**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.** Разработать генератор подмножеств заданного множества.

**Задание 2.** Разработать генератор сочетаний

**Задание 3.** Разработать генератор перестановок

**Задание 4.** Разработать генератор размещений

**Задание 5.**  Решить в соответствии с вариантом задачу:

(Вариант 1, 5, 9, 13) коммивояжера (расстояния сгенерировать случайным образом: 10 городов, расстояния 10 – 300 км, 3 расстояния между городами задать бесконечными);

(Вариант 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.);

(Вариант 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.);

(Вариант 4, 8, 12, 16) об оптимальной загрузке судна с условием центровки (веса контейнеров сгенерировать случайным образом: количество мест на судне для контейнеров – 5, количество контейнеров 8, веса контейнеров 100 – 200 кг., доход от перевозки 10 – 100 у.е.; минимальный вес контейнера для каждого места 50 – 120 кг, максимальный вес контейнера для каждого места 150 – 850 кг);

**Задание 6.** Исследовать зависимость времени вычисления необходимое для решения задачи (в соответствии с вариантом) от размерности задачи:

(Вариант 1, 5, 9, 13) коммивояжера (5–12 городов);

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

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

(Вариант 4, 8, 12, 16) об оптимальной загрузке судна с условием центровки (количество мест на судне для контейнеров 4 – 8);

**ТЕОРЕТИЧЕСКОЕ ВВЕДЕНИЕ:**

1. **Генерация подмножеств заданного множества**



Рис. 1. Генерация множества всех подмножеств



Рис. 2. Шаблон структуры генератора множества всех подмножеств



Рис. 3. Реализация методов структуры **subset**



Рис. 4. Пример применения генератора множества всех подмножеств

**Решение упрощенной задачи о рюкзаке с помощью генератора множества всех подмножеств**

На рис. 6 изображена схема решения задачи о рюкзаке с применением генератора множества всех подмножеств. Задача имеет следующие исходные данные:

 – вместимость (объем) рюкзака;

 – количество предметов;

 – вектор объемов предметов;

 – вектор стоимостей предметов.



Рис. 6. Схема решения задачи о рюкзаке с применением генератора множества всех подмножеств

На рис. 7 и 8 представлен пример реализации функции **knapsack\_s** на языке C++, которая решает задачу о рюкзаке.



Рис.7. Прототип функции **knapsack\_s**



Рис.8. Реализация функции **knapsack\_s**

З

На рис. 9 приведен пример вызова функции **knapsack\_s** для решения задачи о рюкзаке с исходными данными для схемы, представленной на рис. 6.



Рис. 9. Пример использования функции **knapsack\_s**

Оценить зависимость продолжительности вычисления оптимальной комбинации предметов от их общего количества можно с помощью программы, изображенной на рис. 11.



Рис. 11. Вычисление продолжительности решения задачи о рюкзаке при различном количестве предметов

1. **Генерация сочетаний**

На рис. 14 представлена схема построения множества сочетаний  из элементов множества  Закрашенным прямоугольником на рисунке обозначены номера (индексы) элементов битовых последовательностей   и элементов множества  Стрелки связывают битовые последовательности, содержащие три двоичные единицы и сгенерированные сочетания множества  Для каждой стрелки указаны индексы единичных позиций соответствующих битовых последовательностей. Эти индексы используются для выбора элементов из множества для включения в соответствующее сочетание. Очевидно, что такой алгоритм генерации сочетаний имеет сложность  как и алгоритм генерации множества всех подмножеств.



Рис.14. Схема генерации сочетаний на основе множества всех подмножеств

На рис. 15 и 16 представлена реализация генератора сочетаний на языке С++. Генератор реализован в виде структуры **xcombination**.



Рис. 15. Шаблон структуры генератора сочетаний



Рис. 16. Реализация функций генератора сочетаний



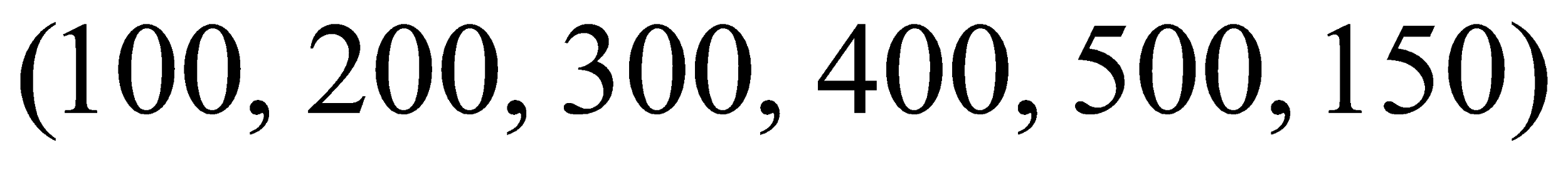
**Решение задачи об оптимальной загрузке судна на основе генератора сочетаний**

