|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт Информационных технологий | |
|  | |
| Кафедра Математического обеспечения и стандартизации информационных технологий | |
|  | |
|  | |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 6** | |
| **по дисциплине** | |
| **«**Структуры и алгоритмы обработки данных**»**  **Тема: «Основные алгоритмы работы с графами»** | |
|  | |
| Выполнил студент группы ИКБО-10-21 | Смольников А.Б. |
| Принял преподаватель | Филатов А.С. |

|  |  |  |
| --- | --- | --- |
| Лабораторная работа выполнена | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись руководителя)* |

Москва 2022

# **Цель работы**

Получить навыки применения методов, позволяющих сократить число переборов в задачах, которые могут быть решены только методом перебора всех возможных вариантов решения.

# **Постановка задачи**

* 1. **Задача 1**

Вариант 22.

1. Разработать алгоритм решения задачи с применением метода, указанного в варианте и реализовать программу.
2. Оценить количество переборов при решении задачи стратегией «в лоб» - грубой силы.
3. Привести анализ снижения числа переборов при применении метода.
4. Составить отчет, отобразив в нем описание выполнения всех этапов разработки, тестирования и код всей программы со скриншотами результатов тестирования.

Таблица 1. Задания варианта №22

|  |  |
| --- | --- |
| Задание 1 | Имеется определенный набор предметов П1, П2,..., Пn (каждый в единственном экземпляре); известны их веса q1, q2, ..., qn и стоимости с1, с2, ..., сn. Грузоподъемность машины равна Q. Спрашивается, какие из предметов нужно взять в машину, чтобы их суммарная стоимость (при суммарном весе ≤Q) была максимальна?  Метод: жадный алгоритм. |

# **Решение**

* 1. **Теоретическое введение**

Задачи, решаемые только полным перебором называются NP-полными задачами, которые возможно решить на недетерминированной машине Тьюринга (НМТ) за полиномиальное время. Эквивалентное определение:

NP задача – это задача, решение которой можно проверить на детерминированной машине Тьюринга за полиномиальное время.

Задача называется NP-полной, если она принадлежит классу NP и любая другая задача из NP сводится к ней за полиномиальное время. То есть речь идет о классе эквивалентных между собой задач, для эффективного решения любой из которых достаточно решить одну.

Решение задачи полным перебором представляет собой просмотр всех решений и выбор правильного(оптимального).

Жадный алгоритм — алгоритм, заключающийся в принятии локально оптимальных решений на каждом этапе, допуская, что конечное решение также окажется оптимальным.

* 1. **Функции задания №1**

Для реализации полного перебора задачи (эквивалент задачи о рюкзаке) используем статический счетчик сравнений и перебор до побитового сдвига 1 на n позиций (перебор до 2^n). Вторым циклом производится перебор всех битов числа, если бит является единицей, то инкрементируем вес и ценность. Если последние максимальны, обновим максимальное значение. Конвертируем полученное значение в двоичное число-строку и вернем ее как набор взятых вещей.

|  |
| --- |
| string BruteForce(int q[],int c[],int n,int Q){  unsigned long long best\_combination=0;  unsigned long long best=0;  bruteCnt+=pow(2,n);  for(int i=0;i<(1<<n);i++){  unsigned long long sum=0;  unsigned long long weight=0;  for(int j=0;j<n;j++){  bruteCnt+=2;  if(i&(1<<j)){  sum+=c[j];  weight+=q[j];  }  }  if(weight<=Q && sum>best){  bruteCnt++;  best=sum;  best\_combination=i;  }  }  //convert to binary  //return binary  string result;  for(int i=0;i<n;i++){  if(best\_combination&(1<<i)) result+='1';  else result+='0';  }  return result;  } |

Функция Greedy реализует жадный алгоритм решения задачи. Критерием оптимального локального решения является удельная стоимость товара, измеряющаяся в единице стоимости на единицу веса. Отсортируем массивы по данному критерию. Набираем товар, пока не достигается максимальный вес, который грузовик способен увезти. При реализации более сложного алгоритма сортировки сложность решения задачи – O(NlogN), в данном случае реализована пузырьковая сортировка, сложность O(N^2). В соответствии с методом при принятии локально оптимальных решений, решение всей задачи является оптимальным. Для более удобного восприятия отсортированные массивы выводятся на экран.

