САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ ФАКУЛЬТЕТ ИНФОКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

Отчет по лабораторной работе № по курсу «Алгоритмы и структуры данных»

Тема: Хеширование. Хеш-таблицы

Вариант 21

Выполнила:

Савченко А.С.

Санкт-Петербург 2024 г.

Содержание отчета

Содержание отчета	2
Задачи по варианту	3
ВЫЧИСЛЕНИЕ ВАРИАНТА	3
Задача №1. Множество	5
Задача №6. Выборы в США	10
Задача №8. Почти интерактивная хеш-таблица	12
Вывод (по всей лабораторной)	16

Задачи по варианту

ВЫЧИСЛЕНИЕ ВАРИАНТА

Текст задачи:

Базовый уровень. Решается 3 задачи, причем должно выполниться два условия:

- Все три задачи не могут стоять подряд в задании. Т.е. решать набор (1,2,3) - нельзя, (1,2,4) - можно
- Номер одной из задач находится по следующей хэш-функции:

$$H(v) = (A \cdot v \mod p) \mod 9,$$

где A - последние 2 числа номера вашей группы, v - ваш номер в списке группы (вариант), p - следующее простое число, которое больше чем количество человек в вашей группе.

(опционально) Номер второй задачи можете найти по принципу:

$$H(v) = ((A \cdot v + B) \bmod p) \bmod 9,$$

где B - это сумма кодов ASCII всех букв вашей фамилии \odot

В этом случае максимум за защиту можно получить 4 балла. В сумме с самой работой (0,5 балла) и отчетом (1 балл) получается 5,5 балла, что достаточно для зачета.

Листинг кола:

```
def var_generator(group_num, group_size, surname,
    student_number):
    A = int(str(group_num)[-2:]) # Последние 2 цифры
    HOMEPA ГРУППЫ

def next_prime(n):
    def is_prime(num):
        if num < 2:
            return False
        for i in range(2, int(num ** 0.5) + 1):
            if num % i == 0:
                return False
            return False
            return True</pre>
```

```
prime = n + 1
      while not is prime(prime):
          prime += 1
      return prime
    p = next prime(group size) # Следующее простое
   # Сумма кодов ASCII всех букв фамилии
  B = sum(ord(char) for char in surname)
  task1 = (A * student number % p) % 9
  task2 = ((A * student number + B) % p) % 9
   # Проверка на подряд идущие задачи
  if task1 == task2 or task1 + 1 == task2:
      task2 = (task2 + 2) % 9
  task3 = (task1 + 2) % 9
   if task3 == task2 or task3 + 1 == task2 or task3
1 == task2 or task3 + 1 == task1 or task3 - 1 ==
task1:
      task3 = (task3 + 2) % 9
  return task1, task2, task3
group num = 3241
group size = 35
surname = "Савченко"
student number = 25
task1, task2, task3 = var generator(group num,
group size, surname, student number)
print(f"Номер первой задачи: {task1}")
```

```
print(f"Номер второй задачи: {task2}")
print(f"Номер третьей задачи: {task3}")
```

ВАРИАНТ:

```
C:\Users\Anastasia\PycharmProjects\1sem_labs\.venv
Номер первой задачи: 8
Номер второй задачи: 6
Номер третьей задачи: 1
```

Задача №1. Множество

Текст задачи:

1 задача. Множество

Реализуйте множество с операциями «добавление ключа», «удаление ключа», «проверка существования ключа».

- Формат входного файла (input.txt). В первой строке входного файла находится строго положительное целое число операций N, не превышающее 5 · 10⁵. В каждой из последующих N строк находится одна из следующих операций:
 - А x добавить элемент x в множество. Если элемент уже есть в множестве, то ничего делать не надо.
 - D x удалить элемент x. Если элемента x нет, то ничего делать не напо.
 - ?x если ключ x есть в множестве, выведите «Y», если нет, то выведите «N»

Аргументы указанных выше операций – **целые числа**, не превышающие по модулю 10^{18} .

- Формат выходного файла (output.txt). Выведите последовательно результат выполнения всех операций «?». Следуйте формату выходного файла из примера.
- Ограничение по времени. 2 сек.
- Ограничение по памяти. 256 мб.
- Пример:

input.txt	output.txt
8	Y
A 2	N
A 5	N
A 3	
? 2	
? 4	
A 2	
D 2	
? 2	

• Примечание.

Эту задачу можно решить совершенно разными способами, включая использование различных средств стандартных библиотек (правда, не всех - в стандартных библиотеках некоторых языков программирования используются слишком предсказуемые методы хеширования). Именно по этой причине ее разумно использовать для проверки реализаций хеш-таблиц, которые понадобятся в следующих задачах этой работы.

