|  |  |  |
| --- | --- | --- |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное автономное  образовательное учреждение высшего образования  «Пермский государственный национальный  исследовательский университет» | | |
|  | Институт компьютерных наук и технологий | |
| **ОТЧЁТ**  по индивидуальной работе №2  по дисциплине «Язык программирования Python»  Вариант 1 | | |
|  | | Работу выполнил  студент группы ИТ-5-2024 1 курса  Онин Д. А  «12» Июня 2025 г. |
| Работу проверил  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Фамилия И.О.  «\_\_\_» \_\_\_\_\_\_\_\_\_ 2024 г. |
| Пермь 2024 | | |

# Содержание

[Содержание 1](#_Toc618071362)

[Постановка задачи 1](#_Toc888527834)

[Ключевая идея 2](#_Toc2023386860)

[Алгоритм решения 4](#_Toc1252911700)

[Тестирование 5](#_Toc1284393976)

[Код программы 5](#_Toc77401058)

# Постановка задачи

Баржа. На барже располагается K грузовых отсеков. В каждый отсек можно поместить некоторое количество бочек с одним из 10 000 видов топлива. Причём извлечь бочку из отсека можно лишь в случае, если все бочки, помещённые в этот отсек после неё, уже были извлечены. Таким образом, в каждый момент времени в каждом непустом отсеке имеется ровно одна бочка, которую можно извлечь, не трогая остальных. Будем называть такие бочки крайними. Изначально баржа пуста. Затем она последовательно проплывает через N доков, причём в каждом доке на баржу либо погружается бочка с некоторым видом топлива в некоторый отсек, либо выгружается крайняя бочка из некоторого отсека. Однако, если указанный отсек пуст, либо если выгруженная бочка содержит не тот вид топлива, который ожидалось, следует зафиксировать ошибку. Если на баржу оказывается погружено более P бочек или если после прохождения всех доков она не стала пуста, следует также зафиксировать ошибку. От вас требуется либо указать максимальное количество бочек, которые одновременно пребывали на барже либо зафиксировать ошибку.

Входные данные: в первой строке три целых числа N, K и P (1 ≤ N, K, P ≤ 100000). Далее следует N строк с описанием действия, выполняемого в очередном доке. Если в нём происходит погрузка, то строка имеет вид «+ A B», где A — номер отсека, в который помещается бочка, а B — номер вида топлива в ней. Если же док занимается разгрузкой, то строка имеет вид «- A B», где A — номер отсека, из которого извлекается бочка, а B — номер ожидаемого вида топлива.

Выходные данные: вывести либо одно число, равное искомому максимуму в случае безошибочного прохождения баржой маршрута, либо вывести слово «Error» в противном случае. Примеры

Пример:

(входные данные)

6 1 2

+ 1 1

+ 1 2

- 1 2

- 1 1

+ 1 3

- 1 3

2 (выходные данные)

# Ключевая идея

Моделировать отсеки баржи с помощью K стеков, отслеживать текущее и максимальное количество бочек, а также фиксировать ошибки при некорректных операциях (пустой отсек, неверное топливо, превышение лимита, остаток бочек).

# Алгоритм решения

1. **Инициализация:**

* **Считать входные данные: N** (число операций), **K** (число отсеков), **P** (максимальное число бочек), где **1 ≤ N, K, P ≤ 100 000**.
* Создать **K** пустых стеков, по одному для каждого отсека.
* Инициализировать переменные:
  + **current\_barrels** = 0 — текущее количество бочек на барже.
  + **max\_barrels** = 0 — максимальное количество бочек за всё время.
  + **error** = False — флаг ошибки.

2.**Обработка операций:**

* Для каждой из **N** операций:
  + Если операция — **погрузка** («+ A B»):
    - Проверить, что номер отсека **A** корректен: **1 ≤ A ≤ K**. Если нет, установить **error = True**.
    - Добавить вид топлива **B** в стек отсека **A**.
    - Увеличить **current\_barrels** на 1.
    - Если **current\_barrels > P**, установить **error = True**.
    - Обновить **max\_barrels = max(max\_barrels, current\_barrels)**.
  + Если операция — **разгрузка** («- A B»):
    - Проверить, что номер отсека **A** корректен: **1 ≤ A ≤ K**. Если нет, установить **error = True**.
    - Если стек отсека **A** пуст, установить **error = True**.
    - Иначе извлечь верхнюю бочку из стека отсека **A** и проверить:
      * Если извлечённая бочка не равна ожидаемому виду топлива **B**, установить **error = True**.
    - Уменьшить **current\_barrels** на 1.

**3.Проверка результата:**

* После обработки всех операций проверить, пусты ли все **K** стеков.
  + Если хотя бы один стек непуст, установить **error = True**.
* Если **error = True**, вывести «Error».
* Иначе вывести **max\_barrels**.

