Лабораторная работа №4

студента группы ПИ-212

Клюковского Артема Александровича

Цель: ознакомиться с темой размещений и создать программу, которая генерирует все возможные размещения из K элементов

Выполнение\_\_\_\_\_\_\_\_\_\_ Защита\_\_\_\_\_\_\_\_\_\_

Генератор размещений с повторениями

Блок-схема

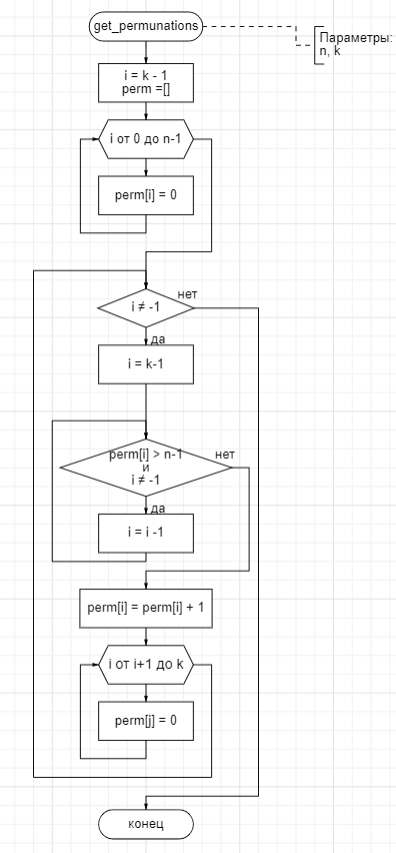


Рисунок - Блок-схема

Текст программы

def get\_permutation(n, k):

i = k - 1

perm = [0 for i in range(k)]

while i != -1:

print(perm)

i = k - 1

while perm[i] > n - 1 and i != -1:

i -= 1

perm[i] += 1

for j in range(i + 1, k):

perm[j] = 0

get\_permutation(5, 3)

Результат программы

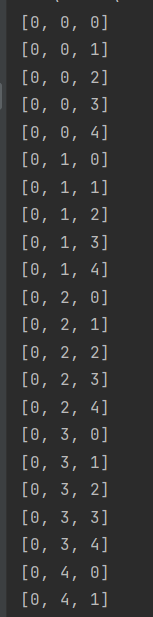


Рисунок - результат программы

Генератор сочетаний без повторений

Блок-схема

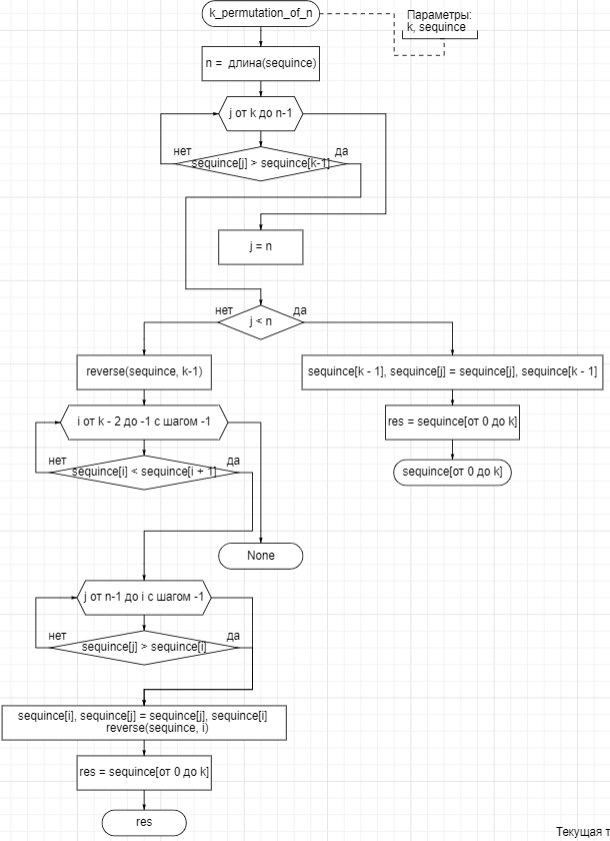


Рисунок - блок-схема k\_permutaions\_of\_n

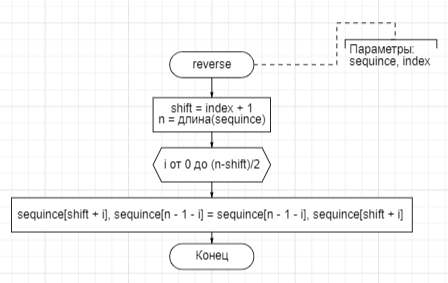


Рисунок - блок-схема reverse

Текст программы

sequince = [0, 1, 2, 3, 4]

def reverse(sequince, index):

shift = index + 1

n = len(sequince)

for i in range((n - shift) // 2):

sequince[shift + i], sequince[n - 1 - i] = sequince[n - 1 - i], sequince[shift + i]

def k\_permutaion\_of\_n(k, sequince):

n = len(sequince)

for j in range(k, n):

if sequince[j] > sequince[k - 1]:

break

else:

j = n

if j < n:

sequince[k - 1], sequince[j] = sequince[j], sequince[k - 1]

return sequince[:k:]

else:

reverse(sequince, k - 1)

for i in range(k - 2, -1, -1):

if sequince[i] < sequince[i + 1]:

break

else:

return None

for j in range(n - 1, i, -1):

if sequince[j] > sequince[i]:

break

sequince[i], sequince[j] = sequince[j], sequince[i]

reverse(sequince, i)

return sequince[:k:]

k = 5

perm = sequince[:k]

while perm is not None:

print(perm)

perm = k\_permutaion\_of\_n(k, sequince)

Результат программы

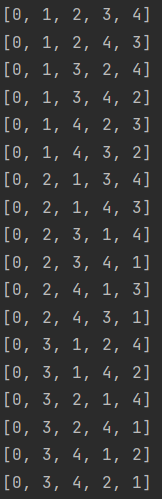


Рисунок - результат программы

Вывод:

Были приобретены навыки по созданию алгоритма для генератора размещений с повторениями и без.