**Лабораторная работа 2. Комбинаторные алгоритмы решения оптимизационных задач**

**ЦЕЛЬ РАБОТЫ:** приобрести навыки разработки генераторов подмножеств, перестановок, сочетаний и размещений на С++; научиться применять разработанные генераторы для решения задач о рюкзаке (упрощенную), коммивояжера, об оптимальной загрузке судна и об оптимальной загрузке судна с центровкой.

**ЗАДАНИЕ ДЛЯ ВЫПОЛНЕНИЯ:**

**Задание 1.** Разобрать и разработать генератор подмножеств заданного множества.

// Main

#include <iostream>

#include "Combi.h"

int main() // с помощью инкременирования битовой маски

{

setlocale(LC\_ALL, "rus");

char AA[][2] = { "A", "B", "C", "D" };

std::cout << std::endl << " - Генератор множества всех подмножеств -";

std::cout << std::endl << "Исходное множество: ";

std::cout << "{ ";

for (int i = 0; i < sizeof(AA) / 2; i++)

std::cout << AA[i] << ((i < sizeof(AA) / 2 - 1) ? ", " : " ");

std::cout << "}";

std::cout << std::endl << "Генерация всех подмножеств ";

combi::subset s1(sizeof(AA) / 2); // создание генератора

int n = s1.getfirst(); // первое (пустое) подмножество

while (n >= 0) // пока есть подмножества

{

std::cout << std::endl << "{ ";

for (int i = 0; i < n; i++)

std::cout << AA[s1.ntx(i)] << ((i < n - 1) ? ", " : " ");

std::cout << "}";

n = s1.getnext(); // cледующее подмножество

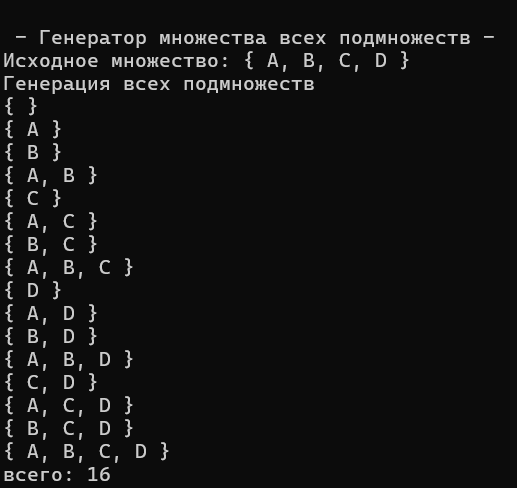
};

std::cout << std::endl << "всего: " << s1.count() << std::endl;

system("pause");

return 0;

}



**Задание 2.** Разобрать и разработать генератор сочетаний.

// Main

#include <iostream>

#include "Combi.h"

int main() // битовая маска, но берем только подмножества с N двоичными единицами

{

setlocale(LC\_ALL, "rus");

char AA[][2] = { "A", "B", "C", "D", "E" };

std::cout << std::endl << " --- Генератор сочетаний ---";

std::cout << std::endl << "Исходное множество: ";

std::cout << "{ ";

for (int i = 0; i < sizeof(AA) / 2; i++)

std::cout << AA[i] << ((i < sizeof(AA) / 2 - 1) ? ", " : " ");

std::cout << "}";

std::cout << std::endl << "Генерация сочетаний ";

combi::xcombination xc(sizeof(AA) / 2, 3);

std::cout << "из " << xc.n << " по " << xc.m;

int n = xc.getfirst();

while (n >= 0)

{

std::cout << std::endl << xc.nc << ": { ";

for (int i = 0; i < n; i++)

std::cout << AA[xc.ntx(i)] << ((i < n - 1) ? ", " : " ");

std::cout << "}";

n = xc.getnext();

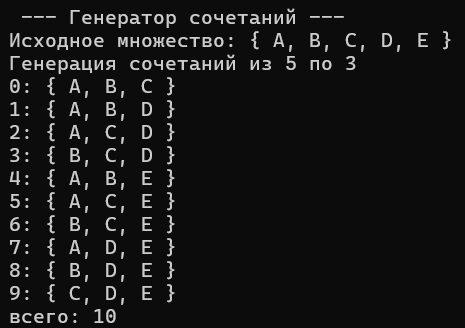
};

std::cout << std::endl << "всего: " << xc.count() << std::endl;

system("pause");

return 0;