На рис. 19 изображена схема решения задачи с применением генератора подмножеств. Задача имеет следующие исходные данные:

 – ограничение по общему весу контейнеров;

 – количество контейнеров;

 – количество свободных мест на палубе;

 – вес контейнеров;

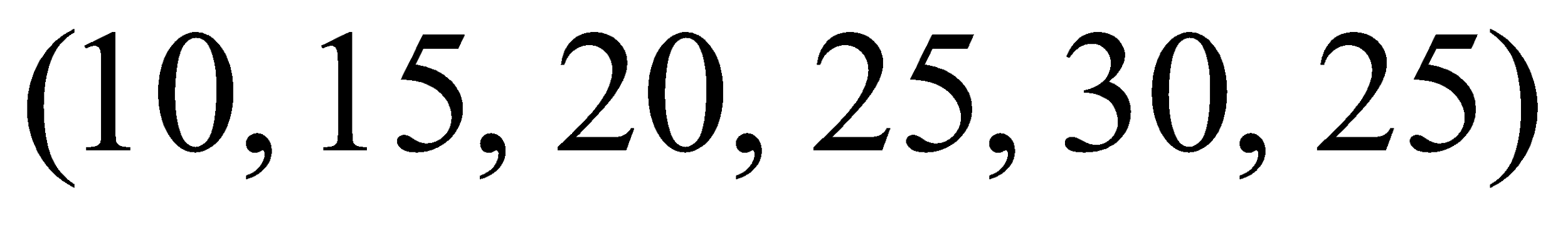
 – доход от перевозки контейнеров.

 Рис. 19. Схема решения задачи об оптимальной загрузке судна

На рис. 20 и 21 представлен пример реализации на языке С++ функции **boat**,решающей задачу об оптимальной загрузке судна.



Рис. 20. Функция **boat**, решающая задачу об оптимальной загрузке судна



Рис. 21. Реализация функции **boat**



Рис. 22. Пример решения задачи об оптимальной загрузке судна

На рис. 24 представлена программа, с помощью которой можно оценить продолжительность решения задачи о загрузке судна при разном количестве контейнеров. В программе фиксируется значение параметра **m** (количество мест для контейнеров) и вычисляется продолжительность работы функции boat в зависимости от параметра n (общее количество контейнеров).



Рис. 24. Вычисление продолжительности решения задачи о загрузке судна при разном количестве контейнеров

**3. Генерация перестановок**

Схема алгоритма генерации множества всех перестановок множества  приведена на рис. 1.



Рис. 4.1. Схема работы алгоритма Джонсона – Троттера

**Реализация генератора перестановок на языке C++**

На рис. 2 и 3 представлена программная реализация генератора перестановок.



Рис. 2. Шаблон структуры генератора перестановок

Рис. 3. Реализация функций генератора перестановок



Рис. 4. Пример применения генератора перестановок

На рис. 6 изображена схема решения задачи коммивояжера с применением генератора перестановок. Задача решается для пяти городов.



Рис. 6. Схема решения задачи коммивояжера

Расстояние между городами задается следующей матрицей 



На рис. 7 и 8 представлен пример реализации на C++ функции **salesman**, вычисляющей оптимальный кольцевой маршрут коммивояжера.



Рис. 7. Функция **salesman**, решающая задачу коммивояжера



Рис. 8. Реализация функции **salesman**

На рис. 9 и 10 приведен пример вызова функции **salesman** для решения задачи с исходными данными к схеме на рис. 6.



Рис. 9. Пример решения задачи коммивояжера

На рис. 11 представлена программа, позволяющая оценить продолжительность решения задачи коммивояжера в зависимости от количества городов.



Рис. 11. Вычисление продолжительности решения задачи коммивояжера при разном количестве городов

**4. Генерация размещений**

На рис. 1 представлена схема построения множества размещений  из элементов множества 



Рис.1. Схема генерации размещений

**Реализация генератора размещений на языке С++**



Рис.2. Шаблон структуры генератора размещений



Рис. 3. Реализация функций генератора размещений



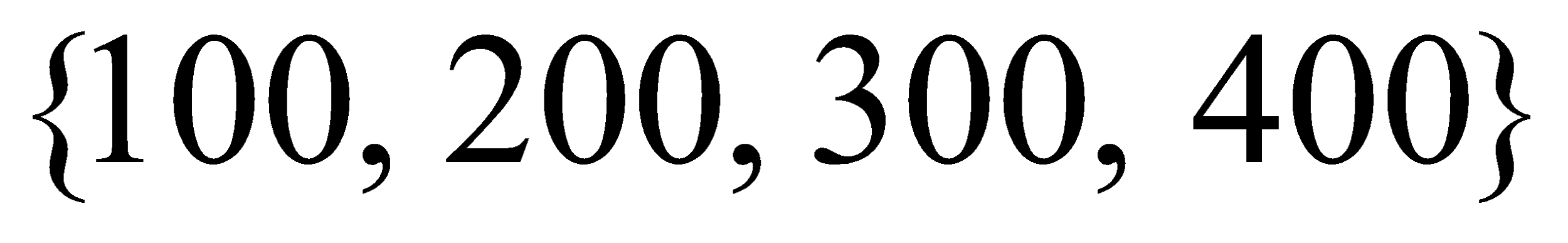
Рис.4. Пример использования генератора перестановок

