Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Дискретна математика

Лабораторна робота №5

«Комбінаторика: перестановки, розміщення, сполучення»

Виконав:

студент групи ІО-93

Руденко Станіслав

Залікова книжка № 9327

Перевірив

Новотарський М.А.

Київ

2020

**Лабораторна робота № 5**

**Тема**: «Комбінаторика: перестановки, розміщення, сполучення»

**Мета роботи**: вивчення правил утворення комбінацій множин: перестановок, розміщень, сполучень.

**Завдання:** Вивчити алгоритми формування перестановок, сполучень та розбиття. Написати програми для виконання даних алгоритмів.

**Теоретичні основи**

**Перестановки**

Комбінації з n елементів, які відрізняються одна від одної тільки порядком елементів, називають перестановками.

Перестановки позначають символом Рn, де n — число елементів, що входять у кожну перестановку. Приклад. Нехай множина М містить три букви А, В, С. Складемо всі можливі комбінації із цих букв: АВС, АСВ, ВСА, CAB, CBA, ВАС (усього 6 комбінацій). Видно, що вони відрізняються одна від одної тільки порядком розташування букв.

Добуток всіх натуральних чисел від 1 до n включно називають n-факторіалом і пишуть: n!= 1 · 2 · 3 · ... · (n - 1) · n. Вважають, що 0! = 1 і n N ∈ . Основна властивість факторіала: (n + 1)! = (n +1) · n!. Отже, число перестановок обчислюємо за формулою: Рn = n!

**Розміщення з повтореннями**

Розміщення з n елементів по k відображають упорядковані комбінації різних елементів множини М, |М|=n. Часто доводиться утворювати упорядковані комбінації з повтореннями деяких елементів. Наприклад, з множини М = {A, Б} 93 можна утворити вісім комбінацій з трьох елементів: ААА, ААБ, АБА, БАА, БАБ, ББА, АББ, БББ. Тут n = 2, k = 3. Такі упорядковані k-комбінації називають кортежами довжини k. Два кортежі (тобто дві загальні комбінації) вважають однаковими, якщо вони мають однакову довжину і на місцях з однаковими номерами стоять однакові елементи.

Кортеж довжини k з n елементів називають розміщенням з повтореннями з n елементів по k .

Дійсно, після заповнення першого місця кортежу довжиною k одним з n елементів (що можливо зробити n варіантами) заповнити друге місце кортежу можна знову будь-яким елементом з усієї множини (повторюючи в одному з варіантів елемент, який знаходиться на першому місці), і так далі k разів.

**Хід роботи**

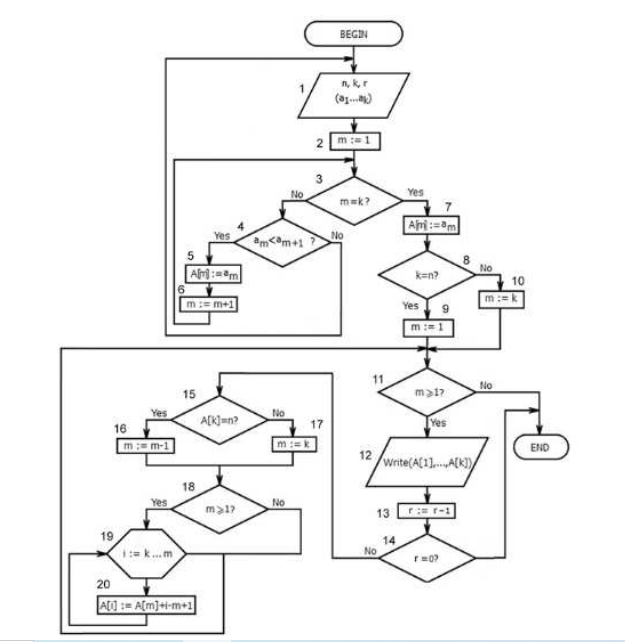
1.Опис структури даних та принципів роботи комбінаторного алгоритму відповідно до варіанту.