}



**Задание 3.** Разобрать и разработать генератор перестановок.

// --- Main

#include <iostream>

#include "Combi.h"

#include <iomanip>

int main() // алгоритм Джонсона – Троттера

{

setlocale(LC\_ALL, "rus");

char AA[][2] = { "A", "B", "C", "D" };

std::cout << std::endl << " --- Генератор перестановок ---";

std::cout << std::endl << "Исходное множество: ";

std::cout << "{ ";

for (int i = 0; i < sizeof(AA) / 2; i++)

std::cout << AA[i] << ((i < sizeof(AA) / 2 - 1) ? ", " : " ");

std::cout << "}";

std::cout << std::endl << "Генерация перестановок ";

combi::permutation p(sizeof(AA) / 2);

\_\_int64 n = p.getfirst();

while (n >= 0)

{

std::cout << std::endl << std::setw(4) << p.np << ": { ";

for (int i = 0; i < p.n; i++)

std::cout << AA[p.ntx(i)] << ((i < p.n - 1) ? ", " : " ");

std::cout << "}";

n = p.getnext();

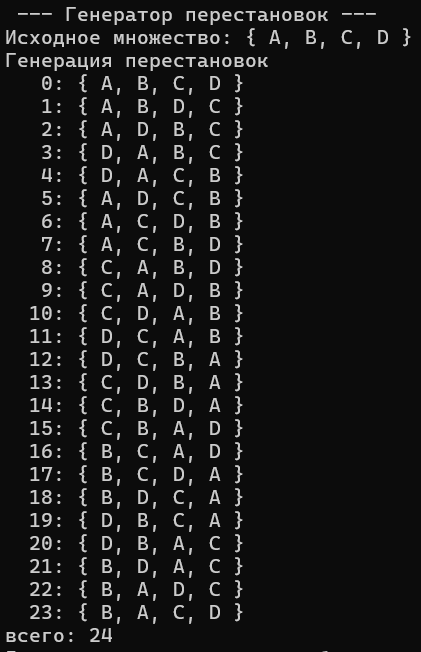
};

std::cout << std::endl << "всего: " << p.count() << std::endl;

system("pause");

return 0;

}



**Задание 4.** Разобрать и разработать генератор размещений.

// --- main

#include <iostream>

#include <iomanip>

#include "Combi.h"

#define N (sizeof(AA)/2)

#define M 3

int main()

{

setlocale(LC\_ALL, "rus");

char AA[][2] = { "A", "B", "C", "D" };

std::cout << std::endl << " --- Генератор размещений ---";

std::cout << std::endl << "Исходное множество: ";

std::cout << "{ ";

for (int i = 0; i < N; i++)

std::cout << AA[i] << ((i < N - 1) ? ", " : " ");

std::cout << "}";

std::cout << std::endl << "Генерация размещений из " << N << " по " << M;

combi::accomodation s(N, M);

int n = s.getfirst();

while (n >= 0)

{

std::cout << std::endl << std::setw(2) << s.na << ": { ";

for (int i = 0; i < M; i++)

std::cout << AA[s.ntx(i)] << ((i < n - 1) ? ", " : " ");

std::cout << "}";

n = s.getnext();

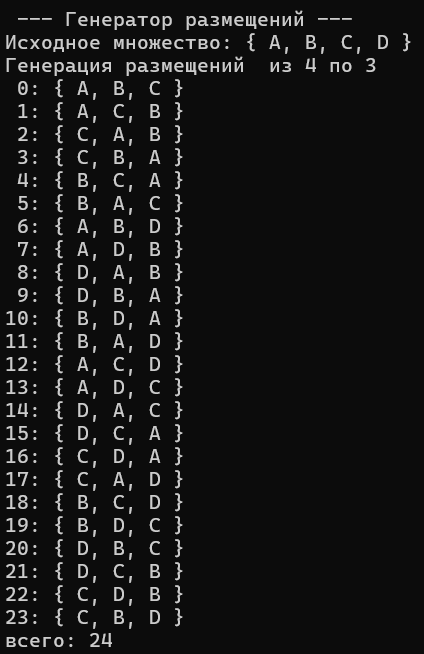
};

std::cout << std::endl << "всего: " << s.count() << std::endl;

system("pause");

return 0;

