Тестовое задание на должность
Junior Python Developer

https://hh.ru/vacancy/49945904

HR

Андрей Филимонов

filimonov@si-security.ru

Результаты тестового задания направляйте на вышеуказанную почту. Дополнительные материалы можно размещать на сторонних ресурсах, например, Github.

Пожелание к выполнению заданий

Код следует писать так, как будто это решения реальных задач, которые предстоит поддерживать вам и вашей команде, т.е. комментарии там, где это целесообразно, приветствуются. Задачи обязательно должны сопровождаться тестами, которые будут доказывать, что решение работает.

Удачи!

Задача 1

Задача 2

Задача 1

На автоматизированном складе есть линия хранения, на которой хранятся ящики с номерами. Ящики на линии хранятся в один длинный ряд. Новые ящики поступают могут быть загружены на вход или на выход. Выгрузка возможна только через выход.

Каждое утро на склад поступает программа поступления и выдачи, состоящая из команд одного из двух видов:

• принять x – загрузить ящик номер x. Ящик может быть загружен сверху или снизу;

• выгрузить x – выгрузить ящик с номером x. Система находит ящик с номером x, выгружает все ящики, стоящие ближе этого к ящика к выходу, выгружает ящик x обратно в том же порядке.

Запросы на отгрузку стоят 1 ед. энергии. Запросы 2-го типа - 2k + 1 единиц энергии, где k – количество ящиков, которые стоят ближе к выходу от выгружаемого (то есть система сначала достаёт k ящиков, потом выгружает нужный, потом загружает вынутые ящики обратно).

Нужно написать функцию, которая, сможет рассчитать минимальное количество энергии может потратить на выполнение программы.

Сигнатура функции:

```
def analyse(programm: list[tuple[str, x]] → int
```

Гарантируется, что не может быть ящиков с одинаковыми номерами и выгружаемый ящик всегда находятся на складе на момент выгрузки.

Пример вызова 1:

```
analyse([
    ('принять', 46),
    ('выгрузить', 46),
    ('принять', 21),
    ('выгрузить', 21),
])
```

Возвращаемое значение 4 (по одному действию на каждую команду)

Пример вызова 2:

```
analyse([
    ('принять', 1),
    ('принять', 2),
    ('выгрузить', 1),
    ('принять', 3),
    ('принять', 4),
    ('выгрузить', 3),
])
```

1 ящик загружаем на выход, 2 на вход, 3 на выход, 4 на вход. Тогда расход энергии будет 6 единиц.

Задача 2

Производитель крутых хакерских ноутбуков объявил о рекламной акции. Продолжительность акции **m** дней. По условиям акции участники должны заходить на сайт кампании и нажимать кнопку «Получить бонус».

Кнопку можно нажимать не чаще одного раза в день. За каждое нажатие участник увеличивает свой уровень начиная с 1-го. Если участник пропускает день его уровень сбрасывается до единицы.

Награда зависит от уровня и награды заранее известны, и составляют **p1, p2, ..., pn** бонусов, где **px** - награда участника уровня **x**. После получения награды за уровень **n** уровень участника сбрасывается обратно до единицы.

Необходимо написать функцию которая будет принимать числа n и m и массив **p1, p2, ..., pn** и возвращать максимальное количество бонусов который может получить участник.

Сигнатура функции:

```
def calculate(m: int, n: int, p: list[int])
```

Пример 1:

входные данные: m=3, n=3, p=[6, 2, 3]

выходные данные: 12

Оптимальное решение - нажимать кнопку в первый и третий день.

Пример 2:

входные данные: m=3, n=3, p=[2, 4, 8]

выходные данные: 14

Оптимальное решение - нажимать кнопку каждый день.