|  |  |
| --- | --- |
| Изображение выглядит как текст, эмблема, герб, нашивка  Автоматически созданное описание | **Министерство образования и науки Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Робототехники и комплексной автоматизации

КАФЕДРА Системы автоматизированного проектирования (РК-6)

**ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

Дисциплина Методы комбинаторных вычислений

Название работы Лабораторная работа №3

Студент Долженко Анастасия Тимофеевна

Группа РК6-52Б

Вариант 7

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Долженко А.Т.**

*подпись, дата фамилия, и.о.*

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Волосатова Т.М.**

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Москва, 2024 г.*

**Оглавление**

[Задание 3](#_Toc178644053)

[Алгоритм работы программы 3](#_Toc178644054)

[Приложение 1. Код программы 5](#_Toc178644055)

[Приложение 2. Результаты работы программы 7](#_Toc178644056)

[Список литературы 7](#_Toc178644057)

# **Задание**

Вариант 7

Перечислить все натуральные числа из m 8-х разрядов, в которых цифры попарно различны и значения і младших разрядов не меньше j. При программной реализации перечислений значения параметров должны задавать опции командной строки с контролем допустимости их величин. Кроме того, должны быть предусмотрены значения параметров по умолчанию, если их опции не заданы, и обеспечен стандартный вывод результатов.

# **Алгоритм работы программы**

В общем случае систематическое перечисление размещений из **n** элементов по **m** элементов при произвольных целочисленных значениях **n>m>0** обеспечивает следующий алгоритм. Его итерации формируют размещения в лексиграфическом порядке, используя соответствующую технику генерации перестановок.

Для описания алгоритма, удобно рассматривать множество натуральных чисел от **1** до **n**, откуда на каждой итерации выбирают **m<n** различных чисел, с целью получить лексиграфически упорядоченную последовательность числовых векторов, которые образуют искомые размещения.

Итерационный процесс должен начинаться с лексиграфически наименьшего размещения, которое образует выборка первых **m** натуральных чисел указанного диапазона в порядке возрастания своих величин

**Amin = (1, 2, ... m).**

На каждой следующей итерации очередное размещение строится по предыдущему. Для этого нужно выполнить следующие **3** операции.

Сначала следует дополнить **(+)** текущее размещение до перестановки **n** чисел, приписав к нему справа, в порядке убывания своих величин все числа из образующего множества, которые отсутствуют в этом размещении.

Затем, по полученной перестановке требуется найти следующую в лексиграфическом порядке перестановку **(L)**. Для этого можно использовать соответствующий фрагмент алгоритма перечисления перестановок в лексиграфическом порядке.

Наконец, необходимо исключить **(-)** из полученной перестановки хвостовых чисел. Оставшиеся т чисел образуют очередное размещение, которое будет лексикографически больше любого из предыдущих.

Приведем пример, где по размещению **(4,1,5)** из множества натуральных чисел от **1** до **5** строится очередное размещение **(4, 2, 1)**:

**A39 = (4, 1, 5) + (3, 2) = (4, 1, 5, 3, 2) = L => (4, 2, 1, 3, 5) - (3, 5) = (4, 2, 1) = A40.**

Аналогичные итерации должны продолжаться до получения лексиграфически наибольшего размещения из **m** максимальных чисел диапазона от **1** до **n**, которые располагаются в порядке убывания своих величин:

**Amax = (n, n-1,… n-m+1).**

В программе реализуется данный алгоритм, дополненный проверками для выполнения заданных в задании условий, а также проверками вводимых данных.

# **Приложение 1. Код программы**

#include <iostream>

#include <vector>

#include <cstdlib>

#include <algorithm>

#define OSN 8

using namespace std;

void printPermutation(const vector<int> &perm) {

for(auto pos: perm)

if (pos < 10) cout << char(pos + '0');

else cout << char(pos + 'A' - 10);

cout << endl;

}

void getPermutation(vector<int> & perm, int m, int i, int j, int index) {

int s;

// ограничение на значений младших разрядов

if (index > m - i - 1) s = j;

else s = 0;

for(int ii = s; ii < OSN; ii++){

// проверка на попарное различие

if (index > 0){

auto pos = find(perm.begin(), perm.begin()+index, ii);

if ( pos != perm.begin()+index ) continue;

}

perm[index] = ii;

if (index < perm.size()-1) getPermutation(perm, m, i, j, index+1);

else printPermutation(perm);

}

}

int main(int argc, char \*argv[]) {

int m, i, j;

if (argc < 4){

m = 4;

i = 3;

j = 5;

}

else{

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

for(int y = 0; argv[x][y]; y++){

if ( !(argv[x][y] >= '0' && argv[x][y] <= '9') ){

cout << "Error. Input must integer" << endl;

return 0;

}

}

}

m = atoi(argv[1]);

if( !(m > 0 && m <= OSN) ){

cout << "Error, must be: (1st number) m > 0 and m <= " << OSN << endl;

return 0;

}

i = atoi(argv[2]);

if(!(i > 0 && i <= OSN)) {

cout << "Error, must be: (2nd number) i > 0 and i <= " << OSN << endl;

return 0;

}

j = atoi(argv[3]);

if( !(j > 0) ){

cout << "Error, must be: (3rd number) j > 0" << endl;

return 0;

}

if(!(i <= m)){

cout << "Error, must be: (1st number) i <= (2nd number) m" << endl;

return 0;

}

if (!(OSN-j >= i)){

cout << "Error, must be: " << OSN << " - (2nd number) j >= (1st number) i" << endl;

return 0;

}

}

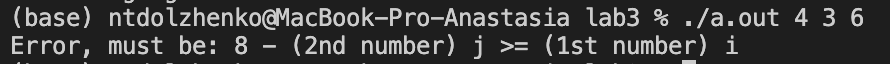
vector<int> perm(m);

getPermutation(perm, m, i, j, 0);

return 0;

}

# **Приложение 2. Результаты работы программы**

1. 
2. Изображение выглядит как текст, снимок экрана, Шрифт, черный

   Автоматически созданное описание

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

# **Список литературы**

1. Методы комбинаторных вычислений – Т.М. Волосатова, С.В. Родионов, 2011
2. http://bigor.bmstu.ru/?cnt/?doc=Comby/base.cou