}



**Задание 5.**  Решить в соответствии с вариантом задачу и результат занести в отчет (Вариант распределяется по списку):

2, 6, 10, 14) упрощенную о рюкзаке (веса предметов и их стоимость сгенерировать случайным образом: вместимость рюкзака 300 кг, веса предметов 10 – 300 кг, стоимость предметов 5 – 55 у.е.; количество предметов – 18 шт.);

#include <iostream>

#include <cstdlib>

#include <ctime>

#include "Combi.h"

#include <tchar.h>

#include "Knapsack.h"

#define NN 18 // Количество предметов

int \_tmain(int argc, \_TCHAR\* argv[]) {

setlocale(LC\_ALL, "rus");

srand(static\_cast<unsigned int>(time(0))); // Инициализация генератора случайных чисел

int V = 300; // Вместимость рюкзака

int v[NN]; // Массив весов предметов

int c[NN]; // Массив стоимостей предметов

// Генерация случайных весов и стоимостей

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

v[i] = rand() % (291) + 10; // Вес от 10 до 300 кг

c[i] = rand() % (51) + 5; // Стоимость от 5 до 55 у.е.

}

short m[NN] = { 0 }; // Количество предметов каждого типа, инициализируем нулями

int maxcc = knapsack\_s(

V, // [in] вместимость рюкзака

NN, // [in] количество типов предметов

v, // [in] размер предмета каждого типа

c, // [in] стоимость предмета каждого типа

m // [out] количество предметов каждого типа

);

std::cout << std::endl << "-------- Задача о рюкзаке --------- ";

std::cout << std::endl << "- количество предметов : " << NN;

std::cout << std::endl << "- вместимость рюкзака : " << V;

std::cout << std::endl << "- размеры предметов : ";

for (int i = 0; i < NN; i++) std::cout << v[i] << " ";

std::cout << std::endl << "- стоимости предметов : ";

for (int i = 0; i < NN; i++) std::cout << c[i] << " ";

std::cout << std::endl << "- оптимальная стоимость рюкзака: " << maxcc;

std::cout << std::endl << "- вес рюкзака: ";

int s = 0;

for (int i = 0; i < NN; i++) s += m[i] \* v[i];

std::cout << s;

std::cout << std::endl << "- выбраны предметы: ";

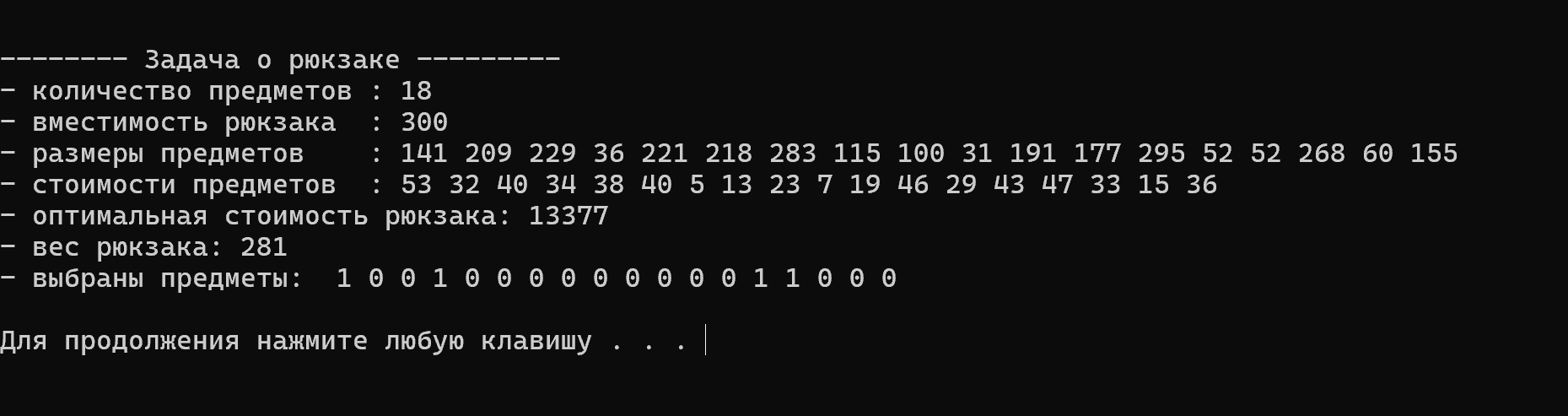
for (int i = 0; i < NN; i++) std::cout << " " << m[i];

std::cout << std::endl << std::endl;

system("pause");

return 0;

