

Отборочный тур МОШ по информатике 2024, 6 класс

15 дек 2023 г., 00:00 — 17 янв 2024 г., 23:59

100 баллов

Даша и лимонад

Даша очень любит лимонад! Сегодня она сходила в магазин и купила n бутылок лимонада, каждая из которых объемом t литров.

Она поставила их всех в один ряд и за один шаг делает следующее: берет одну бутылку, затем, если бутылка не пуста, выпивает из нее 1 литр, ставит обратно и переходит к следующей. После последней бутылки она переходит снова к первой.

Изначально все бутылки полные. Даша хочет узнать: сколько бутылок будет выпито полностью, если она сделает k вышеописанных шагов? Помогите ей выяснить это.

Формат входных данных

Первая строка содержит три целых числа n, t и k ($1 \leq n \leq 10^9, 1 \leq t \leq 10^9, 1 \leq k \leq 10^9$) — количество бутылок, объем каждой бутылки и количество шагов соответственно.

Формат выходных данных

Выведите одно число — количество полностью выпитых бутылок.

Критерии оценивания

Номер подзадачи	Баллы	Ограничения	Комментарий
0	0	—	Примеры из условия.
1	40	$1 \leq n \leq 100$	Баллы начисляются за каждый тест в отдельности.
2	60	Основные ограничения	Баллы начисляются за каждый тест в отдельности.

Примеры

2 2 2	0
2 1 1	1
4 3 10	2
4 3 100	4

Ограничения

Время выполнения: 1 секунда
Память: 256 MB

КодC++

```
1  #include <iostream>
2  #include <cmath>
3
4  using namespace std;
5
6  int main() {
7      long long n, t, k;
8
9      cin >> n >> t >> k;
10     long long a = (t - 1) * n;
11     if (k <= a)
12         cout << 0;
13     else {
14         k -= a;
15         cout << min(n, k);
16     }
17     return 0;
18 }
19
```

100 баллов

Пробка

Глеб оказался в автомобильной пробке. Применяв зоркий глаз, Глеб насчитал, что перед ним стоят n машин. Также он знает, что зеленый свет горит ровно a секунд и за каждую секунду зеленого света с пробки успевают уехать ровно b машин. Красный свет горит ровно c секунд.

Глеб хочет узнать, сколько еще секунд он будет стоять в пробке, если прямо сейчас загорится зеленый свет.

Формат входных данных

Первая строка входных данных содержит одно целое число n ($1 \leq n \leq 10^9$) — количество машин, стоящих перед Глебом.

Вторая строка входных данных содержит три целых числа a, b и c ($1 \leq a, b, c \leq 10^9$).

Формат выходных данных

Выведите единственное число — сколько секунд придется стоять Глебу в пробке.

Обратите внимание, что ответ в этой задаче может быть довольно большим и не помещаться в 32-битные типы данных. Рекомендуется использовать 64-битный тип данных, например `long long` в C++, `long` в Java или `int64` в Pascal.

Критерии оценивания

В этой задаче каждый тест, кроме тестов из условия, оценивается независимо.

Примеры

5
4 2 3
3
11
4 1 3
18
2
2 1 5
8
2
1 2 5
7

Ограничения

Время выполнения: 1 секунда

Память: 256 MB

Код

C++

```
1  #include <bits/stdc++.h>
2
3  using namespace std;
4  #define int long long
5
6  signed main() {
7      int n, a, b, c;
8      cin >> n >> a >> b >> c;
9      int n1 = (n + 1 + b - 1) / b * b;
10     int n2 = n1 / b;
11     int k1 = (n2 - 1) / a * (a + c);
12     cout << k1 + ((n2 - 1) % a + 1);
13 }
14
```

100 баллов

Преобразование Алисы

Алисе подарили массив a , состоящий из n целых чисел. Она очень любит странные способы преобразования массивов, так что сегодня она придумала следующий алгоритм:

```
\begin
  \item Ищет три подряд идущих одинаковых числа
  \item Если они нашлись, то она удаляет одно из этих чисел, и переходит к первому шагу, иначе работа
алгоритма завершается
\end
```

Она просит вас узнать, как будет выглядеть массив после исполнения описанного выше алгоритма.

Формат входных данных

Первая строка входных данных содержит одно целое число n ($1 \leq n \leq 2 \cdot 10^5$) — количество элементов массива.

Вторая строка входных данных содержит n целых чисел a_1, a_2, \dots, a_n ($1 \leq a_i \leq 10^9$) — элементы массива a .

Формат выходных данных

Выведите массив после исполнения алгоритма.

Критерии оценивания

Номер подзадачи	Баллы	Ограничения	Комментарий
0	0	—	Примеры из условия.
1	40	$1 \leq n \leq 100$	Баллы начисляются за каждый тест в отдельности.
2	60	Основные ограничения	Баллы начисляются за каждый тест в отдельности.