|  |
| --- |
| string Greedy(int \*&q,int \*&c,int n,int Q) {  //sort by q and c by c/q and take first elements until Q is reached  string result;  int sum = 0;  int weight = 0;  for(int i=0;i<n;i++){  crudeCnt++;  for(int j=0;j<n-1;j++){  crudeCnt+=2;  if(c[j]/q[j]<c[j+1]/q[j+1]){  int tempc=c[j];  int tempq=q[j];  c[j]=c[j+1];  q[j]=q[j+1];  c[j+1]=tempc;  q[j+1]=tempq;  }  }  }  //cout q and c  for(int i=0;i<n;i++){  cout<<q[i]<<" "<<c[i]<<endl;  }  for(int i=0;i<n;i++){  crudeCnt+=2;  if(weight+q[i]<=Q){  weight+=q[i];  sum+=c[i];  result+='1';  } else result+='0';  }  return result;  } |

* 1. **Интерфейс**

При запуске программы пользователю необходимо ввести количество вершин, количество граней и тройки значений начало-конец-вес. После этого запустится основной алгоритм, выполняющий все задачи.

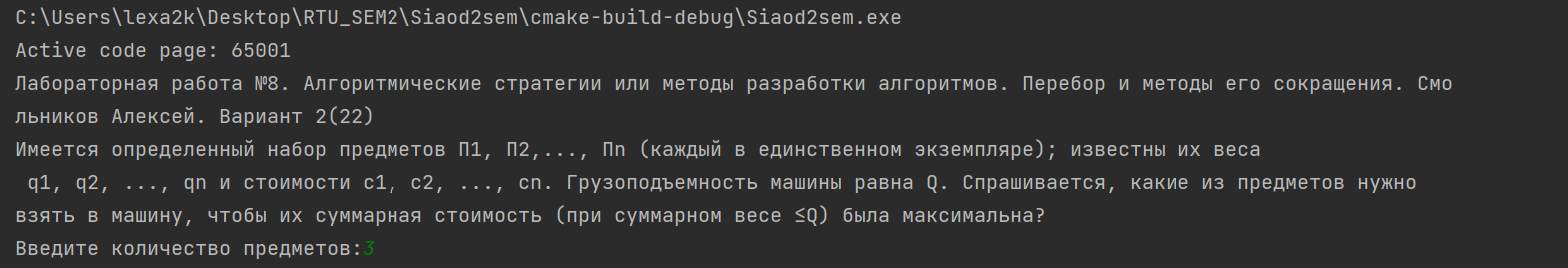


Рисунок 1. Интерфейс программы

# **Тестирование**

Тестирование проведем на данных:

* Грузоподъемность: 10;
* Количество предметов: 3;
* Веса предметов: 5 5 10;
* Ценность предметов: 10 11 20;

Аналитически решая задачу, можно понять, что оптимальный выбор грузов – первый и второй, машина заполнена полностью, вес максимальный. Из рисунков 2 и 3 видно, что алгоритмы правильно производят поиск оптимального решения.

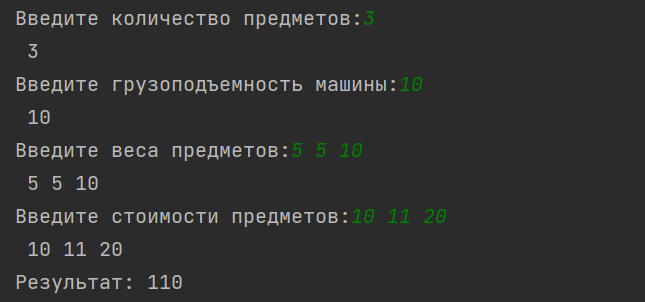


Рисунок 2. Результат полного перебора

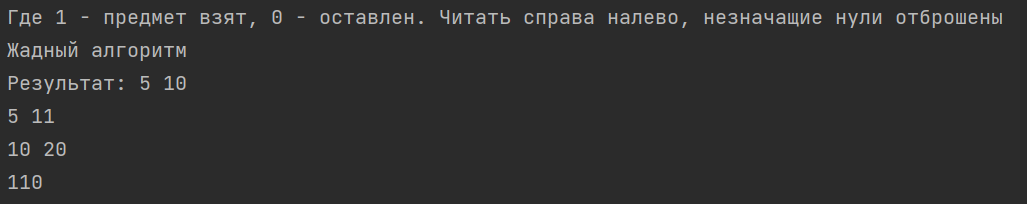


Рисунок 3. Результат жадного алгоритма

Произведем анализ снижения числа сравнений при применении метода жадного алгоритма.