}



**Задание 6.** Исследовать зависимость времени вычисления необходимое для решения задачи (в соответствии с вариантом) от размерности задачи и результат в виде графика с небольшим пояснением занести в отчет:

2, 6, 10, 14) упрощенную о рюкзаке (количество предметов 12 – 20 шт.);

#include <iostream>

#include "..//Boat(Task5)/Combi.h"

#include "..//Boat(Task5)/Combi.cpp"

#include "..//Boat(Task5)/Knapsack.h"

#include <time.h>

#include <iomanip>

#include <tchar.h>

#define NN (sizeof(c)/sizeof(int))

int \_tmain(int argc, \_TCHAR\* argv[])

{

setlocale(LC\_ALL, "rus");

int V = 600, // вместимость рюкзака

v[] = { 25, 56, 67, 40, 20, 27, 37, 33, 33, 44, 53, 12,

60, 75, 12, 55, 54, 42, 43 ,44},

c[] = { 15, 26, 27, 43, 16, 26, 42, 22, 34, 12, 33, 30,

12, 45, 60, 41, 33, 11, 14, 16 };

short m[NN];

int maxcc = 0;

clock\_t t1, t2;

std::cout << std::endl << "-------- Задача о рюкзаке --------- ";

std::cout << std::endl << "- вместимость рюкзака : " << V;

std::cout << std::endl << "-- количество ------ продолжительность -- ";

std::cout << std::endl << " предметов вычисления ";

for (int i = 14; i <= NN; i++)

{

t1 = clock();

maxcc = knapsack\_s(V, i, v, c, m);

t2 = clock();

std::cout << std::endl << " " << std::setw(2) << i

<< " " << std::setw(5) << (t2 - t1);

}

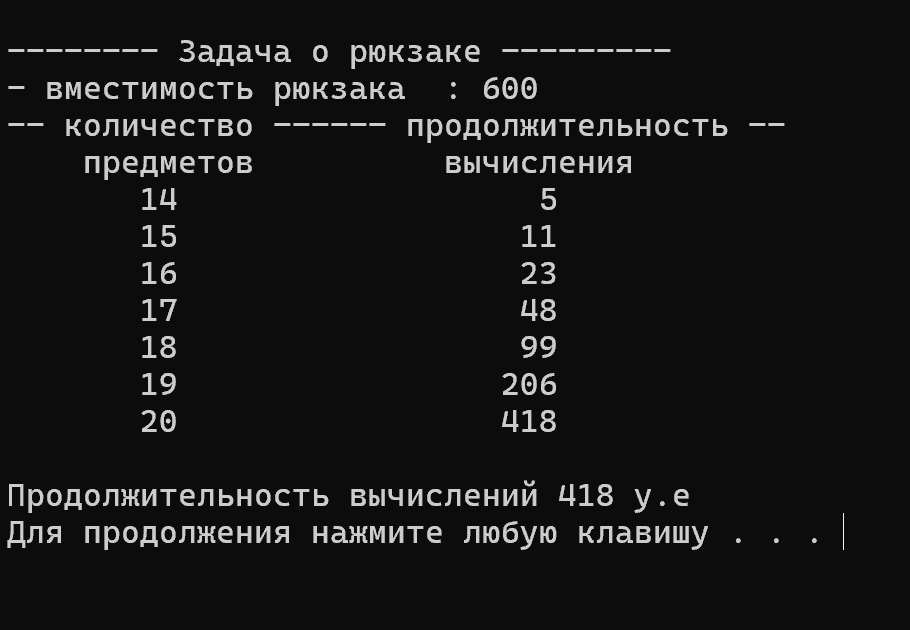
std::cout << std::endl << std::endl;