Примеры

6
4 3 10 10 10 1

4 3 10 10 1

5
1 1 1 1 1

1 1

5
1 2 3 4 9

1 2 3 4 9

Ограничения

Время выполнения: 1 секунда

Память: 256 MB

```
1  #include <bits/stdc++.h>
2
3  #define ll long long
4  #define ld long double
5  #define fast ios_base::sync_with_stdio(0); cin.tie(0);
6
7  using namespace std;
8
9  ll INF = (ll) 1e20;
10 ll mod = (ll) 1e9 + 7;
11
12 mt19937
13   gen(chrono::steady_clock::now().time_since_epoch().count());
14
15 signed main() {
16     fast
17     ll n;
18     cin >> n;
19     vector<ll> a(n);
20     for (int i = 0; i < n; ++i) {
21         cin >> a[i];
22     }
23     vector<ll> ans;
24     for (int i = 0; i < n - 2; ++i) {
25         if (a[i] != a[i + 1] || a[i] != a[i + 2]) {
26             ans.push_back(a[i]);
27         }
28     }
29     if(n > 1)
30         ans.push_back(a[n - 2]);
31     ans.push_back(a[n - 1]);
32     for (auto i : ans) {
33         cout << i << ' ';
34     }
35 }
36
```

100 баллов

Очень простая задача

Целое число называется *простым*, если оно не меньше двух и не делится ни на какое целое положительное число, кроме единицы и самого себя.

От вас требуется написать программу, которая ищет отрезок из L последовательных натуральных чисел, содержащий ровно K простых чисел. Чтобы результаты было легче анализировать, вас просят ограничиться в поисках первыми **тридцатью тысячами** натуральных чисел.

Формат входных данных

На вход программе подаются целые числа L и K ($1 \leq L \leq 30\,000, 0 \leq K \leq L$), каждое в отдельной строке.

Формат выходных данных

Если в пределах до 30 000 найдётся отрезок из L подряд идущих натуральных чисел, среди которых ровно K простых, выведите минимальное и максимальное число на этом отрезке. В противном случае выведите единственное число -1 . Если существует несколько отрезков, удовлетворяющих условию, выведите любой.

Критерии оценивания

Все тесты оцениваются независимо. Разные тесты могут оцениваться в разное количество баллов.

Примеры

```
20
5
```

```
8 27
```

```
100
66
```

```
-1
```

Ограничения

Время выполнения: 1 секунда

Память: 256 MB

Код

C++


```
1  #include <cstdio>
2  #include <iostream>
3  using namespace std;
4
5  #define MAX_PRIME 30000
6
7  bool is_prime(int x)
8  {
9      if (x < 2) {
10         return false;
11     }
12     for (int i = 2; i * i <= x; ++i) {
13         if (x % i == 0) {
14             return false;
15         }
16     }
17     return true;
18 }
19
20 int main()
21 {
22     //freopen("prime-segment.in","r",stdin);
23     //freopen("prime-segment.out","w",stdout);
24     int len, k;
25     cin >> len >> k;
26
27     int cnt = 0;
28     for (int i = 0; i < len; ++i) {
29         cnt += is_prime(i);
30     }
31
32     for (int x = 1; x + len - 1 <= MAX_PRIME; ++x) {
33         cnt += is_prime(x + len - 1) - is_prime(x - 1);
34         if (cnt == k) {
35             cout << x << " " << x + len - 1 << endl;
36             return 0;
37         }
38     }
39     cout << "-1\n";
40     return 0;
41 }
42
```

100 баллов

Пёстрая дата

По текущей дате определите ближайшую следующую дату, запись которой в виде день, месяц и год состоит из различных цифр.

Учтите, что если день или номер месяца меньше 10, то они записываются без лишних нулей. Например, дата «8 марта 2019 года» запишется так: «8.3.2019».

Гарантируется, что изначальная дата корректна.

Формат входных данных

В первой строке задано одно целое число d ($1 \leq d \leq 31$) — день текущей даты.

Во второй строке задано одно целое число m ($1 \leq m \leq 12$) — месяц текущей даты.

В третьей строке задано одно целое число y ($1000 \leq y \leq 10^6$) — год текущей даты.

Формат выходных данных

Выведите одну строку — ближайшую следующую дату, в записи которой все цифры различны.

Критерии оценивания

Все тесты в этой задаче оцениваются независимо.

Примечание

Високосный год — такой год, что в нём 366 дней, а именно добавляется 29 февраля. Год является високосным, если он делится на 400 или делится на 4, но не делится на 100.