Листинг кода:

```
class HashTable:
   def __init__(self):
      self.table = set()
```

```
def add key(self, key):
       self.table.add(key)
  def remove key(self, key):
       self.table.discard(key)
  def check key(self, key):
       return key in self.table
def my set(operations):
  hash table = HashTable()
  results = []
   for operation in operations:
       splited = operation.split()
       op type = splited[0]
       key = int(splited[1])
      if op type == 'A':
          hash table.add key(key)
       elif op type == 'D':
          hash table.remove key(key)
       elif op type == '?':
           if hash table.check key(key):
               results.append('Y')
           else:
               results.append('N')
   return results
if name == " main ":
  with open('input.txt', 'r') as input file:
      n = int(input file.readline().strip())
       operations = [input file.readline().strip() for
 in range(n)]
  ans = my set(operations)
```

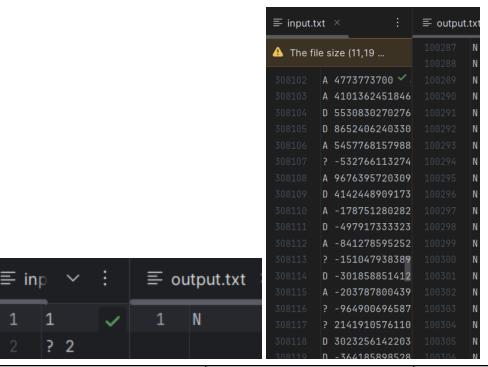
```
with open('output.txt', 'w') as output_file:
   output_file.write("\n".join(ans) + "\n")
```

Текстовое объяснение решения:Из входного файла считываем списком строк операции и число-ключ. В функцию передаем считанный массив строк, создаем пустой результирующий список. Для каждой операции из списка отделаем тип операции от числового ключа, в зависимости от типа операции либо добавляем число в множество, либо удаляем методом discard, если тип операции? проверяем наличие элемента в множестве и помещаем да или нет в результирующий список, который потом выводим в выходной файл в нужном формате.

Результат работы кода на примерах из текста задачи:(скрины input output файлов):



Результат работы кода на минимальных и максимальных значениях:(скрины input output файлов):



	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из текста задачи	0.0005755 c	14.8359375 Мб
Пример из задачи	0.0009368 c	14.73046875 Мб
Верхняя граница диапазона значений входных данных из текста задачи	0.3517592 c	17.94921875 Мб

Вывод по задаче:

В данной задаче я реализовала множество с операциями «добавление ключа», «удаление ключа», «проверка существования ключа», используя встроенного типа данных set() в Python. Операции добавления, удаления и проверки существования выполняются в среднем за O(1) благодаря хеш-таблицам, используемым в set.

Задача №6. Выборы в США

Текст задачи:

Как известно, в США президент выбирается не прямым голосованием, а путем двухуровневого голосования. Сначала проводятся выборы в каждом штате и определяется победитель выборов в данном штате. Затем проводятся государственные выборы: на этих выборах каждый штат имеет определенное число голосов — число выборщиков от этого штата. На практике, все выборщики от штата голосуют в соответствии с результами голосования внутри штата, то есть на заключительной стадии выборов в голосовании участвуют штаты, имеющие различное число голосов. Вам известно за кого проголосовал каждый штат и сколько голосов было отдано данным штатом. Подведите итоги выборов: для каждого из участника голосования определите число отданных за него голосов.

- Формат ввода / входного файла (input.txt). Каждая строка входного файла содержит фамилию кандидата, за которого отдают голоса выборщики этого штата, затем через пробел идет количество выборщиков, отдавших голоса за этого кандидата.
- Формат вывода / выходного файла (output.txt). Выведите фамилии всех кандидатов в лексикографическом порядке, затем, через пробел, количество отданных за них голосов.
- Ограничение по времени. 2 сек.
- Ограничение по памяти. 64 мб.
- Примеры:

№	input.txt	output.txt
1	McCain 10	McCain 16
	McCain 5	Obama 17
	Obama 9	
	Obama 8	
	McCain 1	

№	input.txt	output.txt
2	ivanov 100	ivanov 900
	ivanov 500	petr 70
	ivanov 300	tourist 3
	petr 70	
	tourist 1	
	tourist 2	

№	input.txt	output.txt
3	bur 1	bur 1

Листинг кода:

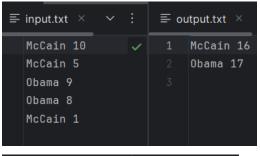
```
cands = {}
with open('input.txt', 'r') as input_file:
   for line in input_file:
```

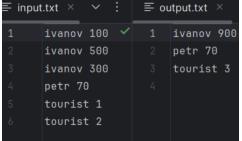
```
name, votes = line.split()
  votes = int(votes)
  if name in cands:
      cands[name] += votes
  else:
      cands[name] = votes
  sorted_cands = sorted(cands.items())
with open('output.txt', 'w') as output_file:
  for name, votes in sorted_cands:
    output_file.write(f"{name} {votes}\n")
```

Текстовое объяснение решения:

Создаем слов где будем хранить голоса за каждого из кандидатов. Чистаем в цикле строки из файла и разделяем их на имя кандидата и кол-во голосов. Если имея уже есть в словаре по по ключу добавляем голоса, в противном случае добавляем имя и кол-во голосов. Сортируем кандидатов в лексикографическом порядке, вывод в файл отсортированный список кортежей в нужном формате.

