**Міністерство освіти і науки України**

**Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"**

**Факультет інформатики та обчислювальної техніки**

**Кафедра ІПІ**

**Звіт**

з лабораторної роботи № 2 з дисципліни

«Алгоритми та структури даних 2. Структури даних»

„ **Метод декомпозиції. Пошук інверсій**”

**Виконав(ла)**

(шифр, прізвище, ім'я, по батькові)

*ІП-11 Трикош Іван Володимирович*

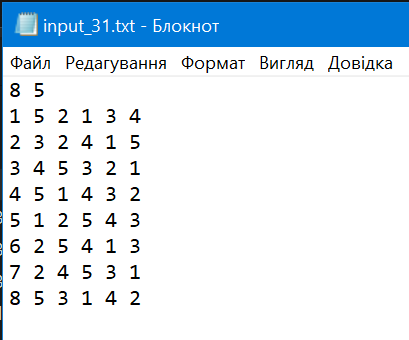
**Перевірив(ла)**

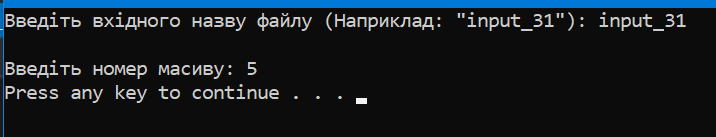
(прізвище, ім'я, по батькові)

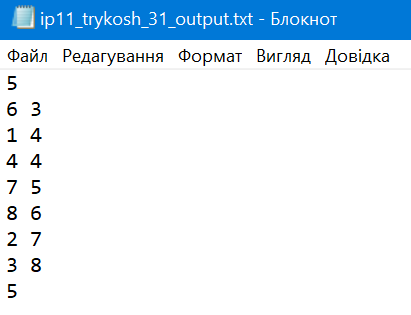
*Халус Олена Андріївна*

Київ 2022

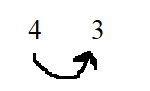
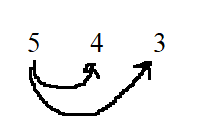
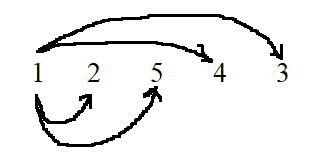
**Приклад роботи**







Обиратимемо пари чисел у масиві так:



Випишемо пари чисел (по порядку та без повторень) з 5-го масиву:

(1, 2) (1, 5) (1, 4) (1, 3) (2, 5) (2, 4) (2, 3) (5, 4) (5, 3) (4, 3)

Випишемо пари чисел (по порядку та без повторень) з інших масивів:

1-ий масив: (5, 2) (5, 1) (5, 3) (5, 4) (2, 1) (2, 3) (2, 4) (1, 3) (1, 4) (3, 4)

2-ий масив: (3, 2) (3, 4) (3, 1) (3, 5) (2, 4) (2, 1) (2, 5) (4, 1) (4, 5) (1, 5)

3-ий масив: (4, 5) (4, 3) (4, 2) (4, 1) (5, 3) (5, 2) (5, 1) (3, 2) (3, 1) (2, 1)

4-ий масив: (5, 1) (5, 4) (5, 3) (5, 2) (1, 4) (1, 3) (1, 2) (4, 3) (4, 2) (3, 2)

6-ий масив: (2, 5) (2, 4) (2, 1) (2, 3) (5, 4) (5, 1) (5, 3) (4, 1) (4, 3) (1, 3)

7-ий масив: (2, 4) (2, 5) (2, 3) (2, 1) (4, 5) (4, 3) (4, 1) (5, 3) (5, 1) (3, 1)

8-ий масив: (5, 3) (5, 1) (5, 4) (5, 2) (3, 1) (3, 4) (3, 2) (1, 4) (1, 2) (4, 2)

Тепер порівнюємо пари чисел у масивах 1-4 та 6-8 з парами чисел з масиву 5. Унікальні пари і будуть інверсіями:

|  |  |
| --- | --- |
| Номер масиву | Унікальні пари |
| 1 | 4 |
| 2 | 7 |
| 3 | 8 |
| 4 | 4 |
| 6 | 3 |
| 7 | 5 |
| 8 | 6 |

І якщо розставити пари чисел номер масиву – унікальні пари, то результат буде збігатися з результатом у файлі:

6 3

1 4

4 4

7 5

8 6

2 7

3 8

Отже, результат роботи програми правильний.

