Учреждение образования

«Белорусский государственный технологический университет»

**«Математическое программирование»**

**Лабораторная работа №2**

**Комбинаторные алгоритмы решения оптимизационных задач**

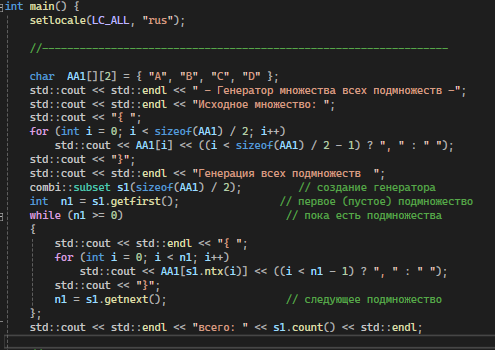
Выполнила:

Студентка 2 курса 5 группы ФИТ

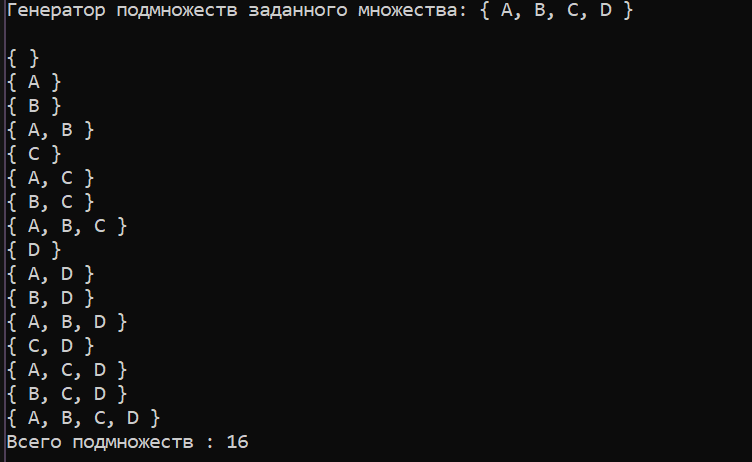
Городилина Анастасия Сергеевна

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

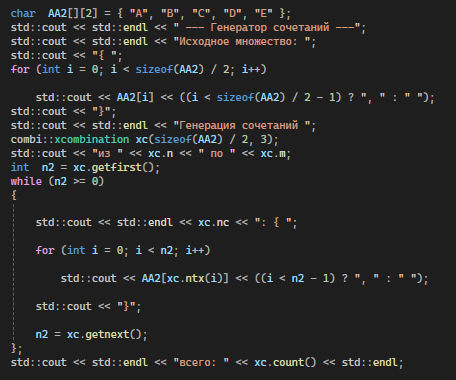


Код расчёта подмножеств исходного множества

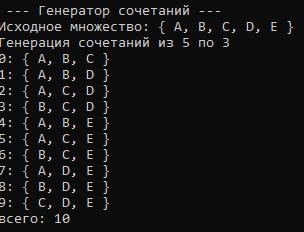


Результат выполнения программы

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

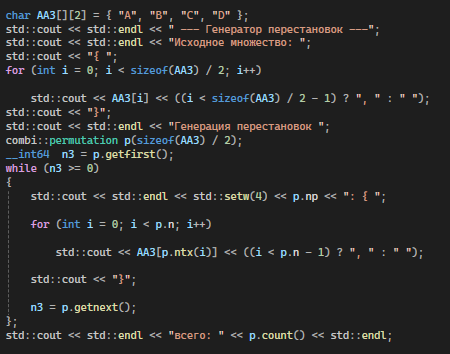


Код генерации сочетаний из исходного подмножества

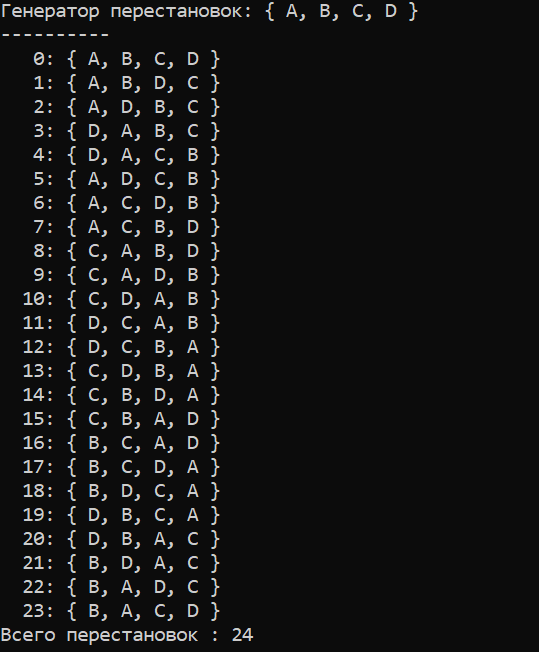


Результат генерации сочетаний из исходного подмножества

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

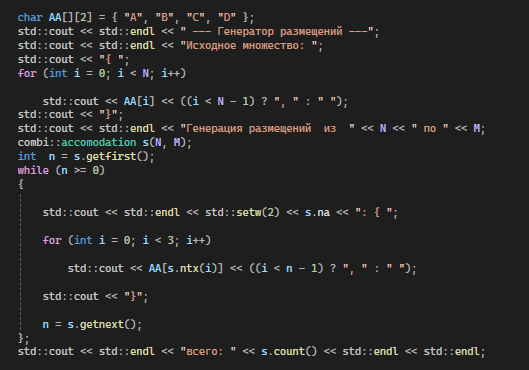


Код генерации перестановок исходного множества

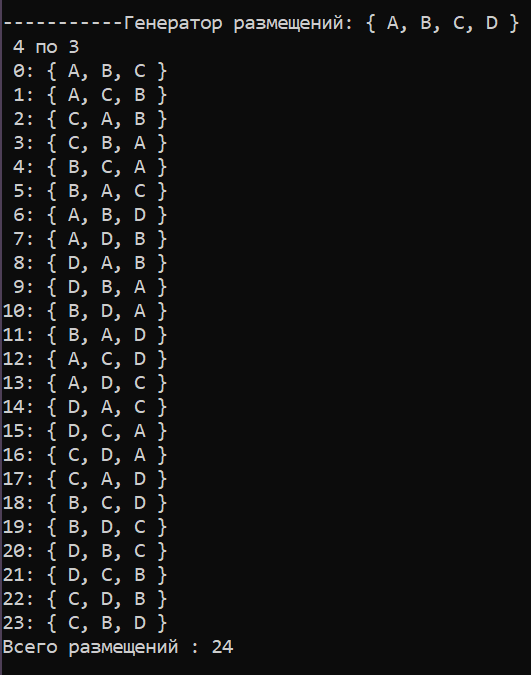


Результат вычисления перестановок исходного множества

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



Код генерации размещений из исходного множества

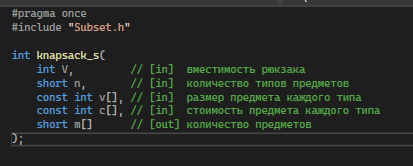


Результат генерации размещений исходного множества

**Задание 5**. 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 "Knapsack.h"  #define NINF 0x80000000  int calcv(combi::subset s, const int v[]) // объем в рюкзаке  {  int rc = 0;  for (int i = 0; i < s.sn; i++) rc += v[s.ntx(i)];  return rc;  };  int calcc(combi::subset s, const int v[], const int c[]) //стоимость в рюкзаке  {  int rc = 0;  for (int i = 0; i < s.sn; i++) rc += (v[s.ntx(i)] \* c[s.ntx(i)]);  return rc;  };  void setm(combi::subset s, short m[]) //отметить выбранные предметы  {  for (int i = 0; i < s.n; i++) m[i] = 0;  for (int i = 0; i < s.sn; i++) m[s.ntx(i)] = 