T[x][y] == T[nx][ny] && G[nx][ny] == -1){
                G[nx][ny] = c;
                och.push_back({nx, ny});
            }
        }
        b++;
    }
}

void no_dfs2(int x, int y, int c){
    vector<pii> och;
    och.push_back({x, y});
    int b = 0;
    G[x][y] = c;
    while(b < och.size()){
        int tx = och[b].first;
        int ty = och[b].second;
        for(int i = 0; i < 8; i++){
            int nx = och[b].first + dx2[i];
            int ny = och[b].second + dy2[i];
            if(in_tr(nx, ny) && T[x][y] == T[nx][ny] && G[nx][ny] == -1){
                G[nx][ny] = c;
                och.push_back({nx, ny});
            }
        }
        b++;
    }
}

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

    cin >> n >> m;
    T.resize(n);
    for(int i = 0; i < n; i++)
        cin >> T[i];

    G.resize(n, vector<int>(m, -1));

    int k = 0, h1;
    for(int i = 0; i < n; i++)
    for(int j = 0; j < m; j++)
```

```
    if(G[i][j] == -1){
    k++;
    if(T[i][j] == '#'){
            h1++;
        no_dfs2(i, j, k);
    }
    else
        no_dfs1(i, j, k);
    }
    cout<<2 * h1 - k + 1<<' '<<k - 1 - h1<<endl;
}
```

Пример программы-решения

Ниже представлено решение на языке Python.

```
asd = input().split()
dx = [-1, -1, 0, 1, 1, 1, 0, -1]
dy = [0, 1, 1, 1, 0, -1, -1, -1]
n = int(asd[0])
m = int(asd[1])
a = []
ans1 = 0
ans2 = 0
visited = []
for i in range(n + 2):
    ss = []
    ad = []
    for j in range(m + 2):
        ss.append('.')
        ad.append(False)
    a.append(ss)
    visited.append(ad)
for i in range(1, n + 1):
    f = input()
    for j in range(1, m + 1):
        a[i][j] = f[j - 1]
for i in range(1, n + 1):
    for j in range(1, m + 1):
        if a[i][j] == '#' and not visited[i][j]:
            ans = []
            visited[i][j] = True
            Q = []
            Q.append([i, j])
            while len(Q) > 0:
                y = Q[len(Q) - 1][0]
                x = Q[len(Q) - 1][1]
                #print(y, x)
                ans.append([y, x])
                Q.pop()
                for k in range(8):
                    if not visited[y + dy[k]][x + dx[k]] and a[y + dy[k]][x + dx[k]] =='#':
                        Q.append([y + dy[k], x + dx[k]])
                        visited[y + dy[k]][x + dx[k]] = True
            ans.sort()
            prev_y = -1
            ch = True
            for k in range(1, len(ans)):
                if (ans[k][0] != prev_y):
```