**Код програми**

def sort\_numb\_inv(Array):

lenght = len(Array) # Довжина масиву

mid = lenght // 2 # Середина масиву

if lenght <= 1: # Якщо довжина масиву <= 1, то повертаємо цей масив і кількість інверсій (0)

return Array, 0

else:

LeftPart, leftnumb = sort\_numb\_inv(Array[:mid]) # Викликаємо функцію для лівої частини

RightPart, rightnumb = sort\_numb\_inv(Array[mid:]) # Викликаємо функцію для правої частини

Array, numb = merge\_numb\_inv(LeftPart, RightPart) # Викликаємо функцію для склеювання лівої та правої частин

return Array, leftnumb + rightnumb + numb # Повертаємо масив та кількість інверсій

def merge\_numb\_inv(Leftpart, Rightpart):

lenght\_Leftpart = len(Leftpart) # Довжина лівої частини початкового масиву

lenght\_Rightpart = len(Rightpart) # Довжина правої частини початкового масиву

i = j = 0 # Індекси лівої та правої частин початкового масиву

numb = 0 # Кількість інверсій

max\_value = max(Leftpart) + max(Rightpart) + 1 # Максимальне значення для масивів

Leftpart += [max\_value] # Додаємо максимальне значення в лівий масив

Rightpart += [max\_value] # Додаємо максимальне значення в правий масив

Array = [] # Склеєний масив

for k in range(lenght\_Leftpart + lenght\_Rightpart):

if Leftpart[i] <= Rightpart[j]: # Якщо елемент з лівого масиву менший за елемент з правого масиву, то додаємо його до склеєного масиву

Array += [Leftpart[i]]

i += 1

else: # Інакше додаємо елемент з правого масиву і збільшуємо кількість інверсій

Array += [Rightpart[j]]

j += 1

numb += lenght\_Leftpart - i # Збільшуємо кількість інверсій

return Array, numb # Повертаємо склеєний масив та кількість інверсій

def change(listofusers, usernumber):

for i in range(len(listofusers)):

if i != usernumber:

k = 0 # Індекс елементів головного масиву (з індексом 'usernumber')

for j in range(len(listofusers[i])):

while listofusers[i][j] != listofusers[usernumber][k]: # Проходимо до головному масиву і шукаємо співпадіння елементів

k += 1 # Збільшуємо індекс

else: # Після зевершення циклу записуємо індекс шуканого елемента з головного масиву в масив і обнуляємо лічильник

listofusers[i][j] = k + 1

k = 0

return listofusers

def main():

filename1 = input("Введіть вхідного назву файлу (Наприклад: \"input\_31\"): ") + ".txt" # Вводимо назву файлу

filename2 = "ip11\_trykosh\_31\_output.txt"

fileIn = open(filename1, "r")

fileOut = open(filename2, "w")

infaboutusers = fileIn.readline()

listofusers = fileIn.readlines()

#Створюємо масив масивів і зводимо його до цілих чисел

infaboutusers = infaboutusers.split()

for i in range(int(infaboutusers[0])):

listofusers[i] = listofusers[i].split()

for i in range(len(listofusers)):

listofusers[i] = listofusers[i][1:]

for j in range(len(listofusers[i])):

listofusers[i][j] = int(listofusers[i][j])

#print("Початкові масиви:", \*listofusers, sep = "\n")

usernumber = int(input("\nВведіть номер масиву: ")) - 1 # Вводимо номер

numbofinv = 0 # Кількість інверсій

listofusers = change(listofusers, usernumber) # Змінюємо масиви

#print("\nЗмінені масиви:", \*listofusers, sep = "\n", end = "\n\n")

listofusers[usernumber], numbofinv = sort\_numb\_inv(listofusers[usernumber]) # Сортуємо головний масив

#print("Змінений сортований головний масив: ", listofusers[usernumber], "\nВідсортовані масиви та кількість в них інверсій", sep = "\n", end = "\n") # Виводимо головний масив

l = [] # Список з номером масиву та кількістю інверій

for i in range(int(infaboutusers[0])):

if i != usernumber:

#print(i + 1,':',listofusers[i], '- ', end = '')

listofusers[i], numbofinv = sort\_numb\_inv(listofusers[i]) # Сортуємо масиви та рахуємо інверсії

#print(numbofinv)

l += [[i + 1, numbofinv]]

# Сортуємо списки за зростанням останнього елемента

for i in range(len(l)):

for j in range(len(l) - 1):

if l[j][1] > l[j + 1][1]:

l[j], l[j + 1] = l[j + 1], l[j]

# Записуємо у файл

fileOut.write(str(usernumber + 1) + "\n")

for i in l:

fileOut.write(str(i[0]) + " " + str(i[1]) + "\n")

fileOut.write(str(usernumber + 1))

fileIn.close()

fileOut.close()

if \_\_name\_\_ == "\_\_main\_\_":

main()