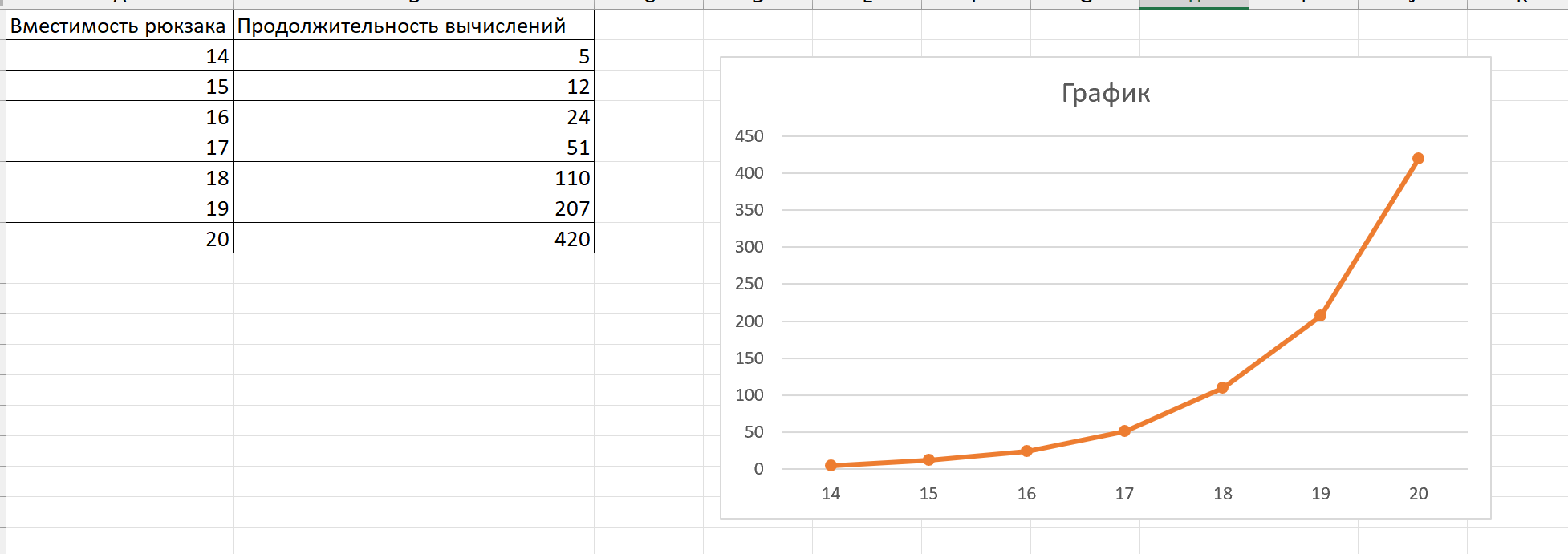
std::cout << "Продолжительность вычислений " << t2-t1 << " y.e" << std::endl;

system("pause");

return 0;

}





**Вопросы для защиты лабораторной работы**

* + - 1. *Что такое комбинаторика?*

Комбинаторика — это раздел математики, изучающий вопросы выбора, расположения и распределения элементов множества в соответствии с заданными правилами.

* + - 1. *Что такое генератор?*

Генератор — это алгоритм или структура данных, позволяющая последовательно генерировать элементы множества, таких как размещения, сочетания, перестановки или подмножества​.

* + - 1. *Чем размещения отличаются от сочетаний?*

Размещения отличаются от сочетаний тем, что в размещениях порядок элементов имеет значение, а в сочетаниях — нет. В размещениях из n элементов по 𝑚 учитывается порядок их следования, а в сочетаниях — только факт их выбора​.

* + - 1. *Каких подмножеств множества любой размерности всегда по одному?*

Всегда существуют два подмножества: пустое подмножество и само множество в полном составе​.

* + - 1. *С помощью какого генератора решается задача о загрузке судна?*

С помощью генератора размещений и генератора сочетаний.

* + - 1. *С помощью какого генератора решается задача о рюкзаке?*

С помощью генератора множества всех подмножеств.

* + - 1. *С помощью какого генератора решается задача о коммивояжере?*

С помощью генератора перестановок.