```
                    prev_y = ans[k][0]
                elif (ans[k][1] - ans[k - 1][1] != 1):
                    ch = False
            if ch:
                ans1 += 1
            else:
                ans2 += 1
print(ans1, ans2)
```

Задача III.1.1.5. Унификация (30 баллов)

В этой задаче в самой простой форме научимся строить логические выводы. Если более точно, то только предварительную обработку некоторых выражений, называемую унификацией. Метод, для которого строится унификация, называется методом резолюций. Это один из самых давних методов теории ИИ, позволяющий автоматически доказывать правильность логических рассуждений. Договоримся о следующих обозначениях (для краткости изложения, некоторые из них не совсем формализованы):

- буквы a, b, c (и только они) обозначают константы, вместо них ничего подставлять нельзя. Константы соответствуют конкретным объектам рассматриваемой области, о которой мы хотим что-то утверждать (конкретным числам, людям, словам и т. п.).
- буквы x, y, z обозначают переменные, вместо любой переменной можно подставить терм, но нельзя подставить предикат. Вместо переменной нельзя подставлять терм, содержащий эту переменную. Если вместо одной переменной подставлен терм t, то в точности этот же терм должен быть подставлен и вместо всех других переменных с таким же именем в предикате, содержащем эту переменную. Определения термов и предикатов смотрите ниже. Ограничений на количество различных переменных вводить не будем, но гарантируется, что во входных тестах в качестве переменных использованы только эти символы.
- буквы f, g, h (и только они, будем называть их функциональными символами) обозначают термы (функции). Каждый терм имеет местность (количество аргументов), в этой задаче будем считать, что любой терм зависит от нуля, одного или двух аргументов. Изначально аргументы термов — это переменные. Примеры исходных термов (без подстановок) следующие: одноместный терм $f(x)$ (например, «отец объекта x»), двухместный терм $g(x, z)$ (например «число, являющееся суммой чисел x и z»). Терм, после подстановки вместо всех его переменных конкретных констант возвращает какой-то объект из множества рассматриваемых объектов. Помимо функциональных символов, термами считаются константы (нульместные термы) и переменные (одноместные термы).
- буквы P, Q, R (и только они) обозначают предикаты (отношения). Предикаты так же имеют местность. В данной задаче будем ограничиваться рассмотрением только двухместных предикатов. После подстановки вместо всех своих переменных некоторых констант и вычисления значений термов, предикат возвращает логическое значение $TRUE$ или $FALSE$, которое интерпретируется как результат логического утверждения относительно аргументов предиката. Пример исходного предиката (без подстановок): $Q(y, z)$ (например «число y делится нацело на число z»).
- подстановкой вместо переменной (например x) терма (например $f(y)$) в предикат P будем называть результат замены всех вхождений этой переменной

в этом предикате на заданный терм. Напомним, чтобы избежать ненужной рекурсии, подставлять вместо переменной терм, содержащий ту же самую переменную запрещено. Примеры подстановок (в терм и предикат):
  - в терме $f(x, z)$ подставим вместо z константу a, получим $f(x, a)$;
  - в терме $f(x, g(x, h(z)))$ подставим вместо x терм $g(a, f(b, y))$, получим $f(g(a, f(b, y)), g(g(a, f(b, y)), h(z)))$. Мы обязаны заменить все вхождения x на $g(a, f(b, y))$;
  - в предикате $P(f(x), g(x, y))$ подставим вместо x терм $h(y)$, получим $P(f(h(y)), g(h(y), y))$.

  Отметим отдельно, что можно по-очереди производить несколько подстановок. В том числе на следующих шагах можно подставлять вместо переменных, появившихся в результате предыдущих подстановок.
- унификацией двух предикатов P_1 и P_2 называется некоторый последовательный набор подстановок отдельно внутрь P_1 и отдельно внутрь P_2, таких, что P_1 и P_2 посимвольно совпадут. Другими словами, два предиката унифицируются, если найдутся две серии независимых друг от друга подстановок таких, что предикаты совпадут с точностью до символа. Унификацией так же называется и результат применения этих подстановок, то есть предикат, который получается, когда P_1 и P_2 совпадают. Далее будем рассматривать только унифицируемые пары предикатов. Очевидно, что два предиката унифицируемы, если они обозначены одним и тем же предикатным символом.

Пример унификации предикатов:

$$P_1 = P(x, g(y, f(x)))$$
$$P_2 = P(f(y), g(b, f(f(a))))$$

Если мы сразу заменим в P_1 переменную x на терм $f(y)$, то получим $P_1 = P(f(y), g(y, f(f(y))))$ и далее потерпим неудачу, так как переменная y должна быть заменена в одном случае на константу b, а в другом на константу a. Поэтому, действуем немного по-другому. Сначала в P_2 заменим переменную y на другую переменную, например, z. Получим $P_2 = P(f(z), g(b, f(f(a))))$. Так как предикаты меняются независимо, то в предикате P_1 ничего не изменится. Теперь выполним в P_1 замену x на $f(z)$, получим $P_1 = P(f(z), g(y, f(f(z))))$. Осталось заменить в P_1 переменную z на константу a, и потом переменную y на константу b. А в P_2 нужно заменить переменную z на константу a. Получим, что оба предиката примут вид $P(f(a), g(b, f(f(a))))$. Это и будет результат их унификации. Возможен и другой порядок выполнения подстановок, но результат унификации будет тот же.

Ваша задача — написать программу, которая для таких пар предикатов строит их унификацию.

Формат входных данных

В первой строке задан первый предикат P_1, во второй строке задан второй предикат P_2. Правила записи и ограничения описаны выше, запись предикатов не содержит пробелов. Если два терма обозначены одинаковыми функциональными символами, то они имеют одинаковую местность. Гарантируется, что заданные предикаты унифицируемы и результат унификации однозначен и не содержит переменных. Каждый предикат записан не более чем 50 символами. Для понимания деталей формата ввода смотрите пример.

