Національний Технічний Університет України

Київський Політехнічний Інститут імені Ігоря Сікорського

Інститут Прикладного Системного Аналізу

Кафедра Системного Проектування

ЛАБОРАТОРНА РОБОТА №1

З дисципліни: «Теорія прийняття рішень»

На тему

«БАГАТОКРИТЕРІАЛЬНИЙ ВИБІР. ВИЗНАЧЕННЯ ОПТИМАЛЬНИХ АЛЬТЕРНАТИВ ЗА ПАРЕТО ТА СЛЕЙТЕРОМ»

Виконав:

Студент групи ДА-81

Дєрюгін Є.О.

Варіант №7

Київ – 2021

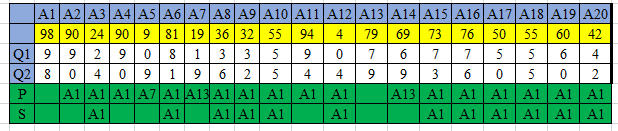
**Мета роботи:**

Ознайомитись з поняттями оптимальності за Парето та за Слейтером при багатокритеріальному виборі.

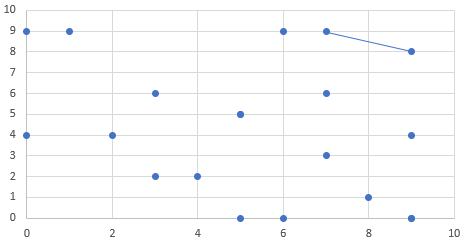
**Дані за варіантом:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № | С | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 7 | 1 | 98 | 90 | 24 | 90 | 09 | 81 | 19 | 36 | 32 | 55 | 94 | 04 | 79 | 69 | 73 | 76 | 50 | 55 | 60 | 42 |
| 2 | 79 | 84 | 93 | 05 | 21 | 67 | 04 | 13 | 61 | 54 | 26 | 59 | 44 | 02 | 02 | 06 | 84 | 21 | 42 | 68 |
| 3 | 28 | 89 | 72 | 08 | 58 | 98 | 36 | 08 | 53 | 48 | 03 | 33 | 54 | 48 | 90 | 33 | 67 | 46 | 68 | 29 |