**Алгоритм генерації сполучень з n по k на множині**

Нехай дано довільну множину R = { r1, r2 , ,..., n}. Необхідно створити алгоритм, який забезпечує побудову всіх сполучень з n по k на множині R . Для реалізації цього алгоритму модифікуємо попередній алгоритм генерації сполучень у лексикографічному порядку. Модифікація полягає в тому, що при формуванні сполучень будемо використовувати одержані числові сполучення як індекси сполучень елементів множини R .

**Приклад**. Розглянемо генерацію сполучень з 5 по 3 на множині R = { r1 r2 r3 r4 r5}. Для цього сформуємо послідовність A = (1,2,3,4,5) та застосуємо до неї алгоритм генерації сполучень при n = 5 та k = 3. В результаті одержимо сполучення: (1,2,3), (1,2,4), (1,2,5), (1,3,4), (1,3,5), (1,4,5), (2,3,4), (2,3,5), (2,4,5), (3,4,5) )

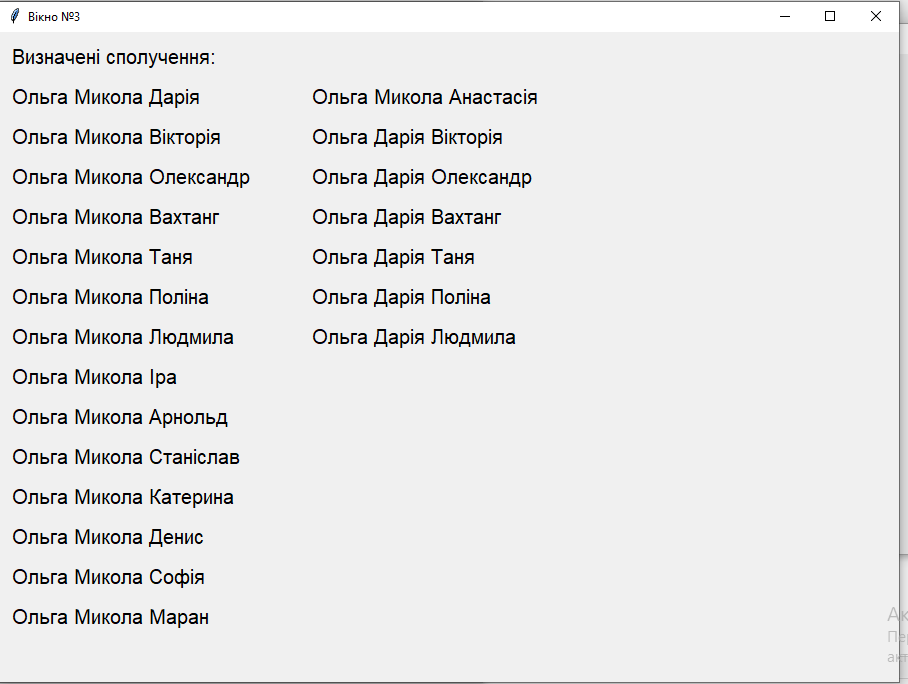
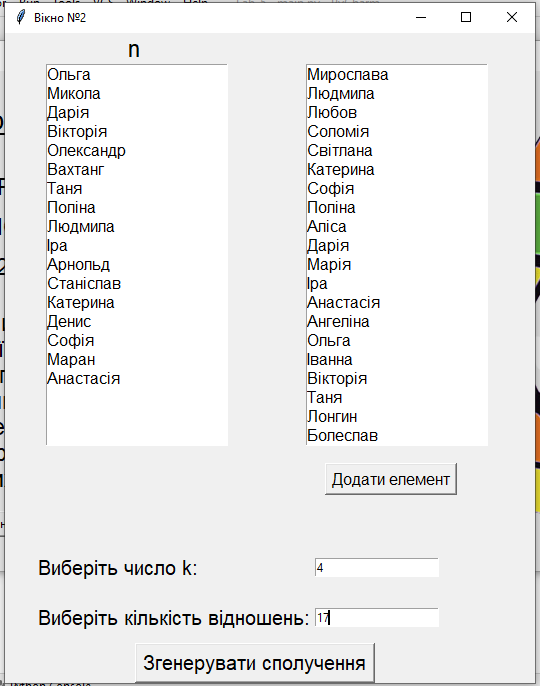
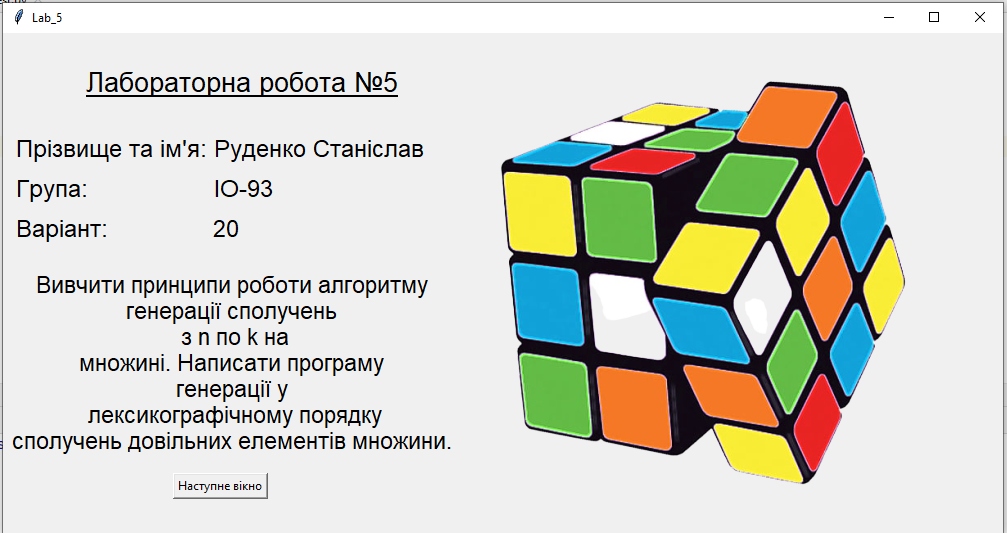
2.Блок схема роботи алгоритму.