Формат выходных данных

Вывести результат унификации двух исходных предикатов в том же формате.

Примеры

Пример №1

Стандартный ввод
P(x,g(y,f(x))) P(f(y),g(b,f(f(a))))

Стандартный вывод
P(f(a),g(b,f(f(a))))

Пример программы-решения

Ниже представлено решение на языке C++.

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

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

string P1, P2;
int t1 = 99, t2 = 499;
vector<int> mask(1000, -1);

string tos(int a){
    stringstream ss;
    string r;
    ss << a;
    ss >> r;
    return r;
}

int toa(string s){
    stringstream ss;
    int r;
    ss << s;
    ss >> r;
    return r;
}

string termf(string P, int pos){
    if(P[pos] <= 'h' && P[pos] >= 'f'){
    int bal = 1, t = pos + 2;
    while(bal > 0){
        if(P[t] == '(') bal++;
        if(P[t] == ')') bal--;
        t++;
    }
    return P.substr(pos, t - pos);
    }
```

```
38      else {
39          return P.substr(pos, 3);
40      }
41  }
42
43  string subst(string P, int posb, int pose, string term){
44      string r = P.substr(0, posb);
45      r += term;
46      r += P.substr(pose + 1);
47
48      return r;
49  }
50
51  bool isvar(string s){
52      bool ok = 1;
53      for(int i = 0; i < s.size(); i++)
54          if(s[i] > '9' || s[i] < '0') ok = 0;
55      return ok;
56  }
57
58  void standart(){
59       t1++;
60      for(int i = 0; i < P1.size(); i++)
61          if(P1[i] == 'x')
62              P1 = subst(P1, i, i, tos(t1));
63
64      t1++;
65      for(int i = 0; i < P1.size(); i++)
66          if(P1[i] == 'y')
67              P1 = subst(P1, i, i, tos(t1));
68
69      t1++;
70      for(int i = 0; i < P1.size(); i++)
71          if(P1[i] == 'z')
72              P1 = subst(P1, i, i, tos(t1));
73
74      for(int i = P1.size()-1; i >= 0; i--){
75          if(P1[i] <= 'c' && P1[i] >= 'a'){
76              string vs;
77              for(int j = 0; j < 3; j++) vs += P1[i];
78              P1 = subst(P1, i, i, vs);
79          }
80      }
81
82      t2++;
83      for(int i = 0; i < P2.size(); i++)
84          if(P2[i] == 'x')
85              P2 = subst(P2, i, i, tos(t2));
86
87      t2++;
88      for(int i = 0; i < P2.size(); i++)
89          if(P2[i] == 'y')
90              P2 = subst(P2, i, i, tos(t2));
91
92      t2++;
93      for(int i = 0; i < P2.size(); i++)
94          if(P2[i] == 'z')
95              P2 = subst(P2, i, i, tos(t2));
96
97      for(int i = P2.size()-1; i >= 0; i--){
```

```
98          if(P2[i] <= 'c' && P2[i] >= 'a'){
99              string vs;
100             for(int j = 0; j < 3; j++) vs += P2[i];
101             P2 = subst(P2, i, i, vs);
102         }
103     }
104 }
105 
106 int main(){
107     ios::sync_with_stdio(0), cin.tie(0), cout.tie(0);
108     cin >> P1 >> P2;
109     standart();
110 
111     while(1){
112         int t = 0;
113         while(t + 3 < P1.size() && t + 3 < P2.size()){
114             if(P1[t] == P2[t])
115                 t++;
116             else
117                 if(isvar(P1.substr(t, 3)) && isvar(P2.substr(t, 3)))
118                 t += 3;
119             else
120                 break;
121         }
122 
123         if(t + 3 >= P1.size() || t + 3 >= P2.size()){
124             for(int i = 0; i < P1.size(); i++)
125                 if( i < P1.size() && P1[i] <= 'c' && P1[i] >= 'a'){
126                     string vs;
127                     vs += P1[i];
128                     P1 = subst(P1, i, i + 2, vs);
129                 }
130 
131             cout<<P1<<endl;
132             return 0;
133         }
134         else{
135             string TP, IP;
136             int f;
137             if(isvar(P2.substr(t, 3)) ) {
138                 TP = P2;
139                 IP = P1;
140                 f = 2;
141             }
142             else{
143                     TP = P1;
144                     IP = P2;
145                     f = 1;
146             }
147 
148             string tx = TP.substr(t, 3);
149             string tt = termf(IP, t);
150 
151             if(f == 1){
152                 for(int i = 0; i < tt.size(); i++)
153                     if(i + 3 < tt.size() && isvar(tt.substr(i, 3))){
154                         int ty = toa(tt.substr(i, 3));
155                         if(mask[ty] == -1){
156                             t1++;
157                             mask[ty] = t1;
```

```
                        }
                    }

                for(int i = 0; i < tt.size(); i++)
                    if(i + 3 < tt.size() && isvar(tt.substr(i, 3))){
                        int ty = toa(tt.substr(i, 3));
                        tt = subst(tt, i, i+2, tos(mask[ty]));
                    }
            }
            else{
                for(int i = 0; i < tt.size(); i++)
                    if(i + 3 < tt.size() && isvar(tt.substr(i, 3))){
                        int ty = toa(tt.substr(i, 3));
                        if(mask[ty] == -1){
                            t2++;
                            mask[ty] = t2;
                        }
                    }

                for(int i = 0; i < tt.size(); i++)
                    if(i + 3 < tt.size() && isvar(tt.substr(i, 3))){
                        int ty = toa(tt.substr(i, 3));
                        tt = subst(tt, i, i+2, tos(mask[ty]));
                    }
            }

            for(int i = 0; i < TP.size(); i++)
                if(TP.substr(i, 3) == tx)
                    TP = subst(TP, i, i + 2, tt);

            if(f == 2){

                P2 = TP;
                P1 = IP;
            }
            else{
                P1 = TP;
                P2 = IP;
            }
        }
    }
}
```

