**Прізвище:** Дорош

**Ім'я:** Володимир

**По-батькові:** Юрійович

**Група:** КН-405

**Варіант:** 5

**Github:** <https://github.com/sQverful/nulp_tpr_labs>

**Кафедра:** САПР

**Дисципліна:** Теорія прийняття рішень

**Перевірив:** Кривий Р.З.

**ЗВІТ**

до лабораторної роботи №1

на тему "Прийняття рішень в умовах невизначеності та ризику**”**

**Мета роботи:** Одержання практичних навичок використання методів прийняття рішень в умовах невизначенності і ризику.

Задача. Опис

Компанія має три альтернативних варіанти своєї стратегії розвитку. Оцінка його прибутку в залежності від стану зовнішнього середовища наведено в таблиці .

А) Прийняти рішення в умовах невизначеності.

Необхідно знайти оптимальні стратегії при песимістичній оцінці (по критерію Вальда), оцінці Лапласа, по критерію Гурвіца. Значення коефіцієнта оптимізму вибрати самостійно. Результати вибору рішення відобразити в таблиці. Зробити висновки по застосуванню критеріїв. Б) Прийняти рішення в умовах ризику

Нехай отримані експертні оцінки ймовірностей стану зовнішнього середовища p1=0.5, p2=0.35, p3=0.15. Оцінити альтернативні рішення по критерію БайесаЛапласа. Результати обчислень цінностей альтернативних рішень занести в туж таблицю. Вибрати найкраще рішення. Порівняти результати вибору з отриманими раніше результатами вибору рішення в умовах невизначеності.

Порядок вирішення завдання:

1. Провести розрахунок для кожному критерію.
2. Вибрати найбільш ефективний варіант рішення.
3. Описати порядок виконання роботи і заповнити таблицю
4. Реалізувати програмне забезпечення, яке б розв'язувало дану задачу. Мова програмування неважлива. Обов'язково: дані мають зчитуватись з файлу і виводитись у табличній формі.

**Короткі теоретичні відомості:**

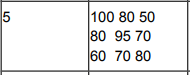
Критерій Вальда прийнятний, коли ситуація ухвалення рішення має такі характеристики:

* 1. нічого не відомо про можливість реалізації «природою» своїх стратегій;
  2. доводиться зважати на різні стратегії «природи»; 3. рішення унікальне, його можна прийняти лише один раз;

4. потрібно виключити будь-який ризик.

Критерій стійкості Гурвіца — один із способів аналізу лінійної стаціонарної динамічної системи на стійкість, розроблений німецьким математиком Адольфом Гурвіцем. Поряд з критерієм Рауса є представником сімейства алгебраїчних критеріїв стійкості, на відміну від частотних критеріїв, таких як критерій стійкості Найквіста. Перевагою методу є принципова простота, недоліком - необхідність виконання операції обчислення визначника, яка пов'язана з певними обчислювальними тонкощами (наприклад, для великих матриць може виявитися чималою обчислювальна помилка).

**Індивідуальне завдання:**



**Виконання індивідуального завдання:**

1. Критерій Баєса

W1 = 100 \* 0.5 + 80 \* 0.35 + 50 \* 0.15 = 85.5

W2 = 80 \* 0.5 + 95 \* 0.35 + 70 \* 0.15 = 83.75

W3 = 60 \* 0.5 + 70 \* 0.35 + 80 \* 0.15 = 66.5

W = max{85.5; 83.75; 66.5} = 85.5 (A1)

2. Критерій Лапласа

W1 = 1 / 3 \* (100 + 80 + 50) = 76.66

W2 = 1 / 3 \* (80 + 95 + 70) = 81.66

W3 = 1 / 3 \* (60 + 70 + 80) = 70

W = max{76.66; 81.66; 70} = 81.66 (A2)

3. Критерій Вальда

W1 = min{100; 80; 50} = 50

W2 = min{80; 95; 70} = 70

W3 = min{60; 70; 80} = 60

W = max{50; 60; 70} = 70 (A2)

4. Критерій Гурвіца

W1 = 0,4 ∙ min{100; 80; 50} + (1 - 0,4) ∙ max{100; 80; 50} = 80

W2 = 0,4 ∙ min{80; 95; 70} + (1 - 0,4) ∙ max{80; 95; 70} = 85

W3 = 0,4 ∙ min{60; 70; 80} + (1 - 0,4) ∙ max{60; 70; 80} = 72 W = max{80; 85; 72} = 85 (A2)