Результат работы кода на примерах из текста задачи:(скрины input output файлов):





Результат работы кода на минимальных и максимальных значениях:(скрины input output файлов):

	Время выполнения	Затраты памяти
Пример из задачи	0.0006989 c	14.9921875 Мб
Пример из задачи	0.0008013 c	14.85546875 Мб
Пример из задачи	0.000775 c	14.9765625 Мб

Задача №8. Почти интерактивная хеш-таблица

Текст задачи:

8 задача. Почти интерактивная хеш-таблица

В данной задаче у Вас не будет проблем ни с вводом, ни с выводом. Просто реализуйте быструю хеш-таблицу.

В этой хеш-таблице будут храниться целые числа из диапазона $[0;10^{15}-1]$ Требуется поддерживать добавление числа x и проверку того, есть ли в таблице число x. Числа, с которыми будет работать таблица, генерируются следующим образом. Пусть имеется четыре целых числа N, X, A, B такие что:

- $1 \le N \le 10^7$
- $1 \le X \le 10^{15}$
- $1 \le A \le 10^3$
- 1 < B < 10¹⁵

Требуется N раз выполнить следующую последовательность операций:

- Если X содержится в таблице, то установить $A \leftarrow (A + A_C) \bmod 10^3$, $B \leftarrow (B + B_C) \bmod 10^{15}$.
- Если X не содержится в таблице, то добавить X в таблицу и установиті
 А ← (A + A_D) mod 10³, B ← (B + B_D) mod 10¹⁵.
- Установить X ← (X · A + B) mod 10¹⁵.

Начальные значения X, A и B, а также N, A_C , B_C , A_D и B_D даны во входном файле. Выведите значения X, A и B после окончания работы.

- Формат входного файла (input.txt). В первой строке входного файла со держится четыре целых числа N, X, A, B. Во второй строке содержится еще четыре целых числа A_C, B_C, A_D и B_D такие что $0 \le A_C, A_D < 10^3$ $0 \le B_C, B_D < 10^{15}$.
- Формат выходного файла (output.txt). Выведите значения X, A и B после окончания работы.
- Ограничение по времени. 5 сек.
- Ограничение по памяти. 256 мб.
- Примеры:

input.txt		
4000		
1100		
output.txt		
3 1 1		

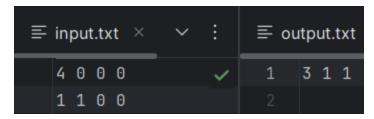
Листинг кода:

```
@njit
def hash table(N, X, A, B, AC, BC, AD, BD):
   table = set()
  for in range(N):
      if X in table:
          A = (A + AC) % 1000
          B = (B + BC) % 10 ** 15
      else:
          table.add(X)
          A = (A + AD) % 1000
          B = (B + BD) % 10 ** 15
      X = (X * A + B) % 10 ** 15
   return X, A, B
if name == " main ":
   with open('input.txt', 'r') as infile:
                               Α,
                      N,
                          Χ,
infile.readline().split())
                    AC, BC, AD, BD = map(int,
infile.readline().split())
  X, A, B = hash table(N, X, A, B, AC, BC, AD, BD)
  with open('output.txt', 'w') as outfile:
      outfile.write(f"{X} {A} {B}")
```

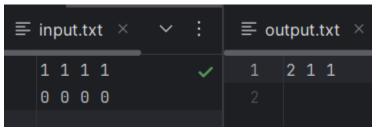
Текстовое объяснение решения:

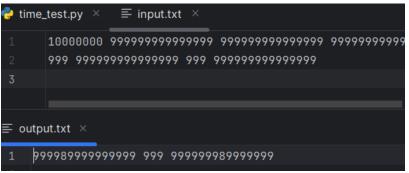
Будем использовать библиотеку numba чтобы уложиться в ограничения времени. @njit - Декоратор, который оптимизирует и ускоряет функцию с помощью ЈІТ-компиляции. Сама функция hash_table: N раз выполняет операции описанные в задаче.

Результат работы кода на примерах из текста задачи:(скрины input output файлов):



Результат работы кода на минимальных и максимальных значениях:(скрины input output файлов):





	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из текста задачи	0.0007799 c	15.0234375 Мб
Пример из задачи	0.0005965 c	14.9453125 Мб
Верхняя граница диапазона значений входных данных из текста задачи(без numba)	7.4637536 c	16.76953125 Мб
Верхняя граница диапазона значений входных данных из	3.026858 с	91.53125 Мб

текста задачи(c numba)		
------------------------	--	--

Вывод по задаче:

Сначала я реализовала алгоритм без сторонних библиотек, но не укладывался в ограничения времени(7.4637536 с), а это значило что либо стоит использовать другой ЯП или сторонние библиотеки, такой вляется numba. С ней код укладывается в ограничения, хоть и требует больше памяти.

Вывод (по всей лабораторной)

При выполнении лабораторной работы я поработала с хэш-таблицами, множествами, и узнала новую библиотеку numba. Самой интересной задачей оказалось вычисление задач варианта)