Математика. 8-9 класс

Задача III.1.2.1. (15 баллов)

На новогодние праздники родители возили Оксану на Регульские острова. Они вылетели из Москвы в 12:00 по московскому времени и приземлились на островах на следующий день в 7:30 по местному времени. При возвращении они стартовали в 9:30 по местному времени и приземлились в Москве в 18:00 по московскому времени в тот же день. Считая, что продолжительность полета была одинаковой в обоих случаях, найдите, сколько времени было на Регульских островах, когда Оксана приземлилась в Москве?

Решение

Пусть разница во временных поясах между Москвой и Регульскими островами равна x часов, а время перелета — y часов. Тогда получаем систему:

$$\begin{cases} x + y = 19,5 \\ y - x = 8,5 \end{cases}$$

Откуда, $x = 5,5$, $y = 14$. Следовательно, в момент прилета в Москву на островах было 23:30.

Система оценки

- Полное обоснованное решение — 15 баллов.
- Составлена система — 10 баллов.
- Задача не решена или решена неверно — 0 баллов.

Ответ: 23:30.

Задача III.1.2.2. (15 баллов)

Портье перепутал ключи от 100 закрытых номеров гостиницы. Он не знает, какой ключ к какому номеру подходит. Какое наименьшее количество попыток открывания нужно сделать, чтобы определить ключи ко всем номерам?

Решение

Подойдем к первому номеру и будем пробовать открыть его, последовательно подбирая ключи. Если за 99 попыток открыть не удалось, то оставшийся ключ будет от этого номера. Для второго номера будет достаточно 98 попыток, для третьего — 97 попыток и т. д. Всего попыток:

$99+98+97+\cdots+2+1 =$ (по формуле суммы арифметической прогрессии) $= \dfrac{99+1}{2}99 = 4950.$

Система оценки

- Полное обоснованное решение — 15 баллов.
- Количество отличается от верного на 1 для каждой двери — 5 баллов.
- Задача не решена или решена неверно — 0 баллов.

Ответ: 4950.