3.Код програми

import itertools  
import tkinter as tk  
import tkinter.font as font  
import pickle  
import random  
  
  
class Window1(tk.Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.title(**"Вікно №1"**)  
  
 C = tk.Canvas(self, bg=**"blue"**, height=1000, width=500)  
 filename = tk.PhotoImage(file=**"logo.png"**)  
 background\_label = tk.Label(self, image=filename)  
 background\_label.image = filename  
 background\_label.place(x=200, y=0, relwidth=1, relheight=1)  
 C.pack()  
  
 myFont = tk.font.Font(family=**"Helvetica"**, size=20, underline=True, )  
 title = tk.Label(self, text=**"Лабораторна робота №5"**, font=**"Helvetica 24"**)  
 title[**"font"**] = myFont  
 title.place(x=80, y=30)  
  
 name = tk.Label(text=**"Прізвище та ім'я: Руденко Станіслав"**, font=**"Helvetica 18"**)  
 name.place(x=10, y=100)  
 group = tk.Label(text=**"Група: ІО-93"**, font=**"Helvetica 18"**)  
 group.place(x=10, y=140)  
 var = 9327 % 26 + 1  
 variant = tk.Label(text=**"Варіант: "** + str(var), font=**"Helvetica 18"**)  
 variant.place(x=10, y=180)  
  
 task\_var = **"""  
 Вивчити принципи роботи алгоритму   
 генерації сполучень   
 з n по k на  
 множині. Написати програму   
 генерації у   
 лексикографічному порядку  
 сполучень довільних елементів множини.   
 """** task = tk.Label(text=task\_var, font=**"Helvetica 17"**)  
 task.place(x=0, y=210)  
  
 next\_window = tk.Button(text=**"Наступне вікно"**, command=lambda: Window2())  
 next\_window.place(x=170, y=440)  
  
  
class Window2(tk.Toplevel):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.geometry(**"530x650"**)  
 self.title(**"Вікно №2"**)  
 names = tk.Label(self, text=**"n"**, font=**"Helvetica 17"**)  
 names.place(x=120, y=0)  
 self.set\_n = tk.Listbox(self, width=20, height=20, font=**"Comic 12"**)  
 self.set\_n.place(x=40, y=30)  
  
