Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №4

за 1 семестр

По дисциплине: «Дискретная математика»

Тема: «Комбинаторика. Группы»

Выполнил:

Студент 2 курса

Группы ПО-4(1)

Иваненко И.Л.

Проверил:

Глущенко Т.А.

2020

Цель работы: “Научиться решать основные задачи комбинаторики”

Вариант 10

***Задание.***

1. *Номер вашего варианта равен остатку от деления вашего порядкового номера по списку на 12. Если ваш порядковый номер кратен 12, выполняете 12 вариант.*

|  |  |  |
| --- | --- | --- |
|  |  |  |

1. *По пункту 4: решить рекуррентное соотношение и программно реализовать вычисление 5 члена числовой последовательности через рекурсию и полученное решение.*



Код программы:

def recurrence\_ratio(a, n, k1, k2):

a.append(k1 \* a[n - 1] - k2 \* a[n - 2])

print(a)

if n == 4:

print("5 член числовой последовательности: " + str(a[n]))

return

recurrence\_ratio(a, n + 1, k1, k2)

recurrence\_ratio([-3, 1], 2, 2, 1)

Результат выполнения:

[-3, 1, 5]

[-3, 1, 5, 9]

[-3, 1, 5, 9, 13]

5 член числовой последовательности: 13

1. *При разложении степени*  *коэффициенты при произведениях* *,* *, рассчитываются по формуле:* *(число перестановок с повторениями) и носят название полиномиальных или мультиномиальных коэффициентов.  
   Например, вычислим коэффициент при произведении*  *в разложении**.  
   Он равен:* *.*
2. *Написать программу генерации перестановок (рекурсивный вариант). Четные номера – реализуют лексикографический порядок генерации. Нечетные номера – реализуют антилексикографический порядок генерации.*

Код программы:

def permutations(arr):

for perm in itertools.permutations(arr):

print(perm)

def antilex\_permutations(arr):

arr = arr[::-1]

for perm in itertools.permutations(arr):

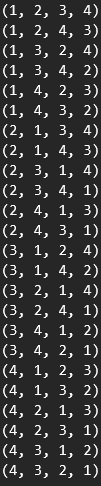
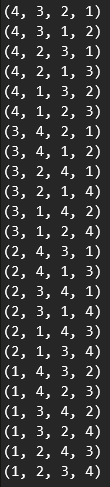
print(perm)

permutations([1, 2, 3, 4])

print("\n\n\n")

antilex\_permutations([1, 2, 3, 4])

Результат выполнения:

1. *Написать программу генерации k- элементных подмножеств n-элементного множества в лексикографическом порядке. Вход: k – мощность подмножества, n - мощность множества.*

Код программы:

def permutations(arr, k = None):

if k == None:

for perm in itertools.permutations(arr):

print(perm)

else:

for perm in itertools.permutations(arr, k):

print(perm)

def antilex\_permutations(arr, k = None):

arr = arr[::-1]

if k == None:

for perm in itertools.permutations(arr):

print(perm)

else:

for perm in itertools.permutations(arr, k):

print(perm)

permutations([1, 2, 3, 4], 2)

print("\n\n\n")

antilex\_permutations([1, 2, 3, 4], 2)

Результат выполнения:

*Вариант 10.*

1. Позывные американских радиостанций состоят из трех или четырех букв и начинаются с *k* или *w*. Сколько существует различных позывных?

В английском алфавите 26 букв. У нас есть 4 разных варианта позывных:

1) Начинается с буквы k и состоит из трех букв, Например: kab

1 \* 26 \* 26

2) Начинается с буквы w и состоит из трех букв, Например: wab

1 \* 26 \* 26

3) Начинается с буквы k и состоит из четырех букв, Например: kabc

1 \* 26 \* 26 \* 26

4) Начинается с буквы w и состоит из четырех букв, Например: wabc

1 \* 26 \* 26 \* 2

Ответ: (26^2 + 26^2 + 26^3 + 26^3 = 2\*26^2 + 3\*26^2 = 1352 + 35152 = 36504)

1. Сколько целых чисел между 1 и 2003 делится на 5, 7 или 11?

Делится на 5: 2003//5 = 400

Делится на 7: 2003//7 = 286

Делится на 11: 2003//11 = 182

Делится на 5 и 7: 2003//5\*7 = 57

Делится на 5 и 11: 2003//5\*11 = 36

Делится на 7 и 11: 2003//7\*11 = 26

Делится на 5, 7 и 11: 2003//5\*7\*11 = 5

1. К несчастью, судья на выставке цветов не разбирается в орхидеях. Если он выбирает победителей случайным образом среди 18 участниц, то сколько имеется способов вручить первый, второй и третий приз?

18 \* 17 \* 16 = 48 96

1. Разложить .



(a+b)10 = a10 + 10a9b + 45a8b2 + 120a7b3 + 210a6b4 + 252a5b5 + 210a4b6 + 120a3b7 + 45a2b8 + 10ab9 + ab10

1. Решить линейное однородное рекуррентное соотношение *2-го* порядка:  
   

an - 2an-1 + an-2 = 0

p2 – 2p + 1 = 0

p = 1

an = c + dn

a0 = c + d0 = -3

c = -3

a1 = c + d = 1

-3 + d = 1

d = 4

Ответ: -3 + 4n

***Группы.***

***Задание 1 .***

Для заданной в *таблице 1* группы:

1. Построить таблицу *Кэли*. Определить *нейтральный* элемент. Для каждого элемента группы указать *обратный* элемент.