Задача III.1.2.3. (20 баллов)

На доске записаны 576 натуральных чисел. Среди любых 122 чисел найдется хотя бы одно четное число, а среди любых 456 чисел найдется хотя бы одно нечетное число. Может ли сумма всех записанных чисел равняться $2020 \cdot 2021$?

Решение

Среди всех чисел может быть не более 121 нечетных чисел и не более 455 четных чисел. Так как чисел всего 576, то четных чисел ровно 455 и нечетных чисел ровно 121. Сумма всех чисел — нечетное число, а $2020 \cdot 2021$ — четное число.

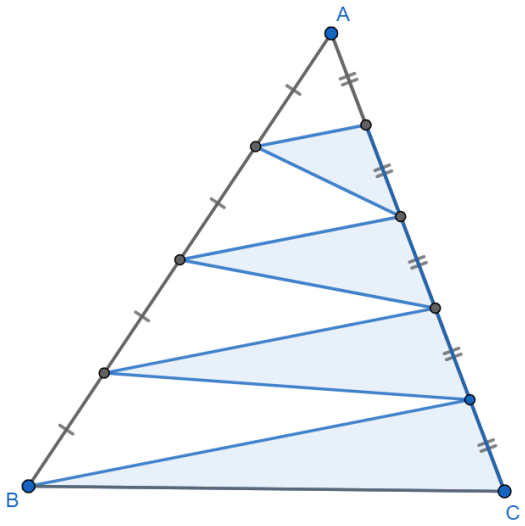
Система оценки

- Полное обоснованное решение — 20 баллов.
- Определено количество четных и нечетных чисел — 10 баллов.
- Задача не решена или решена неверно — 0 баллов.

Ответ: нет.

Задача III.1.2.4. (20 баллов)

Точка случайным образом (равномерно) брошена в треугольник ABC. Найдите вероятность попадания точки в один из закрашенных треугольников.



Решение

Отметим точки D, E, F, G, H, I, J и опустим перпендикуляры BN, FM, EL, DK на прямую AC. Искомая вероятность равна отношению суммы площадей закрашенных треугольников DGH, EHI, FIJ и BJC к площади треугольника ABC.

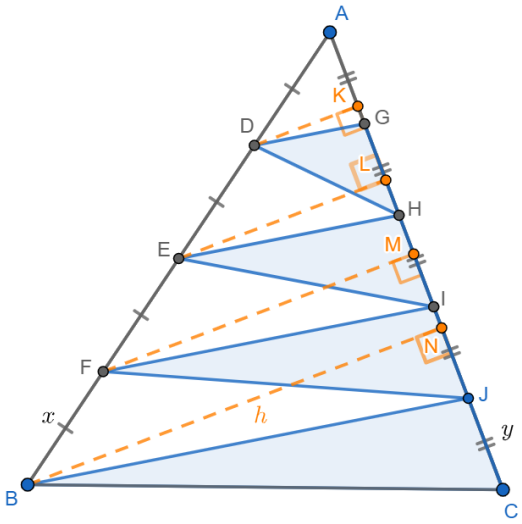
Введем обозначения:

$AD = DE = EF = FB = x$,

$AG = GH = HI = IJ = JC = y$,

$BN = h$.

Прямоугольные треугольники ADK, AEL, AFM, ABM подобны по общему острому углу, отсюда получаем $DK = \frac{1}{4}h$, $EL = \frac{1}{2}h$, $FM = \frac{3}{4}h$.



Тогда $S_{ABC} = \frac{1}{2}BN \cdot AC = \frac{5}{2}hy$, а сумма площадей закрашенных треугольников равна:

$$S_{DGH} + S_{EHI} + S_{FIJ} + S_{BJC} = \frac{1}{2}DK \cdot GH + \frac{1}{2}EL \cdot HI + \frac{1}{2}FM \cdot IJ + \frac{1}{2}BM \cdot JC =$$

$$= \frac{1}{2}y \cdot (\frac{1}{4}h + \frac{1}{2}h + \frac{3}{4}h + h) = \frac{5}{4}hy$$

.

Искомая вероятность равна: $\left(\frac{5}{4}hy\right) : \left(\frac{5}{2}hy\right) = \frac{1}{2}$.