**Решение задачи об оптимальном размещении контейнеров на судне с помощью генератора размещений**

На рис. 6 изображена схема, поясняющая решение этой задачи с помощью генератора размещений. Задача имеет следующие исходные данные:

 – общее количество контейнеров;

 – количество свободных мест на палубе судна;

– вес контейнеров 

 – доход от перевозки контейнеров 

 – минимальный вес контейнеров (

 – максимальный вес контейнеров 

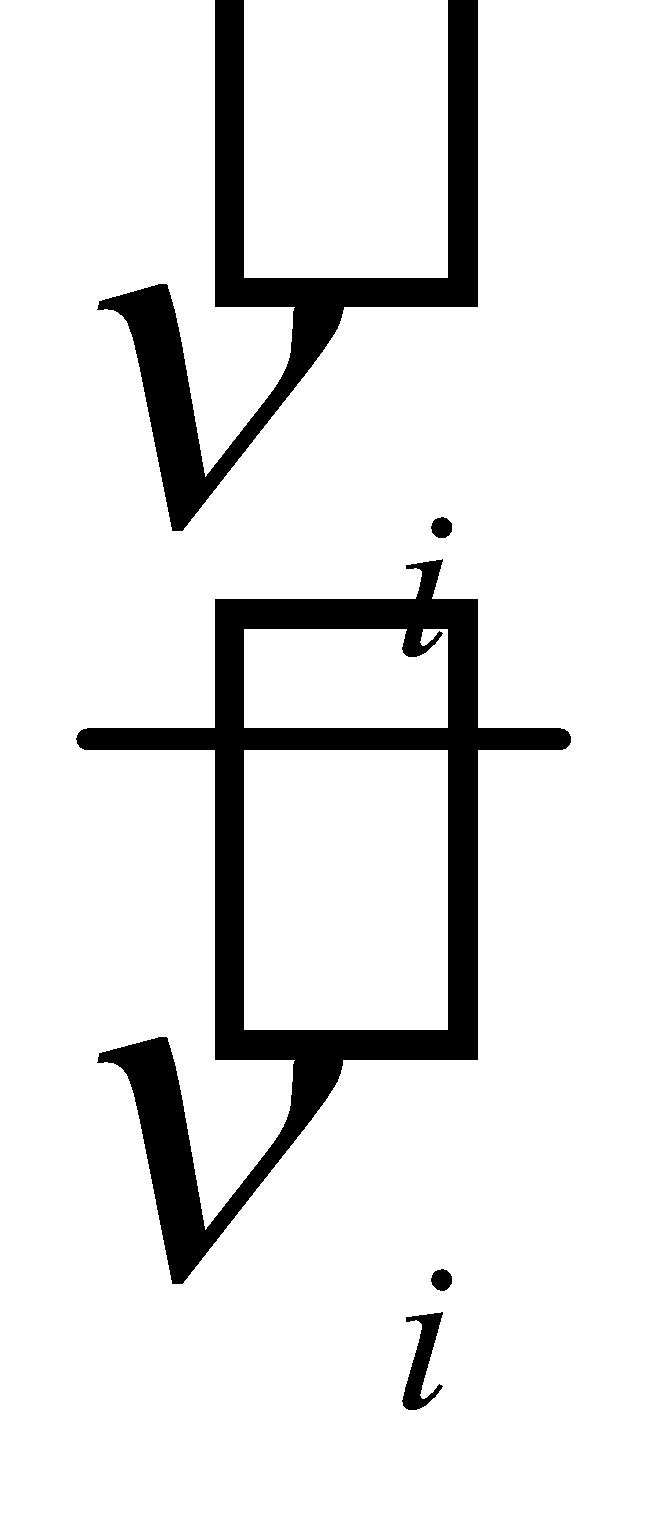


Рис. 6. Схема решения задачи об оптимальном размещении контейнеров на судне



Рис. 7. Функция **boat\_c**, решающая задачу об оптимальном размещении контейнеров на судне



Рис. 8. Реализация функции **boat\_c**



Рис. 9. Пример решения задачи об оптимальном размещении контейнеров на судне

На рис. 11 представлена программа, позволяющая оценить продолжительность решения задачи о размещении контейнеров в зависимости от количества свободных мест на палубе судна



Рис. 11. Оценка продолжительности решения задачи о размещении контейнеров на судне