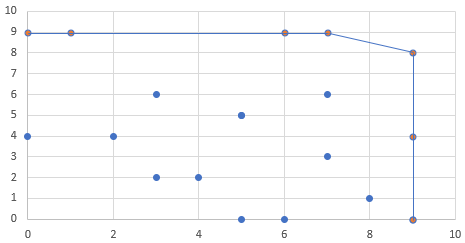
**Перший рядок:**



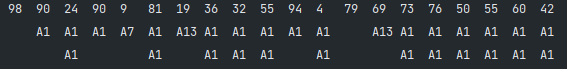
**Графічний розв’язок за Парето:**



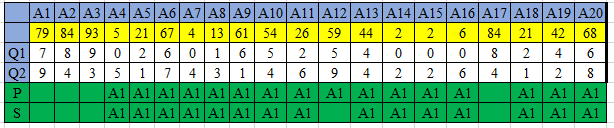
**Графічний розв’язок за Слейтером:**



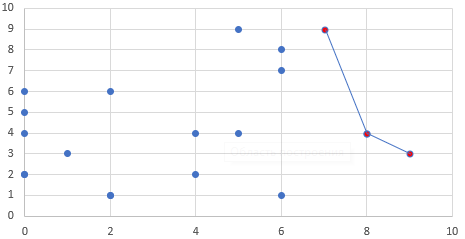
**Програмний розв’язок:**



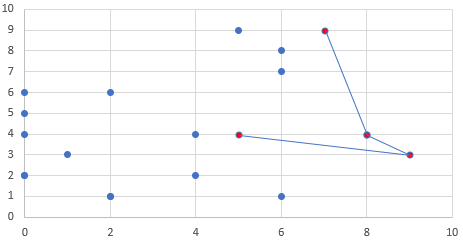
**Другий рядок:**



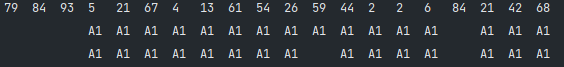
**Графічний розв’язок за Парето:**



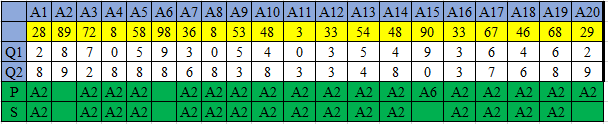
**Графічний розв’язок за Слейтером:**



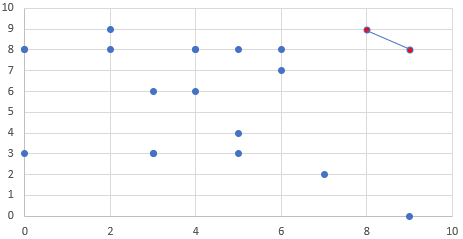
**Програмний розв’язок:**



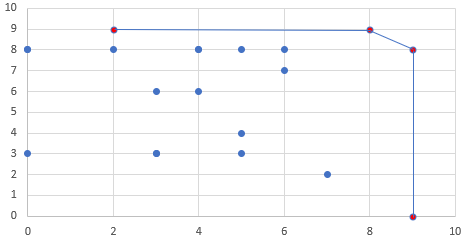
**Третій рядок:**



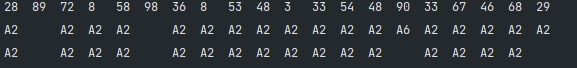
**Графічний розв’язок за Парето:**



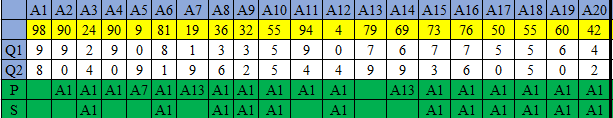
**Графічний розв’язок за Слейтером:**

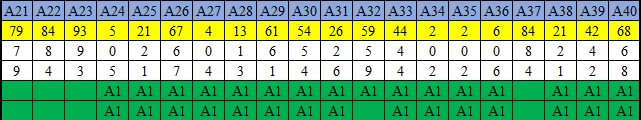


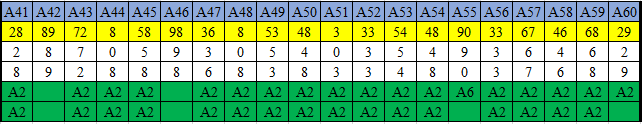
**Програмний розв’язок:**



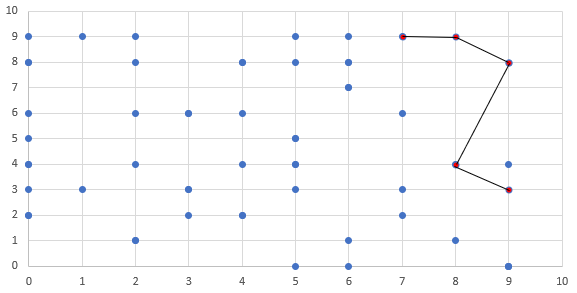
**Всі рядки:**



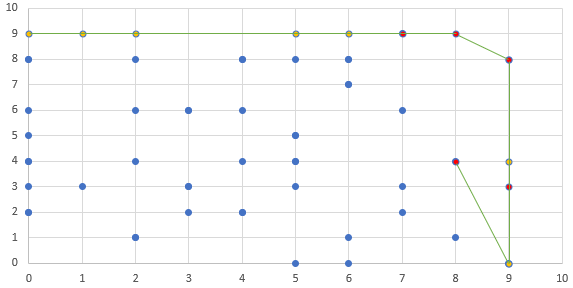




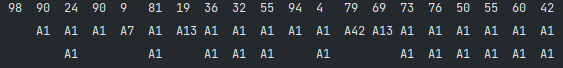
**Графічний розв’язок за Парето:**

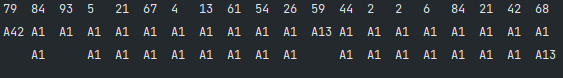


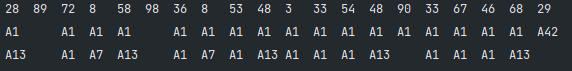
**Графічний розв’язок за Слейтером:**



**Програмний розв’язок:**







**Висновок:**

При виконанні лабораторної роботи я ознайомився з двома методами знаходження оптимального рішення:

* За Слейтером
* За Парето

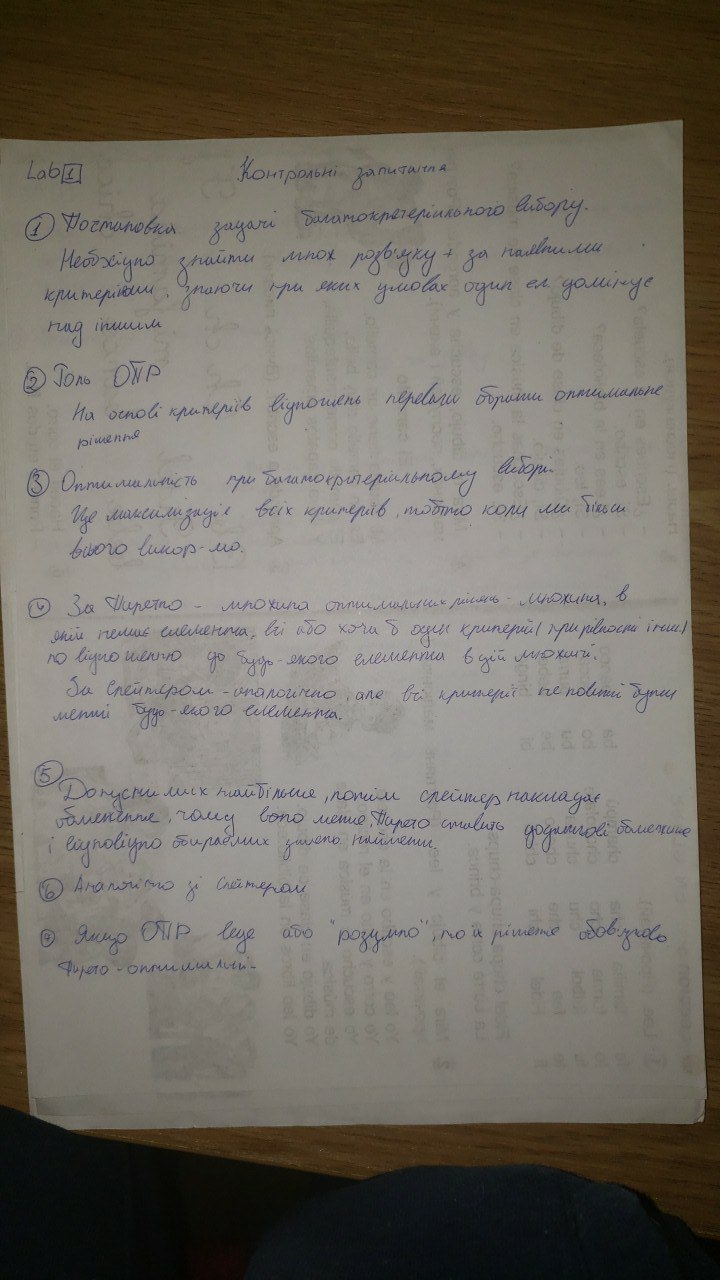
Умова Слейтера – у кінцевій множині не повинно бути елементів, що по всім критеріям менші за множини розв’язку.

Умова Парето більш строга – не повинно бути елемента, який по одному з критеріїв менший, а умови рівності інших критеріїв, та у випадку, коли всі критерії менші за інший елемент у множині результатів.

У результаті реалізації методів та знаходження графічного рішення, можна зробити висновок, що отриманий розв’язок є правильним.

Якщо порівнювати ці два методи, то маємо, що Парето накладає більше обмежень, що зменшує кількість отриманих результатів, а Слейтер потребує меншу кількість порівнянь, тому він буде швидше за Парето.