Система оценки

- Полное обоснованное решение — 20 баллов.
- Допущена одна вычислительная ошибка, при этом ход решения верен — 15 баллов.
- Сформулирована идея о том, что искомая вероятность равна отношению площади закрашенных треугольников к общей площади ΔABC — 5 баллов.
- Задача не решена или решена неверно — 0 баллов.

Ответ: $\frac{1}{2}$.

Задача III.1.2.5. (30 баллов)

Найдите все действительные корни уравнения:

$$(x^2 + 7x + 12)(x^2 - 3x + 2)(x^2 + 2x + 4) = 108.$$

Решение

Разложим первый и третий квадратные трехчлены в левой части уравнения на множители, а затем перегруппируем сомножители:

$$(x + 3)(x + 4)(x - 1)(x - 2)(x^2 + 2x + 4) = (x^2 + 2x - 3)(x^2 + 2x - 8)(x^2 + 2x + 4).$$

После замены $t = x^2 + 2x$ получим уравнение: ;$(t-3)(t-8)(t+4) - 108 = 0$.;

После раскрытия скобок и приведения подобных слагаемых имеем:

$$t^3 - 7t^2 - 20t - 12 = 0.$$

Знакопеременная сумма коэффициентов многочлена в левой части равна нулю $(1 + 7 - 20 + 12 = 0)$, поэтому один из корней уравнения $t = -1$. Разложим левую часть на множители:

$$(t+1)(t^2 - 8t - 12) = 0.$$

Корнями данного уравнения являются числа -1, $4 + 2\sqrt{7}$ и $4 - 2\sqrt{7}$.

Выполним обратную замену.

а) при $t = -1$ получаем $x^2 + 2x = -1$, отсюда $x_1 = -1$.

б) при $t = 4 + 2\sqrt{7}$ получаем $x^2 + 2x - (4 + 2\sqrt{7}) = 0$; это уравнение имеет два действительных корня $x_{2,3} = -1 \pm \sqrt{(5 + 2\sqrt{7})}$.

в) при $t = 4 - 2\sqrt{7}$ получаем уравнение $x^2 + 2x - 4 + 2\sqrt{7} = 0$, упрощенный дискриминант $D_1 = 5 - 2\sqrt{7}$ которого отрицателен, поэтому действительных корней оно не имеет.

Система оценки

- Полное обоснованное решение — 30 баллов.
- Верное в целом решение, но допущена одна вычислительная ошибка — 25 баллов.
- После замены получено кубическое уравнение — 15 баллов.
- Найдены корни этого уравнения — 5 баллов (в дополнение к 15).
- Задача не решена или решена неверно — 0 баллов.

Ответ: -1; $-1 + \sqrt{(5 + 2\sqrt{7})}$; $-1 - \sqrt{(5 + 2\sqrt{7})}$.

Математика. 10-11 класс

Задача III.1.3.1. (15 баллов)

Планету Пандора осваивают две компании: разведывательная и строительная. Для них привезли с Земли роботов-разведчиков и роботов-строителей. Для увеличения количества роботов своей специальности обе компании тайно отправляют на склад хакеров, которые перепрограммируют роботов. В результате этого десятая часть роботов-разведчиков считают себя строителями, а десятая часть роботов-строителей считают себя разведчиками. Если же рассматривать всех роботов, то пятая часть роботов считают себя разведчиками. Какую часть составляли роботы-разведчики от общего числа привезенных с Земли роботов?

Решение

Пусть x — количество привезенных с Земли роботов-строителей, y — количество привезенных роботов-разведчиков. После работы хакеров $0,9y + 0,1x = 0,2(x + y)$. Отсюда $x = 7y$. Поэтому $\frac{y}{x+y} = \frac{1}{8}$.

Система оценки

- Полное обоснованное решение — 15 баллов.
- Составлена система — 10 баллов.
- Задача не решена или решена неверно — 0 баллов.

Ответ: $\frac{1}{8}$.

Задача III.1.3.2. (15 баллов)

В рамках тестирования генератора случайных чисел программист Вася решает уравнения вида $2Ax - 3A + 4 = 0$. Считая, что величина A равномерно распределена на отрезке $[-1; 4]$, найдите вероятность того, что число x окажется больше $1/2$.

