РАЗБОР ЗАДАЧ

Открытый чемпионат ЧГУ
по программированию

2021

A – Точное время

```
String time = in.nextLine().trim();
String revertedTime = time.substring(3, 4) + time.substring(2, 3)
                    + time.substring(1, 2) + time.substring(0, 1);

if (time.equals(revertedTime)) {
    out.print("YES");
} else {
    Pattern pattern = Pattern.compile("(([0-1][0-9])|(2[0-3]))[0-5][0-9]");
    if (pattern.matcher(time).matches() &&
pattern.matcher(revertedTime).matches()) {
        out.print("NO");
    } else {
        out.print("YES");
    }
}
```

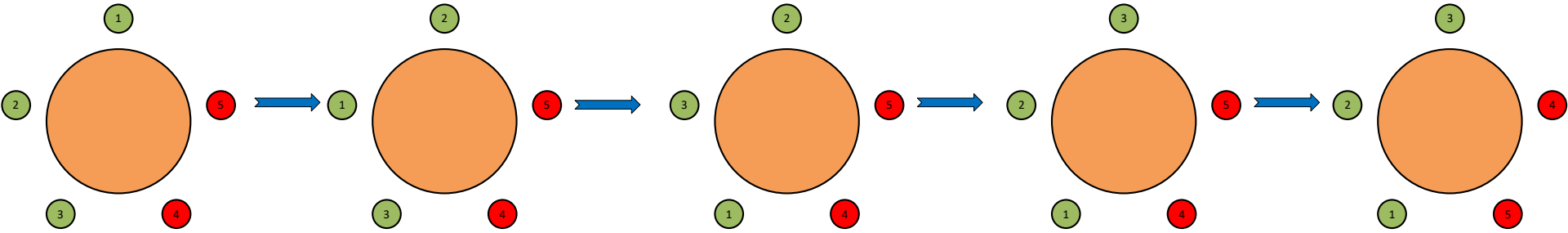
Алгоритм решения

1. Если «обычное» время и «перевёрнутое» совпадают, то мы можем точно назвать время. Ответ **YES**.
2. Если «обычное» время и «перевёрнутое» различаются, то возможны два варианта:
   A. Если «обычное» и «перевёрнутое» время могут являются корректным временем от 00:00 до 23:59, то мы не можем однозначно сказать, какое сейчас время на самом деле. Ответ **NO**.
   B. В противном случае, только одно из «обычного» или «перевёрнутого» времён является корректным временем. Ответ **YES**.

B – Неправильно сели

Алгоритм решения

1. Разделим всех участников заседания на две группы, в которых количество участников либо совпадает, либо отличается на 1.
2. Отсортируем каждую группу по отдельности. Это потребует N*(N-1)/2 пересаживаний, где N – количество людей в группе (вспомните алгоритм сортировки «пузырьком»).
3. Находим сумму пересадок для обеих групп. Это и будет ответом.




```
int n = in.nextInt();
int a = n / 2;
int b = n - a;
out.print((a * (a - 1) / 2) + (b * (b - 1) / 2));
```

C – Шах и мат

			W				
			W				
W	W	W		W	W	W	W
W	W	W	B	W	W	W	W
B	B	B	B	B	B	B	B
B	B	B		B	B	B	
							B

Массив w
Координаты (номер горизонтали) белых фигур

3	3	3	1	3	3	3	3
4	4	4	2	4	4	4	4

Массив b
Координаты (номер горизонтали) чёрных фигур

5	5	5	4	5	5	5	5
6	6	6	5	6	6	6	7

```
int steps = 0;
for (int i = 0; i < 8; i++) {
    steps += (b[i][0] - w[i][1] - 1) + (b[i][1] - w[i][0] - 3);
}

out.print(steps);
```

D – Путешествие

```
int n = in.nextInt();
int m = in.nextInt();
in.nextLine();

String[] king = new String[n];
String[] queen = new String[m];
for (int i = 0; i < n; i++) {
    king[i] = in.nextLine();
}
for (int i = 0; i < m; i++) {
    queen[i] = in.nextLine();
}

int[][] t = new int[n + 1][m + 1];
for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
        if (king[i - 1].equals(queen[j - 1])) {
            t[i][j] = Math.max(t[i - 1][j - 1] + 1, Math.max(t[i - 1][j], t[i][j - 1]));
        } else {
            t[i][j] = Math.max(t[i - 1][j], t[i][j - 1]);
        }
    }
}

out.print(t[n][m]);
```

Алгоритм нахождения наибольшей общей подпоследовательности

E – Реновация

Пусть X - высота первого дома, а всего домов в комплексе i.
Тогда X + (X + 1) + … + (X + i - 1) = N.
i*X + (0 + 1 + … + (i-1)) = N
i*X + i*(i - 1) / 2 = N
i*X = N - i*(i - 1) / 2
Так как X – целое число, то значит N - i*(i - 1) / 2 должно делиться на i.

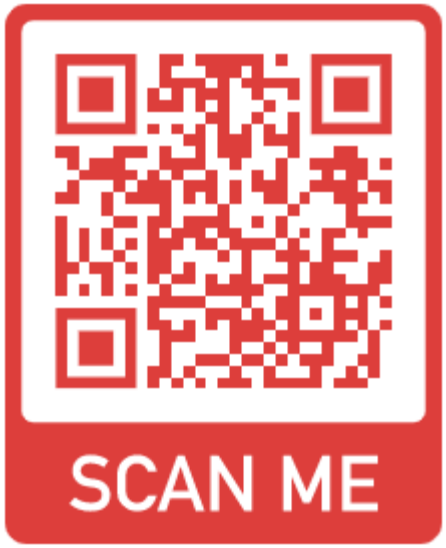
```
int n = in.nextInt();
int cnt = 0;
for (int i = 2; i * (i - 1) / 2 < n; i++) {
    if ((n - i * (i - 1) / 2) % i == 0) {
        cnt++;
    }
}
out.print(cnt);
```

F - Деревья

Алгоритм решения

1. Восстанавливаем по описаниям деревья.
2. Для обоих деревьев сортируем для каждого узла дерева его поддеревья, используя операцию сравнения, которая задаёт полный порядок.
3. Сравниваем получившиеся отсортированные деревья. Если структура деревьев совпадает, то ответ YES, в противном случае NO.

Условия, тесты, решения



https://github.com/peneksglazami/chsu-contest-2021