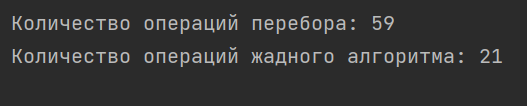


Рисунок 4. Количество операций

В соответствии с рисунком 4 видно, что количество при применении метода индивидуального задания на маленьком наборе данных в 3 предмета количество сравнений снижается в почти в 4 раза. Прирост эффективности очевиден.

# **Выводы**

В результате выполнения работы были:

* получены навыки применения методов, позволяющих сократить число переборов в задачах, решаемых полным перебором;
* были реализованы алгоритмы, связанные с решением эквивалента задачи о рюкзаке на языке программирования С++.

# **Исходный код программы**

main.cpp

|  |
| --- |
| //  // Created by lexa2k on 30.11.2022.  //  #include <iostream>  #include <string>  #include <cmath>  using namespace std;  //Имеется определенный набор предметов П1, П2,..., Пn (каждый в единственном экземпляре); известны их веса  // q1, q2, ..., qn и стоимости с1, с2, ..., сn. Грузоподъемность машины равна Q. Спрашивается, какие из предметов нужно  // взять в машину, чтобы их суммарная стоимость (при суммарном весе ≤Q) была максимальна?  static long long bruteCnt=0;  static long long crudeCnt=0;  string BruteForce(int q[],int c[],int n,int Q){  unsigned long long best\_combination=0;  unsigned long long best=0;  bruteCnt+=pow(2,n);  for(int i=0;i<(1<<n);i++){  unsigned long long sum=0;  unsigned long long weight=0;  for(int j=0;j<n;j++){  bruteCnt+=2;  if(i&(1<<j)){  sum+=c[j];  weight+=q[j];  }  }  if(weight<=Q && sum>best){  bruteCnt++;  best=sum;  best\_combination=i;  }  }  //convert to binary  //return binary  string result;  for(int i=0;i<n;i++){  if(best\_combination&(1<<i)) result+='1';  else result+='0';  }  return result;  }  string Greedy(int \*&q,int \*&c,int n,int Q) {  //sort by q and c by c/q and take first elements until Q is reached  string result;  int sum = 0;  int weight = 0;  for(int i=0;i<n;i++){  crudeCnt++;  for(int j=0;j<n-1;j++){  crudeCnt+=2;  if(c[j]/q[j]<c[j+1]/q[j+1]){  int tempc=c[j];  int tempq=q[j];  c[j]=c[j+1];  q[j]=q[j+1];  c[j+1]=tempc;  q[j+1]=tempq;  }  }  }  //cout q and c  for(int i=0;i<n;i++){  cout<<q[i]<<" "<<c[i]<<endl;  }  for(int i=0;i<n;i++){  crudeCnt+=2;  if(weight+q[i]<=Q){  weight+=q[i];  sum+=c[i];  result+='1';  } else result+='0';  }  return result;  }  int main() {  system("chcp 65001");  cout<<"Лабораторная работа №8. Алгоритмические стратегии или методы разработки алгоритмов. Перебор и методы его сокращения. Смольников Алексей. Вариант 2(22)"<<endl;  cout<<"Имеется определенный набор предметов П1, П2,..., Пn (каждый в единственном экземпляре); известны их веса\n"  " q1, q2, ..., qn и стоимости с1, с2, ..., сn. Грузоподъемность машины равна Q. Спрашивается, какие из предметов нужно\n"  "взять в машину, чтобы их суммарная стоимость (при суммарном весе ≤Q) была максимальна?"<<endl;  cout<<"Введите количество предметов: ";  int n;  cin>>n;  cout<<"Введите грузоподъемность машины: ";  int Q;  cin>>Q;  int\* q = new int[n];  cout<<"Введите веса предметов: ";  for(int i=0;i<n;i++){  cin>>q[i];  }  int\* c = new int[n];  cout<<"Введите стоимости предметов: ";  for(int i=0;i<n;i++){  cin>>c[i];  }  cout<<"Результат: "<<BruteForce(q,c,n,Q)<<endl;  cout<<"Где 1 - предмет взят, 0 - оставлен. Читать справа налево, незначащие нули отброшены"<<endl;  cout<<"Жадный алгоритм"<<endl;  cout<<"Результат: "<<Greedy(q,c,n,Q)<<endl;  cout<<"Количество операций перебора: "<<bruteCnt<<endl;  cout<<"Количество операций жадного алгоритма: "<<crudeCnt<<endl;  return 0;  } |