Решение

При $A = 0$ уравнение не имеет решений, а при $A \neq 0$ получаем $x = \frac{3A-4}{2A}$. Решим неравенство $\frac{3A-4}{2A} > \frac{1}{2}$:

$$\frac{3A-4}{A} - 1 > 0 \Leftrightarrow \frac{2(A-2)}{A} > 0 \Leftrightarrow A \in (-\infty; 0) \cup (2; +\infty).$$

На отрезке $[-1; 4]$ решением будет объединение промежутков $[-1; 0)$ и $(2; 4]$.

Искомая вероятность равна отношению суммы длин отрезков $[-1; 0]$ и $[2; 4]$ к длине отрезка $[-1; 4]$: $P = \frac{1+2}{5} = \frac{3}{5}$.

Система оценки

Баллы суммируются:

- Верно составлено неравенство ИЛИ верно найдено значение A, соответствующее значению $x = \frac{1}{2}$ — 5 баллов.
- Решено неравенство — 5 баллов.
- Верно вычислена вероятность — 5 баллов.

Задача не решена или решена неверно — 0 баллов.

Ответ: $\frac{3}{5}$.

Задача III.1.3.3. (20 баллов)

На гранях игрального кубика записаны числа 1, 2, 2, 3, 6, 6. В каждом туре два игрока по очереди бросают кубик. Первый игрок выигрывает, если при бросании на гране кубика выпадает 6 очков, второй — если 3 очка. Игра заканчивается в случае выигрыша одного из игроков. Найдите вероятность выигрыша второго игрока (не обязательно в первом туре).

Решение

Событие A = «выигрыш второго игрока» является объединением независимых событий A_n = «второй игрок выиграет именно в n-м туре», где n принимает все значения от 1 до бесконечности. Событие A_1 является пересечением событий «первый игрок не получает 6» и «второй игрок получает 3». $P(A_1) = \frac{4}{6} \cdot \frac{1}{6} = \frac{1}{9}$. Событие A_2 происходит, если в первом туре игра не закончилась, а во втором туре победил второй игрок. $P(A_2) = \left(\frac{4}{6} \cdot \frac{5}{6}\right) \cdot \left(\frac{4}{6} \cdot \frac{1}{6}\right) = \frac{5}{9} \cdot \frac{1}{9}$. Аналогично, $P(A_3) = \frac{5}{9} \cdot \frac{5}{9} \cdot \frac{1}{9}$ и $P(A_{n+1}) = P(A_n) \cdot \frac{5}{9}$. То есть вероятности событий $P(A_n)$ представляют собой бесконечно убывающую геометрическую прогрессию с первым членом $\frac{1}{9}$ и знаменателем $\frac{5}{9}$. $P(A) = \sum_{n=1}^{\infty} P(A_n) = \dfrac{\frac{1}{9}}{1 - \frac{5}{9}} = 0,25$.

Система оценки

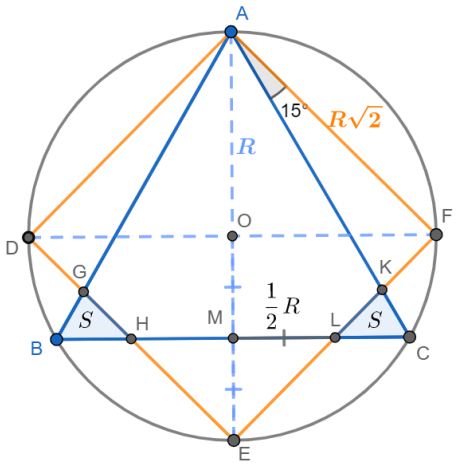
- Полное обоснованное решение — 20 баллов.
- Найдена вероятность выигрыша в 1 туре — 5 баллов.
- Определены вероятности последующих событий — 10 баллов. (В дополнение к 5 баллам.)
- Задача не решена или решена неверно — 0 баллов.

Ответ: 0,25.

Задача III.1.3.4. (20 баллов)

Вписанные в круг квадрат и правильный треугольник имеют общую вершину. В круг случайным образом (равномерно) бросают точку. Найдите вероятность, что она попадет в треугольник, но не попадет в квадрат.

