**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.

#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

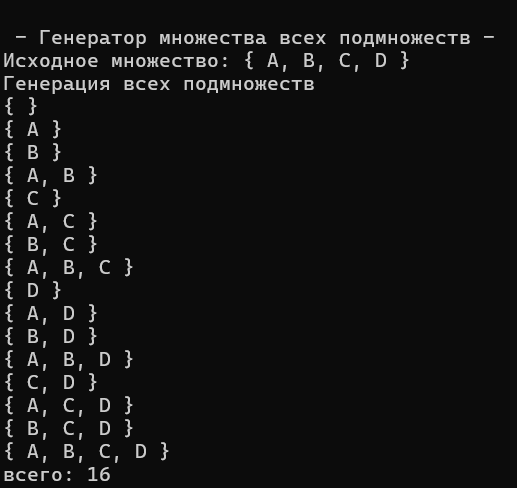
    };

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

    system("pause");

    return 0;

}



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

#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();

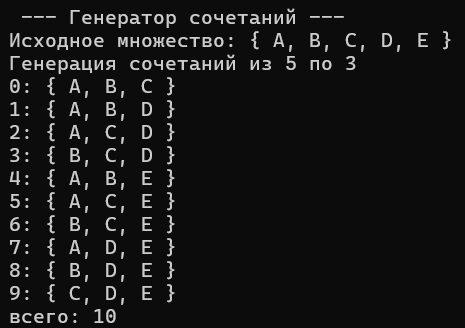
    };

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

    system("pause");

    return 0;

}



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

#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();

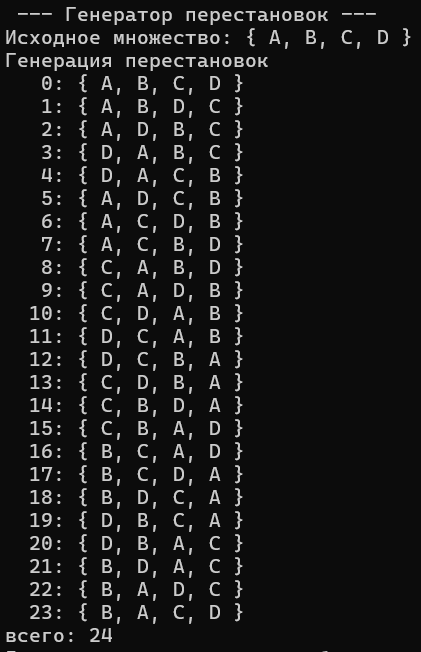
    };

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

    system("pause");

    return 0;

}



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

#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();

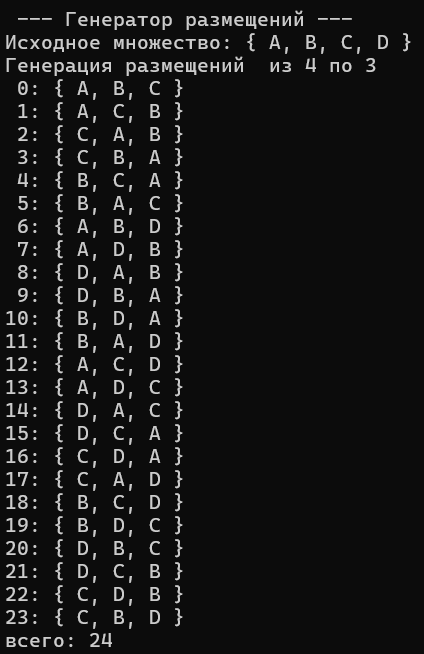
    };

    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):

3, 7, 11, 15) об оптимальной загрузке судна (веса контейнеров сгенерировать случайным образом: ограничение по общему весу – 1500 кг., количество мест на судне для контейнеров – 5, количество контейнеров 25, веса контейнеров 100 – 900 кг., доход от перевозки 10 – 150 у.е.);



#include <iostream>

#include <iomanip>

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

#include <time.h>

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

#define MM 6

#define SPACE(n) std::setw(n)<<" "

int wmain()

{

    setlocale(LC\_ALL, "rus");

    int V = 1500,

        v[] = { 250, 560, 670, 400, 200, 270, 370, 330, 330, 440, 530, 120,

               200, 270, 370, 330, 330, 440, 700, 120, 550, 540, 420, 170,

               600, 700, 120, 550, 540, 420, 430, 140, 300, 370, 310 };

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

               42, 22,  34,  43,  16,  26,  14,  12,  25,  41,  17,  28,

               12, 45,  60,  41,  33,  11,  14,  12,  25,  41,  30 };

    short r[MM];

    int maxcc = 0;

    clock\_t t1, t2;

    std::cout << std::endl << "-- Задача об оптимальной загрузке судна -- ";

    std::cout << std::endl << "-  ограничение по весу    : " << V;

    std::cout << std::endl << "-  количество мест        : " << MM;