# Тестирование

|  |  |  |  |
| --- | --- | --- | --- |
| Тестовый пример | Входные данные | Ожидаемый результат | Описание |
| 1 | 6 1 2  + 1 1  + 1 2  - 1 2  - 1 1  + 1 3  - 1 3 | Максимальное количество бочек на барже: 2 | Корректная последовательность операций |
| 2 | 3 1 1  + 1 1  + 1 2  - 1 2 | Error | Превышение максимального числа бочек |
| 3 | 2 1 2  + 2 1  - 1 1 | Error | Неверный индекс отсека |
| 4 | 2 1 2  + 1 1  + 1 2 | Error | Баржа не пуста в конце |
| 5 | 3 1 2  + 1 1  + 1 2  - 1 1 | Error | Неверный тип топлива при выгрузке |
| 6 | 4 2 3  + 1 1  + 2 2  - 2 2  - 1 1 | Максимальное количество бочек на барже: 2 | Работа с несколькими отсеками |
| 7 | 1 1 1  - 1 1 | Error | Выгрузка из пустого отсека |
| 8 | 5 3 4  + 1 1  + 2 2  + 3 3  + 4 1  - 4 1 | Error | Неверный индекс отсека (4 > 3) |
| 10 | 3 1 2  + 1 1  + 1 1  - 1 2 | Error | Максимальные допустимые значения |

# Код программы

class Stack:  
 def \_\_init\_\_(self):  
 self.\_items = []  
  
 def push(self, item):  
 self.\_items.append(item)  
  
 def pop(self):  
 if self.is\_empty():  
 raise ValueError("Попытка извлечь бочку из пустого отсека")  
 return self.\_items.pop()  
  
 def is\_empty(self):  
 return len(self.\_items) == 0  
  
 def peek(self):  
 if self.is\_empty():  
 return None  
 return self.\_items[-1]  
  
  
class Barge:  
 def \_\_init\_\_(self, compartments\_count, max\_barrels):  
 if not (1 <= compartments\_count <= 100000):  
 raise ValueError("Количество отсеков должно быть от 1 до 100,000")  
 if not (1 <= max\_barrels <= 100000):  
 raise ValueError("Максимальное количество бочек должно быть от 1 до 100,000")  
 self.\_compartments = [Stack() for \_ in range(compartments\_count)]  
 self.\_max\_barrels = max\_barrels  
 self.\_current\_barrels = 0  
 self.\_max\_seen\_barrels = 0  
 self.\_error = False  
  
 def load\_barrel(self, compartment, fuel\_type):  
 if not (1 <= compartment <= len(self.\_compartments)):  
 self.\_error = True  
 return  
 self.\_compartments[compartment - 1].push(fuel\_type)  
 self.\_current\_barrels += 1  
 if self.\_current\_barrels > self.\_max\_barrels:  
 self.\_error = True  
 self.\_max\_seen\_barrels = max(self.\_max\_seen\_barrels, self.\_current\_barrels)  
  
 def unload\_barrel(self, compartment, expected\_fuel):  
 if not (1 <= compartment <= len(self.\_compartments)):  
 self.\_error = True  
 return  
 stack = self.\_compartments[compartment - 1]  
 if stack.is\_empty():  
 self.\_error = True  
 return  
 fuel = stack.pop()  
 if fuel != expected\_fuel:  
 self.\_error = True  
 self.\_current\_barrels -= 1  
  
 def has\_error(self):  
 return self.\_error  
  
 def is\_empty(self):  
 return all(stack.is\_empty() for stack in self.\_compartments)  
  
 def get\_max\_barrels(self):  
 return self.\_max\_seen\_barrels  
  
  
class BargeSimulator:  
 def \_\_init\_\_(self):  
 self.\_barge = None  
  
 def run(self):  
 print("Введите параметры и операции согласно формату:\n"  
 "Первая строка: N K P (кол-во операций, отсеков, макс. бочек)\n"  
 "Далее N строк: '+ A B' для погрузки или '- A B' для разгрузки\n"  
 "Пример:\n6 1 2\n+ 1 1\n+ 1 2\n- 1 2\n- 1 1\n+ 1 3\n- 1 3")  
 # Запрос параметров N, K, P  
 while True:  
 try:  
 input\_str = input("Введите N K P(Три целых числа от 1 до 100,000 через пробел): ").strip()  
 n, k, p = map(int, input\_str.split())  
 if not (1 <= n <= 100000 and 1 <= k <= 100000 and 1 <= p <= 100000):  
 raise ValueError("N, K, P должны быть от 1 до 100,000")  
 self.\_barge = Barge(k, p)  
 break  
 except ValueError:  
 print("Ошибка: некорректный ввод. Введите три целых числа от 1 до 100,000 через пробел.")  
  
 # Запрос операций  
 operations = []  
 for i in range(n):  
 while True:  
 try:  
 op = input(f"Операция {i + 1}: ").strip().split()  
 if len(op) != 3 or op[0] not in ['+', '-']:  
 raise ValueError("Неверный формат. Используйте '+ A B' или '- A B'")  
 action, a, b = op[0], int(op[1]), int(op[2])  
 operations.append((action, a, b))  
 break  
 except ValueError:  
 print("Ошибка: некорректная операция. Введите '+ A B' или '- A B', где A и B — целые числа.")  
  
 # Выполнение операций  
 for action, a, b in operations:  
 if action == '+':  
 self.\_barge.load\_barrel(a, b)  
 elif action == '-':  
 self.\_barge.unload\_barrel(a, b)  
  
 # Вывод результата  
 if self.\_barge.has\_error() or not self.\_barge.is\_empty():  
 print("Error")  
 else:  
 print(f"Максимальное количество бочек на барже: {self.\_barge.get\_max\_barrels()}")  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 simulator = BargeSimulator()  
 simulator.run()