

|  |  |
| --- | --- |
| **Министерство образования и науки**  **Российской Федерации**  **Государственное образовательное учреждение высшего образования «МОСКОВСКИЙ АВТОМОБИЛЬНО-ДОРОЖНЫЙ**  **ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  **(МАДИ)»** |  |

**Кафедра «Высшая математика»**

**Отчет по дисциплине «Структуры и алгоритмы обработки данных»**

**Лабораторная работа №3**

**«Методы динамического программирования»**

**Выполнил:**

Учебная группа 1бПМ\_\_\_\_\_\_\_\_\_  
ФИО Кузнецов А-С.О.\_\_\_\_\_

**Принял:**

Должность \_\_\_\_\_

Звание \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ФИО \_ Кутейников И.А.\_\_\_\_\_\_

«\_\_\_ »\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024г.

Москва 2024

Цель:

С помощью методов динамического программирования реализовать

алгоритм решения зада в соответствии с вариантом. Предусмотреть ввод

начальных условий и вывод решения.

Варианты:

1 Задача о разрезании стержня;

2 Перемножение цепочки матриц;

3 Наидлиннейшая общая последовательность;

Результаты лабораторной работы оформить в виде отчета с результатами

работы программы.

Формулировка задач и их аналитическое решение представлены в книге

Т.Кормен, Ч.Лейзерсон, Р.Ривест, К.Штайн - Алгоритмы. Построение и

анализ.

в главе 15

Задача 1 - 15.1

Задача 2 - 15.2

Задача 3 - 15.3

Код:

import java.util.Scanner;  
public class Prog23 {  
 public static int matrixChainOrder(int p[], int n) {  
 int[][] m = new int[n][n];  
 int[][] s = new int[n][n];  
  
 for (int i = 1; i < n; i++) {  
 m[i][i] = 0;  
 }  
  
 for (int l = 2; l < n; l++) {  
 for (int i = 1; i < n - l + 1; i++) {  
 int j = i + l - 1;  
 m[i][j] = Integer.*MAX\_VALUE*;  
 for (int k = i; k <= j - 1; k++) {  
 int q = m[i][k] + m[k + 1][j] + p[i - 1] \* p[k] \* p[j];  
 if (q < m[i][j]) {  
 m[i][j] = q;  
 s[i][j] = k;  
 }  
 }  
 }  
 }  
  
 *printOrder*(s, 1, n - 1);  
 System.*out*.println();  
  
 return m[1][n - 1];  
 }  
  
 public static void printOrder(int s[][], int i, int j) {  
 if (i == j) {  
 System.*out*.print("A" + i);  
 } else {  
 System.*out*.print("(");  
 *printOrder*(s, i, s[i][j]);  
 *printOrder*(s, s[i][j] + 1, j);  
 System.*out*.print(")");  
 }  
 }  
  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
  
 System.*out*.print("Введите количество матриц: ");  
 int n = sc.nextInt();  
  
 int[] p = new int[n + 1];  
 System.*out*.print("Введите размеры матриц: ");  
 for (int i = 0; i < n + 1; i++) {  
 p[i] = sc.nextInt();  
 }  
  
 System.*out*.println("Минимальное количество умножений: " + *matrixChainOrder*(p, n + 1));  
 }  
}

Результат:

Введите количество матриц: 3

Введите размеры матриц: 10 50 20 30

((A1A2)A3)

Минимальное количество умножений: 16000

Вывод:

В ходе данной лабораторной работы была решена задача оптимального перемножения матриц при помощи методов динамического программирования. Было получено понимание того, как работает динамическое программирование, а также, как при помощи полученных знаний оптимизировать решения множества задач.