Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №4

за 3 семестр

По дисциплине: «Дискретная математика»

Тема: «Комбинаторика. Группы»

Выполнил:

Студент 2 курса

Группы ПО-4(2)

Кречко К. А.

Проверила:

Глущенко Т.А.

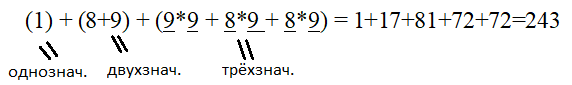
Брест 2020

**ЛАБОРАТОРНАЯ РАБОТА No 4**

**Комбинаторика**

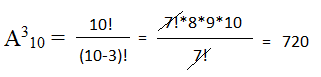
**Вариант – 2**

**1.** Сколько существует целых чисел между *0* и *1000*, содержащих ровно одну цифру 6?



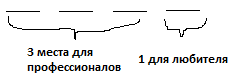
**62.** В скачках участвуют десять лошадей. Сколько существует вариантов призовой тройки лошадей?

Это размещение без повторений:

****

**3.** Предстоит выбрать команду четырех игроков в гольф из пяти профессиональных игроков и пяти любителей. Сколько разных команд может состоять из трех профессионалов и одного любителя?

Это сочетание без повторений: 

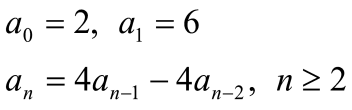


C53  \* C51 = 10 \* 5 = 50

**4.** Вычислить коэффициенты при произведении https://lh4.googleusercontent.com/NKIMHAcaNNWd9e8IBbgA0coHvuitLl8AC2vrL7ghKTa7llgeAys2PaePl1_ytKtUSS8c5drLC36HF12NPm-uJ6QC3FQ69u4jDgJ4QXfMa4Sp5gXtAT76b5sWiNxCYeuJoKLT6Mo в разложении https://lh3.googleusercontent.com/2btQKLWty_mokjKKH9n0PHkVa07TvAcZ6yoN6RBiOFP70YqCM4JF_FpoabawR0Ydu1ESXfeBBC4L70X04jto-bztl51g2j_hjIwCNkzYB5ftIbeIY3QyddoB2C94wVB59ItWUJs

Коэффициент будет равен:



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


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

Код:

def recurrent (a,n):

a.append (4\*a[n-1]-4\*a[n-2])

print(a)

if n==4:

print("Пятый член = ",a[4])

return

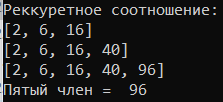
recurrent (a,n+1)

a=[2,6]

print("Реккуретное соотношение: ")

recurrent(a,2)

Результат:



**7.** Написать программу генерации перестановок (рекурсивный вариант). Четные номера – реализуют лексикографический порядок генерации.

Код:

**C++**

#include <algorithm>

#include <string>

#include <iostream>

using namespace std;

int main()

{

setlocale(LC\_ALL, "ru");

system("color f0");

string s;

cout << "Введите последовательность: ";

cin >> s;

sort(s.begin(), s.end());

do{

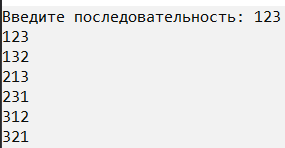
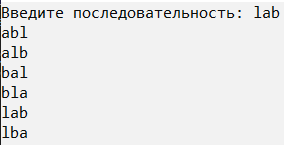
cout << s << endl;

} while (next\_permutation(s.begin(), s.end()));

return 0;

}

Результат:

**Python:**

def all\_perms(arr):

if len(arr) <=1:

yield arr

else:

for perm in all\_perms(arr[1:]):

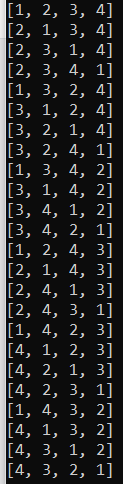
for i in range(len(arr)):

yield perm[:i] + arr[0:1] + perm[i:]

for el in all\_perms([1,2,3,4]):

print(el)

Результат:



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

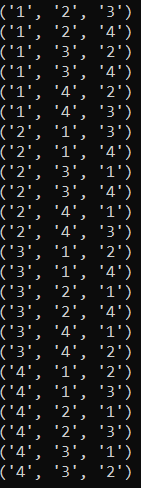
Код: Python

from itertools import \*

for i in permutations ('1234',3):

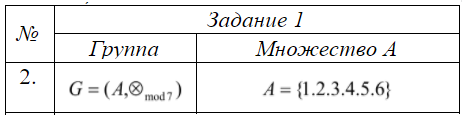
print(i, end='\n')

Результат:



**Группы**

**Задание 1**

****

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

Код:

#include <iostream>

using namespace std;

int main() {

setlocale(LC\_ALL, "ru");

system("color f0");

int const SIZE = 7;

int arr[SIZE][SIZE];

int n; //нейтральный элемент

for (int i = 0; i < SIZE; i++) { //заполнение вершин

arr[i][0] = i;

arr[0][i] = i;

}

cout << "Обратные элементы: " << endl;

for (int i = 1; i < SIZE; i++) {

for (int j = 1; j < SIZE; j++) {

arr[i][j] = (arr[i][0] \* arr[0][j]) % 7; //заполнение таблицы

if (arr[i][j] == arr[i][0] && arr[i][j] == arr[0][j]) { //проверка на нейтральный элемент

n = arr[i][j];

}

if (arr[i][j] == 1 && arr[i][0] != arr[0][j]) { //проверка на обратность ((стр\*стл)%7==нейтр элем)

cout << arr[i][0] << " - " << arr[0][j] << endl; // вывод пары обратных элементов

}

}

}

cout << endl << "Таблица Кэли:" << endl;

for (int i = 0; i < SIZE; i++) {

for (int j = 0; j < SIZE; j++) {

cout << arr[i][j] << " ";

}

cout << endl;

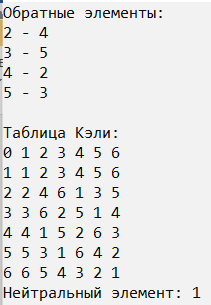
}

cout << "Нейтральный элемент: " << n << endl;

return 0;

}

Результат:



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

Группа является циклической, т.к. м.б. порождена одним элементом a в разных степенях.

**3.** Определить является ли группа абелевой.

Абелева группа — группа, в которой групповая операция является коммутативной.

Любая циклическая группа является абелевой.

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

Код:

cout << "Подгруппы: " << endl;

int num = 1, j = 0, i = 1, d = 1;

do {

int n = 1;

cout << "<" << arr[i][j] << "> = { ";

while ((arr[i][j] \* n) % 7 != 1) { //пока не дошли до нейтрального элемента

cout << (arr[i][j] \* n) % 7 << ", ";

n++;

num++;

}

cout << "0 } = " << num <<";"<< endl;

num = 1;

j++; d++;

} while (d < 8);

cout << endl;

int const x = 4;

int arr2[x][x];

for (int i = 0; i < 4; i++) { //заполнение вершин

arr2[i][0] = i\*2;

arr2[0][i] = i\*2;

}

for (int i = 1; i < x; i++) {

for (int j = 1; j < x; j++) {

arr2[i][j] = ((arr2[i][0] \* arr[0][j])\*2) % 7; //заполнение таблицы

}

}

cout << "Таблица Кэли для подгруппы <2>" << endl; //вывод

for (int i = 0; i < x; i++) {

for (int j = 0; j < x; j++) {

cout << arr2[i][j] << " ";

}

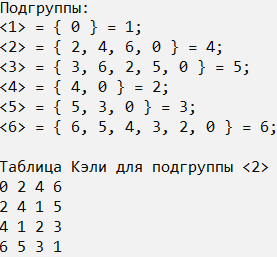
cout << endl;

}

return 0;

}

Результат:



**5.** Разложить группу на *смежные* классы по одной из подгрупп. Построить *фактор множество* для группы по данной подгруппе.

Левым смежным классом группы GG по множеству HH назовем множество вида aH={a⋅x|x∈H}⊆G

Фактор-множеством группы G по подгруппе Н называется множество всех левых, или что то же самое, правых смежных классов по подгруппе Н.

Код:

cout << " H= {" << arr[i][j] << ", " << arr[i][j + 3] << "}" << endl;

do {

cout << m << "H = {" << arr[i][j] + 1 << ", " << arr[i][j + 3] + 1 << "}" << endl;

r++;

num++;

m++;

j++;

} while (arr[i][j + 3] + 1 != 7);

m--;

cout << "Фактор множество: " << endl;

cout << "G/H = {H" << ", ";

do {

cout << (m - 3) \* (-1) << "H" << ", ";

m--;

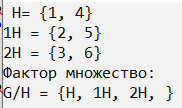
} while (m > 0);

cout << "}" << endl << endl;

return 0;

}

Результат:



**6.** Построить таблицу *Кэли* для *фактор-группы* по одной из подгрупп.

Код:

#include <iostream>

using namespace std;

int main() {

setlocale(LC\_ALL, "ru");

system("color f0");

cout << "Таблица Кэли для фактор-группы: " << endl;

int y[4][4];

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 4; j++) {

y[i][j] = (y[i][0] \* y[0][j]) % 7;

if (y[i][j] == 0) {

cout << "H ";

}

else {

cout << y[i][j] << "H ";

}

}

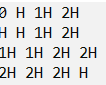
cout << endl;

}

return 0;

}

Результат:



***Задание 2 . (вручную)***