 self.man\_list = tk.Listbox(self, width=20, height=20, font=**"Comic 12"**)  
 self.man\_list.place(x=300, y=30)  
 self.names\_man = (  
 **"Петро"**, **"Станіслав"**, **"Юрій"**, **"Микола"**, **"Олександр"**, **"Афанасій"**, **"Георгій"**, **"Андрій"**, **"Богдан"**,  
 **"Арнольд"**, **"Лев"**, **"В'ячеслав"**, **"Вахтанг"**, **"Маран"**, **"Денис"**, **"Конрад"**, **"Болеслав"**, **"Лонгин"**, **"Таня"**, **"Вікторія"**,  
 **"Іванна"**, **"Ольга"**, **"Ангеліна"**, **"Анастасія"**, **"Іра"**, **"Марія"**, **"Дарія"**,  
 **"Аліса"**, **"Поліна"**, **"Софія"**, **"Катерина"**, **"Світлана"**, **"Соломія"**, **"Любов"**, **"Людмила"**, **"Мирослава"**)  
 for i in self.names\_man:  
 self.man\_list.insert(0, i)  
  
 used = []  
 for i in range(0, 17):  
 name = random.choice(self.names\_man)  
 while name in used:  
 name = random.choice(self.names\_man)  
 self.set\_n.insert(0, name)  
 used.append(name)  
  
  
 add = tk.Button(self, text=**"Додати елемент"**, font=**"Comic 12"**, command=lambda: self.add\_element())  
 add.place(x=320, y=430)  
  
 global value\_k  
 tk.Label(self, text=**"Виберіть число k:"**, font=**"Helvetica 15"**).place(x=30, y=520)  
 value\_k = tk.Entry(self)  
 value\_k.place(x=310, y=525)  
  
 global number  
 tk.Label(self, text = **"Виберіть кількість відношень:"**, font=**"Helvetica 15"**).place(x=30, y=570)  
 number = tk.Entry(self)  
 number.place(x=310, y=575)  
  
 generate = tk.Button(self, text=**"Згенерувати сполучення"**, font=**"Helvetica 15"**, command=lambda: self.set\_generate())  
 generate.place(x=130, y=610)  
  
 def add\_element(self):  
 selected = self.man\_list.get(self.man\_list.curselection())  
 self.set\_n.insert(tk.END, selected)  
  
 def set\_generate(self):  
 list\_names = self.set\_n.get(0, tk.END)  
 generated\_list = list(map(list, itertools.combinations(list\_names, int(value\_k.get()))))  
 value\_list = list()  
 for i in range(0, int(number.get())):  
 value\_list.append(generated\_list[i])  
 print(value\_list)  
 with open(**'set.txt'**, **'+wb'**) as f:  
 pickle.dump(value\_list, f)  
 Window3()  
  
  
class Window3(tk.Toplevel):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 if int(number.get()) <= 14:  
 self.geometry(**"500x650"**)  
 if int(number.get()) > 14:  
 self.geometry(**"900x650"**)  
  
 self.title(**"Вікно №3"**)  
 with open(**'set.txt'**, **'rb'**) as f:  
 data\_value = pickle.load(f)  
 tk.Label(self, text = **"Визначені сполучення:"**, font=**"Comic 15"**).place(x=10, y=10)  
 coord\_x = 10  
 coord\_y = 50  
 flag = 0  
 for i in data\_value:  
 tk.Label(self, text=i, font=**"Helvetica 15"**).place(x=coord\_x, y=coord\_y)  
 coord\_y+=40  
 flag+=1  
 if flag == 14:  
 coord\_y = 50  
 if value\_k.get() == **"3"**:  
 coord\_x +=300  
 if value\_k.get() == **"4"**:  
 coord\_x += 400  
 if value\_k.get() == **"5"**:  
 coord\_x += 500  
  
  
  
  
app = Window1()  
app.title(**"Lab\_5"**)  
app.geometry(**"1000x500"**)  
app.mainloop()

4.Приклади роботи програми



Висновок:

Ознайомились з роботою алгоритму генерації сполучень з n по k на множині. Розроблені відповідні тестові програми. Результати успішної роботи тестових програм наведені вище та підтверджують правильність обраних рішень – кінцева мета роботи досягнута.