Решения задач школьного тура ВсОШ по информатике. 9-11 класс. Челябинск, 2017

Задача А. **Карточный трюк**

На переменах Саша любит показывать своим одноклассникам фокусы.

Необходимый атрибут любого Сашиного фокуса - карточная колода, состоящая из 20 карт.

Номиналы карт Сашиной колоды: **1**, **2**, **3**, **4**, **5**, **6**, **7**, **8**, **9**. На следущей перемене Саша хочет показать свой самый любимый фокус: он вызывает двух одноклассников к себе, тасует колоду, а затем, пока она не кончится, достает из нее по 2 карты, одну для каждого одноклассника. Фокус заключается в том, что участники всегда получают карту одного и того же номинала.

Каждый день у Саши разная колода карт. Зная сегодняшнюю колоду, выведите **Yes**, если Саша сможет выполнить трюк, и **No**, если у него нет шансов.

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

using namespace std;

int main() {
    for (int i = 0; i < 10; i++) {
        string q, w;
        cin >> q >> w;
        if (q != w) {
            puts("No");
            return 0;
        }
    }
    puts("Yes");
     return 0;
}
```

Задача B. **Трыц-тыц калькулятор**

На уроке информатики ребятам задали написать простой калькулятор, который умел бы только прибавлять и отнимать. Петя старался достаточно долго, но у него ничего не получилось. Тогда он решил попросить помощи у фиксиков.

Чтобы проверить работу калькулятора, учитель вводит исходное число *N*, а в следующей строке вводит операции, которые нужно последовательно применить к этому числу. Операции могут быть двух видов: 1) "прибавить некоторое число" и обозначается как "+25" (прибавить число 25), и 2) "отнять некоторое число", обозначается как "-10" (отнять число 10).

Все операции записаны слитно и образуют одну строку без пробелов.

Вы исполняете роль фиксиков. Помогите Пете вычислить значение числа *N* после выполнения всех операций, записанных во второй строке.

C++

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

using namespace std;

int main() {
    int n;
     cin >> n;
     string s;
     cin >> s;
     string ans;
     for (int i = 0; i < (n + 1) / 2; i++) {
          ans += 'a';
          s[i] += 1;
          if (s[i] > 'z')
               s[i] = 'a';
     }
     for (int i = (n + 1) / 2, j = (n) / 2 - 1; i < n; i++, j--) {
          if (s[i] < s[j]) {
               ans += s[j] - s[i] - 1 + 'a';
          }
          else {
               ans += 'z' - s[i] + s[j];
          }
     }
     cout << ans;
}
```

Python

```
n = int(input())
s = input()
print(n + eval(s))
```

Задача C. Киберспорт

Матвею очень нравится игра «Pokermon Stop?». Так сильно нравится, что он решил взойти на киберспортивную арену.

По его наблюдениям, у самых сильных покермонов имена являются палиндромами. Напомним, палиндромом называется слово, которое одинаково читается слева направо и справа налево. Имена всех покермонов, для удобства произношения состоят из строчных английских букв. Также известно, что у всех покермонов длина имени одинакова и равна n.

Матвей прекрасно знает, что скрестив двух разных покермона он получит нового, с другим именем. Имя нового покемона определяется согласно следующим правилам:

- Каждому символу ставится в соответствие число — номер этой буквы в английском алфавите. Так букве "d" будет соответствовать номер 4, а букве "a" номер 1.
- Числа, соответствующие буквам, находящимся на i-ой позиции в обоих именах покермонов складываются. Если сумма превышает размер английского алфавита, то из нее вычитают 26. Получается число si.
- i-ый символ в имени нового покермона есть символ, которому соответствует число si.

Таким образом, скрестив покермона "aby" и "bab", получится покермон "cca".

У Матвея уже есть покермон, которого зовут P. Помогите ему узнать лексикографически минимальное имя покермона, которого нужно поймать, чтобы при скрещивании получить покермона с именем-палиндромом.

Не смотря на то, что покермон Матвея мог уже иметь имя-палиндром, Матвей все равно хочет попрактиковаться в скрещивании.