1;  };  int knapsack\_s(  int V, // [in] вместимость рюкзака  short n, // [in] количество типов предметов  const int v[], // [in] размер предмета каждого типа  const int c[], // [in] стоимость предмета каждого типа  short m[] // [out] количество предметов каждого типа {0,1}  )  {  combi::subset s(n);  int maxc = NINF, cc = 0;  short ns = s.getfirst();  while (ns >= 0)  {  if (calcv(s, v) <= V)  if ((cc = calcc(s, v, c)) > maxc)  {  maxc = cc;  setm(s, m);  }  ns = s.getnext();  };  return maxc;  }; |

Код файла Knapsack.cpp

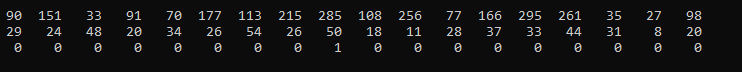


Заголовочный файл Knapsack.h

|  |
| --- |
| const int length = 18;  int weights[length];  int prices[length];  short out[length];  srand(time(NULL));  for (int i = 0; i < length; i++) {  weights[i] = (double)(rand() / (double)RAND\_MAX) \* (300 - 10) + 10;  prices[i] = (double)(rand() / (double)RAND\_MAX) \* (55 - 5) + 5;  }  for (auto w : weights) {  std::cout << std::setw(4) << w << " ";  }  std::cout << std::endl;  for (auto p : prices) {  std::cout << std::setw(4) << p << " ";  }  std::cout << std::endl;  time\_t t1 = clock();  knapsack\_s(300, length, weights, prices, out);  time\_t t2 = clock();  for (auto a : out) {  std::cout << std::setw(4) << a << " ";  }  std::cout << std::endl; |

Код файла Combinatorics.cpp

Результат:



Результат выполнения расчёта

**Задание 6.** Исследовать зависимость времени вычисления необходимое для решения задачи от размерности задачи и результат в виде графика с небольшим пояснением занести в отчет: упрощенную о рюкзаке (количество предметов 12 – 20 шт.);

Графики по рассчитанным результатам вычисления

**Вывод**: в ходе этой лабораторной работы я изучила элементы комбинаторики, реализовала возможность подсчёта размещения, сочетания и перестановки из исходного множества. А также научилась вычислять подмножества исходного множества.