Решение



Пусть в окружность с центром O и радиусом R вписаны правильный треугольник ABC и квадрат $ADEF$, тогда стороны треугольника и квадрата равны $R\sqrt{3}$ и $R\sqrt{2}$ соответственно. Обозначим точки пересечения треугольника и квадрата как G, H, K и L (см. рисунок). Тогда искомая вероятность P равна отношению суммы площадей закрашенных треугольников BGH и CKL к площади круга. В силу симметрии относительно прямой AC площади закрашенных треугольников равны; обозначим площадь каждого из них как S.

1. Найдем CK. Из прямоугольного треугольника AFK:

$$AK = AF : cos15^\circ = R\sqrt{2} : \frac{\sqrt{6}+\sqrt{2}}{4} = \frac{4R}{\sqrt{3}+1} = 2(\sqrt{3}-1)R,$$

   где значение $cos15^\circ = cos(45^\circ - 30^\circ) = cos45^\circ cos30^\circ + sin45^\circ sin30^\circ = \frac{\sqrt{6}+\sqrt{2}}{4}$. Тогда $CK = AC - AK = R(\sqrt{3} - 2\sqrt{3} + 2) = (2 - \sqrt{3})R$.
2. Найдем CL. Т.к. AM — медиана ΔABC, то $AO : OM = 2 : 1$ и $OM = \frac{1}{2}R$. Следовательно, $EM = OE - OM = \frac{1}{2}R$. Треугольник EML — равнобедренный прямоугольный, поэтому $ML = ME = \frac{1}{2}R$. Отсюда $CL = CM - ML = \frac{\sqrt{3}-1}{2}R$.
3. $S = S_{CKL} = \frac{1}{2}CL{\cdot}CK{\cdot}sin\angle LCK = \frac{\sqrt{3}}{4}CL{\cdot}CK = \frac{\sqrt{3}}{4}{\cdot}\frac{\sqrt{3}-1}{2}R{\cdot}(2-\sqrt{3})R = \frac{9-5\sqrt{3}}{8}R^2$.
4. Окончательно получаем: $P = \frac{2S}{\pi R^2} = \frac{9-5\sqrt{3}}{4\pi}$.

Система оценки

- Полное обоснованное решение — 20 баллов.
- Допущена одна вычислительная ошибка, при этом ход решения верен — 15 баллов.
- Сформулирована идея о том, что искомая вероятность равна отношению площади закрашенных треугольников к площади круга — 5 баллов.
- Задача не решена или решена неверно — 0 баллов.

Ответ: $\frac{9-5\sqrt{3}}{4\pi}$.

Задача III.1.3.5. (30 баллов)

Найдите наименьшее натуральное m, при котором существуют натуральные x и y, удовлетворяющие уравнению:

$$(x^2 + 4y^2)^2 + 4my(4y^2 + x^2) = m^2x^2$$

.

Решение

Сделаем замену $2y = t$ — четное натуральное число. Рассмотрим это уравнение как квадратное относительно m, и найдем его корни: $m = \frac{(x^2+t^2)(t+\sqrt{x^2+t^2})}{x^2}$ (второй корень отрицательный, так как $\sqrt{x^2 + t^2} > t$).

Отсюда:

$$mx^2 = (x^2 + t^2)(t + \sqrt{x^2 + t^2}).$$

Пусть $d = \text{НОД}(x; t)$ и $x = x_0d$, $t = t_0d$. Подставляя в равенство, получим:

$$mx_0^2 = d(x_0^2 + t_0^2)(t_0 + \sqrt{x_0^2 + t_0^2}).$$

x_0 и x_0^2 взаимно просто с t_0, t_0^2 и с $x_0^2 + t_0^2$. Следовательно, m должно быть кратно $x_0^2 + t_0^2$ и $x_0^2 + t_0^2$ должно быть полным квадратом (иначе $\sqrt{x_0^2 + t_0^2}$ будет иррациональным числом и равенство не будет верным). Наименьшая сумма квадратов, являющаяся квадратом, это $3^2 + 4^2 = 5^2$. Значит, $m \geqslant 25$. При $m = 25$, $x = 3$ и $y = 2$ получаем верное равенство.

Система оценки

- Полное обоснованное решение — 30 баллов.
- Верное в целом решение, но допущена одна вычислительная ошибка — 25 баллов.
- Уравнение рассмотрено как квадратное относительно m и найдены его корни — 5 баллов.
- Задача не решена или решена неверно — 0 баллов.

Ответ: 25.