Python

```
n = int(input())
s1 = input()
s = []
for i in s1:
    s += i
ans = ""
for i in range((n + 1) // 2):
    ans += 'a'
    s[i] = chr(ord(s[i]) + 1)
    if s[i] > 'z':
        s[i] = 'a'
j = (n) // 2 - 1
for i in range((n + 1) // 2, n) :
    if s[i] < s[j]:
        ans += chr(ord(s[j]) - ord(s[i]) - 1 + ord('a'))
    else:
        ans += chr(ord('z') - ord(s[i]) + ord(s[j]))
    j -= 1
print(ans)
```

Задача D. **Именные перестановки**

Петя - заядлый игрок в Майнкрафт. Поэтому 1 сентября Петя пришел в школу, и тут же спросил никнеймы (имена в игре) всех своих одноклассников. В первый день все учителя, вместо того, чтобы рассказывать новое и интересное, весь урок знакомились и записывали имена в классный журнал.

От скуки, Петя записал все перестановки своего никнейма в Майнкрафте ("n00b"). Получилось: **n00b, n0b0, n00b, nb00, nb00, 0n0b** и т.д.

Скучный урок все не кончался, а Петя задумался, хватит ли ему тетрадки, чтобы записать все перестановки никнеймов своих одноклассников? Помогите Пете для каждого никнейма быстро находить список всех его перестановок. Все буквы в никнейме следует считать различными. Таким образом, у никнейма "n00b" каждая перестановка будет встречаться два раза (две цифры "0" просто меняются местами и получается новая перестановка). Никнейм может состоять из строчных английских букв и/или цифр.

C++

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

using namespace std;

int main() {
    string s;
    cin >> s;
    int n = s.size();
    vector<int> a(n);
    for (int i = 0; i < n; ++i)
        a[i] = i;
    do {
        string cur = "";
        for (int i = 0; i < n; ++i)
            cur += s[a[i]];
        cout << cur << "\n";
    } while (next_permutation(a.begin(), a.end()));

}
```

Python

```
def Permutations(n):

    def Perm(i):
        if i == n:
            for x in p:
                print(a[x], end="")
            print()
        else:
            for j in range(i+1, n+1):
```

```
                Perm(i+1);
                p[i], p[j] =  p[j], p[i]
            Perm(i+1)
            k = p[i]
            for j in range(i, n):
                p[j] = p[j+1]
            p[n] = k

    Perm(0)

a = sorted(input())
n = len(a)
p = list(range(n))
Permutations(n-1)

Python (с использованием встроенной функции)
from itertools import permutations as perm

s = "".join(sorted(list(input())))
for x in perm(s, len(s)):
    print(*x, sep="")
```

Задача E. **Леша и музыка**

Леша очень любит слушать музыку. Для прослушивания музыки он пользуется приложением его друга Жени.

Фишка Жениного приложения в том, что в начале месяца оно удаляет все предыдущие плейлисты, чтобы разнообразить музыкальные вкусы пользователей. Поэтому каждый месяц Леша составляет новый плейлист.

Если в конце месяца у Леши было в плейлисте B песен, то за пользование приложением с него взимается плата $НОК(A, B)$, где A — некоторое число, генерируемое приложением.

Леша с Женей — близкие друзья, поэтому Леша хочет поддержать Женю как разработчика и заплатить за его приложение не меньше T. С другой стороны, денег у Леши не очень много, поэтому из всех возможных вариантов (тех, что не меньше T) он хочет выбрать минимальный.

Помогите ему в этом — подберите такое число песен в плейлисте B, что $F = НОК(A, B)$ - минимально среди всех $F \geq T$.

Если удовлетворяющих условию количеств песен в плейлисте B несколько, то выберите из них минимальное, чтобы Леше не пришлось долго составлять плейлист.

Наименьшее общее кратное (НОК) двух целых чисел m и n есть наименьшее натуральное число, которое делится на m и n без остатка.

```
#include<bits/stdc++.h>
using namespace std;

map<long long, int> b, c;

int main() {
     long long a, t;
     cin >> a >> t;
     long long lcm = (t + a - 1) / a * a;
     for (long long i = 2; a > 1; ++i) {
          while (a % i == 0)
               b[i]++, a /= i;
          if (i * i > a) {
               b[a]++;
               break;
          }
     }
     for (long long i = 2; lcm > 1; ++i) {
          while (lcm % i == 0)
               c[i]++, lcm /= i;
          if (i * i > lcm) {
               c[lcm]++;
               break;
          }
     }
     long long ans = 1;
     for (auto it : c)
```

```
            if (c[it.first] > b[it.first])
                for (int j = 0; j < it.second; ++j)
                    ans *= it.first;
        cout << ans;
}
```