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

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

    for (int i = 25; i <= NN; i++)

    {

        t1 = clock();

        int maxcc = boat(V, MM, i, v, c, r);

        t2 = clock();

        std::cout << std::endl << SPACE(7) << std::setw(2) << i

            << SPACE(15) << std::setw(5) << (t2 - t1);

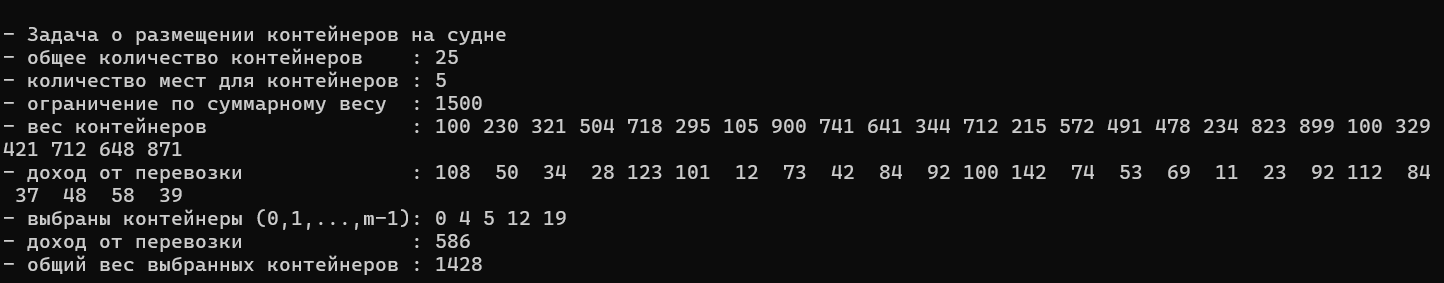
    }

    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:

3, 7, 11, 15) об оптимальной загрузке судна (количество мест на судне для контейнеров – 6, количество контейнеров 25 – 35

#include <iostream>

#include <iomanip>

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

#include <time.h>

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

#define MM 6

#define SPACE(n) std::setw(n)<<" "

int wmain()

{

    setlocale(LC\_ALL, "rus");

    int V = 1500,

        v[] = { 250, 560, 670, 400, 200, 270, 370, 330, 330, 440, 530, 120,

               200, 270, 370, 330, 330, 440, 700, 120, 550, 540, 420, 170,

               600, 700, 120, 550, 540, 420, 430, 140, 300, 370, 310 };

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

               42, 22,  34,  43,  16,  26,  14,  12,  25,  41,  17,  28,

               12, 45,  60,  41,  33,  11,  14,  12,  25,  41,  30 };

    short r[MM];

    int maxcc = 0;

    clock\_t t1, t2;

    std::cout << std::endl << "-- Задача об оптимальной загрузке судна -- ";

    std::cout << std::endl << "-  ограничение по весу    : " << V;

    std::cout << std::endl << "-  количество мест        : " << MM;

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

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

    for (int i = 25; i <= NN; i++)

    {

        t1 = clock();

        int maxcc = boat(V, MM, i, v, c, r);

        t2 = clock();

        std::cout << std::endl << SPACE(7) << std::setw(2) << i

            << SPACE(15) << std::setw(5) << (t2 - t1);

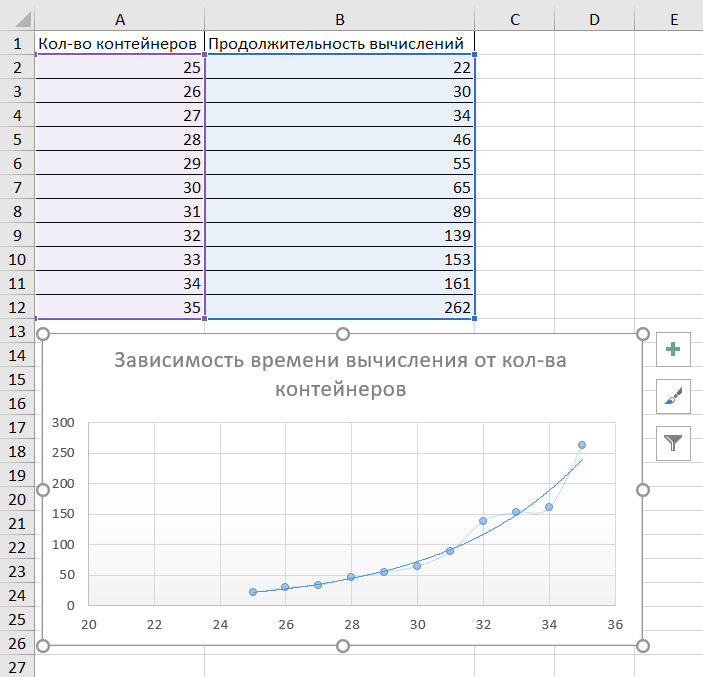
    }

    std::cout << std::endl << 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.