**Контрольні питання:**



**Лістинг:**

public class ParetoSleyter {  
 static int[][] *arr* = new int[][]{  
 {98, 90, 24, 90, 9, 81, 19, 36, 32, 55, 94, 4, 79, 69, 73, 76, 50, 55, 60, 42},  
 {79, 84, 93, 05, 21, 67, 04, 13, 61, 54, 26, 59, 44, 02, 02, 06, 84, 21, 42, 68},  
 {28, 89, 72, 8, 58, 98, 36, 8, 53, 48, 03, 33, 54, 48, 90, 33, 67, 46, 68, 29}};  
 static int[] *arr1* = new int[]{98, 90, 24, 90, 9, 81, 19, 36, 32, 55, 94, 4, 79, 69, 73, 76, 50, 55, 60, 42,  
 79, 84, 93, 05, 21, 67, 04, 13, 61, 54, 26, 59, 44, 02, 02, 06, 84, 21, 42, 68,  
 28, 89, 72, 8, 58, 98, 36, 8, 53, 48, 03, 33, 54, 48, 90, 33, 67, 46, 68, 29};  
  
 //домінація по парето  
 static boolean ParetoDominant(int first, int second) {  
 if ((first / 10 >= second / 10 && first % 10 > second % 10) || (first / 10 > second / 10 && first % 10 >= second % 10))  
 return true;  
 return false;  
 }  
  
 //домінація по слейтеру  
 static boolean SleyterDominant(int first, int second) {  
 if (first / 10 > second / 10 && first % 10 > second % 10) return true;  
 return false;  
 }  
  
 static int[] Pareto(int array[]) {  
 int[] res = new int[array.length];  
 for (int i = 1; i < array.length; i++) {//по одному перевіряємо вхідний массив  
 for (int j = 0; j < i; j++) {  
 if (res[i] == 0 && res[j] == 0) {  
 if (*ParetoDominant*(array[j], array[i])) {//перевіряємо чи з массиву результатів немає елемента  
 // який домінує над новим елементом  
 res[i] = j + 1;  
 //"вилучаємо елемент" з множини результатів  
 break;  
 }  
  
 if (*ParetoDominant*(array[i], array[j])) {  
 //навпаки з минулим  
 res[j] = i + 1;  
 //"вилучаємо елемент" з множини результатів  
 }  
 }  
 }  
 }  
  
 return res;  
 }  
  
 static int[] Sleyter(int array[]) {  
 int[] res = new int[array.length];  
 for (int i = 1; i < array.length; i++) {  
 for (int j = 0; j < i; j++) {  
 if (res[i] == 0 && res[j] == 0) {  
 if (*SleyterDominant*(array[j], array[i])) {//перевіряємо чи з массиву результатів немає елемента  
 // який домінує над новим елементом  
 res[i] = j + 1;  
 //"вилучаємо елемент" з множини результатів  
 break;  
 }  
 if (*SleyterDominant*(array[i], array[j])) {  
 //навпаки з минулим  
 res[j] = i + 1;  
 //"вилучаємо елемент" з множини результатів  
 }  
 }  
 }  
 }  
  
 return res;  
 }  
  
 public static void main(String[] args) {  
 int[] ParetoRes = *Pareto*(*arr1*);  
 int[] SlayterRes = *Sleyter*(*arr1*);  
 for (int i = 0; i < 60; i++)  
 System.*out*.print(*arr1*[i] + "\t");  
 System.*out*.println();  
 for (int i = 0; i < 60; i++) {  
 if (ParetoRes[i] == 0)  
 System.*out*.print(" " + "\t");  
 else  
 System.*out*.print("A" + (ParetoRes[i] + "") + "\t");  
 }  
 System.*out*.println();  
 for (int i = 0; i < 60; i++) {  
 if (SlayterRes[i] == 0)  
 System.*out*.print(" " + "\t");  
 else  
 System.*out*.print("A" + (SlayterRes[i] + "") + "\t");  
 }  
  
 }  
}