Примеры

18 12 2018
4.3.2019

1 2 3456
7.2.3456

Ограничения

Время выполнения: 1 секунда
Память: 256 MB

Код

C++

```
1  #include <fstream>
2  #include <iostream>
3  using namespace std;
4
5  bool leap_year(int y) {
6      return (y % 4 == 0 && y % 100 != 0 || y % 400 == 0);
7  }
8
9  int day_in_month(int m, int y) {
10     if (m == 2)
11         if (leap_year(y))
12             return 29;
13         else
14             return 28;
15     else if (m == 4 || m == 6 || m == 9 || m == 11)
16         return 30;
17     else
18         return 31;
19 }
20
21 void inc(int & d, int & m, int & y) {
22     d += 1;
23     int dm = day_in_month(m, y);
24     if (d > dm) {
25         d = 1;
26         m += 1;
27         if (m > 12) {
28             m = 1;
29             y += 1;
30         }
31     }
32 }
33
34 bool good_data(int d, int m, int y) {
35     int a[10];
36     for (int i = 0; i < 10; i++) a[i] = 0;
37     a[d % 10] += 1;
38     if (d > 9) a[d / 10] += 1;
39     a[m % 10] += 1;
40     if (m > 9) a[m / 10] += 1;
41     while (y > 0) {
42         a[y % 10] += 1;
43         y /= 10;
44     }
45     bool good = true;
46     for (int elem : a)
47         if (elem > 1) {
48             good = false;
49             break;
50         }
51     return good;
52 }
53 int main() {
54     int day, month, year;
55
56     cin >> day >> month >> year;
57     inc(day, month, year);
58     while (!good_data(day, month, year))
59         inc(day, month, year);
60     cout << day << '.' << month << '.' << year;
61
62     return 0;
```


100 баллов

Закрась по правилам

Дан клетчатый прямоугольник из N строк по M клеток, каждая из которых или уже закрашена, или еще не закрашена. Если в каком-либо квадрате размером 2×2 три клетки уже закрашены, то можно закрасить и четвёртую клетку.

Оцените, сколько клеток могут в итоге оказаться закрашенными.

Формат входных данных

Первая строка входных данных содержит два целых числа N и M ($1 \leq N, M \leq 500$) — количество строк и столбцов в прямоугольнике. Следующие N строк по M символов описывают клетки прямоугольника. Символ «.» соответствует незакрашенной клетке, а «#» — закрашенной клетке. Строки нумеруются от 1 до N , столбцы — от 1 до M .

Формат выходных данных

Выведите одно число — максимальное количество клеток, которые могут оказаться закрашенными.

Критерии оценивания

Каждый тест оценивается независимо. Решения, правильно работающие для ограничений $1 \leq N, M \leq 50$, будут набирать не менее 60 баллов.

Примеры

```
2 2
##
#.
```

4

```
3 4
#...
#...
###.
```

9

```
3 5
...##
#....
#.#..
```

5

Ограничения

Время выполнения: 2 секунды

Память: 256 MB

Код

C++

```

1  #include <iostream>
2  #include <vector>
3  #include <cstdio>
4
5  using namespace std;
6
7  int table[1000][1000], n, m;
8  vector < pair < int, int> > st;
9
10 int sumSquare(int i, int j)
11 {
12     return table[i][j] + table[i + 1][j] + table[i][j + 1] +
13     table[i + 1][j + 1];
14 }
15 void pushSquare(int i, int j)
16 {
17     if ((0 <= i) && (i < n - 1) && (0 <= j) && (j < m - 1))
18         if (sumSquare(i, j) == 3)
19             st.push_back(make_pair(i, j));
20 }
21 void pushNeighbors(int i, int j)
22 {
23     pushSquare(i - 1, j - 1);
24     pushSquare(i, j - 1);
25     pushSquare(i - 1, j);
26     pushSquare(i, j);
27 }
28 void updTile(int i, int j)
29 {
30     if ((0 <= i) && (i < n) && (0 <= j) && (j < m))
31         if (!table[i][j])
32             table[i][j] = 1;
33             pushNeighbors(i, j);
34 }
35
36 void updSquare(int i, int j)
37 {
38     updTile(i, j);
39     updTile(i + 1, j);
40     updTile(i, j + 1);
41     updTile(i + 1, j + 1);
42 }
43
44 int main()
45 {
46     // freopen("f.in", "r", stdin);
47     //     freopen("f.out", "w", stdout);
48     cin >> n >> m;
49     char c;
50     for (int i = 0; i < n; i++)
51     {
52         for (int j = 0; j < m; j++)
53         {
54             cin >> c;
55             if (c == '#')
56                 table[i][j] = 1;
57         }
58     }
59     for (int i = 0; i < n - 1; i++)
60         for (int j = 0; j < m - 1; j++)
61             if (sumSquare(i, j) == 3)

```

```
62         st.push_back(make_pair(i, j));
63     while (!st.empty())
64     {
65         pair < int, int > p = st.back();
66         st.pop_back();
67         if (sumSquare(p.first, p.second))
68             updSquare(p.first, p.second);
69     }
70     int answer = 0;
71     for (int i = 0; i < n; i++)
72         for (int j = 0; j < m; j++)
73             answer += table[i][j];
74     cout << answer << endl;
75     return 0;
76 }
77
```