def get\_expression(g1, g2):

return (g1 \* g2) % 63

def get\_cayley\_table(arr):

cayley\_table = dict()

for g1 in arr:

cayley\_table[g1] = dict()

for g2 in arr:

cayley\_table[g1][g2] = get\_expression(g1, g2)

return cayley\_table

def is\_neutral(e):

if len(e) == 6:

return True

else:

return False

def get\_neutral\_el(cayley\_table):

e = []

for g1 in cayley\_table:

e = []

for g2 in cayley\_table[g1]:

if g2 \* g1 == g2:

e.append(True)

if(is\_neutral(e)):

return g1

return False

def print\_reverse\_elements(cayley\_table, e):

for g1 in cayley\_table:

for g2 in cayley\_table[g1]:

if cayley\_table[g1][g2] == e:

print(str(g1) + ":" + str(g2))

array = [1,2,4,8,16,32]

cayley\_table = get\_cayley\_table(array)

print("Таблица Кэли:")

for g in array:

print(cayley\_table[g])

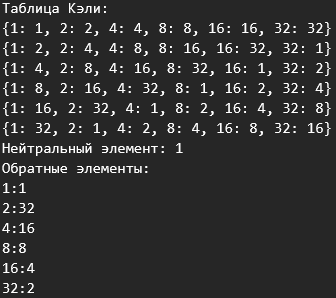
e = get\_neutral\_el(cayley\_table)

print("Нейтральный элемент: " + str(e))

print("Обратные элементы:")

print\_reverse\_elements(cayley\_table, e)

Результат выполнения:



1. Определить является ли группа *циклической*, указать *порядок* каждого элемента в группе.

Группа является циклической, т.к. 1 может породить все остальные элементы группы:

g1 = [1]

g1.append(get\_expression(g1[-1], 1))

g1.append(get\_expression(g1[-1], 1))

g1.append(get\_expression(g1[-1], 1))

g1.append(get\_expression(g1[-1], 1))

print(g1)



2, 22, 23, 24, 25

1. Определить является ли группа *абелевой*.

Да, так как для любых а и b справедливо, что a \*b = b \* a

1. Указать все *подгруппы* заданной группы и их порядок. Определить *порождающие элементы* для подгрупп. Для одной из подгрупп построить таблицу *Кэли*.

Код программы:

array = [1,2,4,8,16,32]

def subgroup(g):

if g == 1:

return "Нейтральный элемент 1"

subgroup = [g]

while True:

subgroup.append(get\_expression(subgroup[-1], g))

if subgroup[-1] == 1:

return subgroup

for g in array:

print(str(subgroup(g)) + " Порядок подгруппы: " + str(len(subgroup(g))))

Результат выполнения:

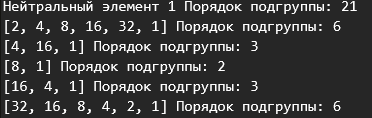
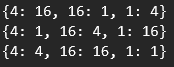


Таблица Кэли для одной из подгрупп:



1. Разложить группу на *смежные* классы по одной из подгрупп. Построить *фактор множество* для группы по данной подгруппе.
2. Построить таблицу *Кэли* для *фактор-группы* по одной из подгрупп.

def group\_classes(cayley\_table\_arr, arr):

index = len(arr) // 2

for g2 in range(index):

print(str(arr[g2]) + "H:" + str(cayley\_table\_arr[0][g2]) + ":" + str(cayley\_table\_arr[0][g2 + index]))

def get\_factor\_set(arr):

index = len(arr) // 2

factor\_set = []

for i in range(index):

element = (str(arr[i]) + str("H"))

factor\_set.append(element)

return factor\_set

def factor\_cayley\_table(arr):

factor\_cayley\_table = []

for g1 in range(len(arr)):

row = []

for g2 in range(len(arr)):

el = (str(get\_expression(arr[g1], arr[g2])) + "H")

row.append(el)

factor\_cayley\_table.append(row)

return factor\_cayley\_table

cayley\_table = get\_cayley\_table(array)

print("Классы для заданной группы:")

ct = get\_cayley\_table\_arr(array)

group\_classes(ct, array)

print("Фактор-множкство:")

fts = get\_factor\_set(array)

sg = subgroup(4)

print("Подгруппа:")

print(sg)

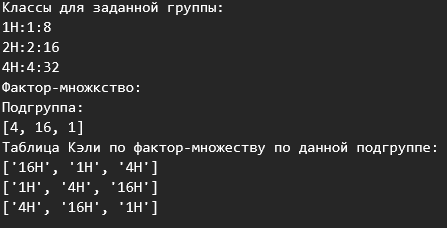
print("Таблица Кэли по фактор-множеству по данной подгруппе:")

fct = factor\_cayley\_table(sg)

for row in fct:

print(row)

Результат выполнения:



***Задание 2 . (вручную)***

1. Представить заданную подстановку произведением независимых циклов. Варианты заданий указаны в *таблице 2*. Подстановка из *6* элементов задана в *таблице 2* второй строкой подстановки. Например: подстановка записана в таблице как .
2. Определить порядок *p* заданной подстановки.
3. Составить таблицу *Кэли* для циклической группы, порожденной вашей постановкой. Группа будет иметь вид: . Для вашей подстановки найти обратную подстановку в полученной группе.
4. Определить четность подстановки:  
   – через число транспозиций в разложении подстановки в произведение транспозиций;  
   – через число инверсий в подстановке.
5. Определить *тип* подстановки.
6. Определить является ли ваша группа *абелевой*.
7. Варианты заданий указаны ниже, в *таблице 2*.

***Таблица 2.***

|  |  |
| --- | --- |
| *№* | *Подстановка* |
| *1.* |  |
| *2.* |  |
| *3.* |  |
| *4.* |  |

Вывод: в ходе выполнения данной лабораторной работы ознакомился с основными задачами комбианторики и теории групп.