Код програми:

1. Метод Баєса

public static String bayes(String input) {  
 StringBuilder sbResult = new StringBuilder("[");  
 double[] coefs = ArrayProcessing.*getDoubleValuesFromFile*(input);  
 double[][] values = ArrayProcessing.*getIntValuesFromFile*(input);  
 double[] counted = new double[3];  
  
 double temp = 0.;  
 double bayesResult = 0;  
 int positionA = 0;  
  
 for (int i = 0; i < values.length; i++) {  
 for (int j = 0; j < coefs.length; j++) {  
 values[i][j] = values[i][j] \* coefs[j];  
 temp = temp + values[i][j];  
 }  
 counted[i] = temp;  
 temp = 0;  
 if (bayesResult <= counted[i]) {  
 bayesResult = counted[i];  
 positionA = i + 1;  
 }  
 }  
  
 sbResult.append(bayesResult + ", A" + positionA);  
 sbResult.append("]");  
 return sbResult.toString();  
  
}

1. Метод Лапласа

public static String laplace(String input) {  
 StringBuilder sbResult = new StringBuilder("[");  
 double[][] values = ArrayProcessing.*getIntValuesFromFile*(input);  
 double[] counted = new double[3];  
  
 double laplaceResult = 0;  
 int positionA = 0;  
  
 for (int i = 0; i < values.length; i++) {  
 for (int j = 0; j < values.length; j++) {  
 counted[i] = (counted[i] + values[i][j]);  
 }  
 counted[i] = counted[i] / 3;  
 }  
  
 for (int i = 0; i < counted.length - 1; i++) {  
 if (counted[i] <= counted[i + 1]) {  
 laplaceResult = counted[i + 1];  
 positionA = i + 2;  
 }  
 }  
  
 sbResult.append(laplaceResult + ", " + "A" + positionA);  
 sbResult.append("]");  
 return sbResult.toString();  
}

1. Метод Гурвіца

public static String gurvic(String input) {  
 StringBuilder sbResult = new StringBuilder("[");  
 double[][] values = ArrayProcessing.*getIntValuesFromFile*(input);  
 double[] counted = new double[3];  
  
 Arrays.*sort*(values[0]);  
 Arrays.*sort*(values[1]);  
 Arrays.*sort*(values[2]);  
  
 for (int i = 0; i < counted.length; i++) {  
 counted[i] = (0.4 \* values[i][0]) + ((1 - 0.4) \* values[i][2]);  
 }  
  
  
 int positionA = 0;  
 int positionB = 0;  
 double gurvicResult = 0;  
  
  
 for (int i = 0; i < counted.length; i++) {  
 for (int j = i + 1; j < counted.length; j++) {  
 if (counted[i] >= counted[j]) {  
 gurvicResult = counted[i];  
 positionA = i + 1;  
 }  
 if (gurvicResult == counted[j]) {  
 positionB = j + 1;  
 }  
 }  
 if (gurvicResult == 0) {  
 gurvicResult = counted[counted.length - 1];  
 positionA = counted.length;  
 }  
 }  
  
 if (positionB == 0) {  
 sbResult.append(gurvicResult + ", " + "A" + positionA);  
 } else {  
 sbResult.append(gurvicResult + ", " + "A" + positionA)  
 .append(", " + "A" + positionB);  
 }  
  
 sbResult.append("]");  
  
 return sbResult.toString();  
  
}

1. Метод Вальда

public static String valda(String input) {  
 StringBuilder sbResult = new StringBuilder("[");  
 double[][] values = ArrayProcessing.*getIntValuesFromFile*(input);  
 double[] counted = new double[3];  
  
 Arrays.*sort*(values[0]);  
 Arrays.*sort*(values[1]);  
 Arrays.*sort*(values[2]);  
  
 for (int i = 0; i < counted.length; i++) {  
 counted[i] = values[i][0];  
 }  
  
 int positionA = 0;  
 int positionB = 0;  
 double valdaResult = 0;  
  
  
 for (int i = 0; i < counted.length; i++) {  
 for (int j = i + 1; j < counted.length; j++) {  
 if (counted[i] >= counted[j]) {  
 valdaResult = counted[i];  
 positionA = i + 1;  
 }  
 if (valdaResult == counted[j]) {  
 positionB = j + 1;  
 }  
 }  
 if (valdaResult == 0) {  
 valdaResult = counted[counted.length - 1];  
 positionA = counted.length;  
 }  
 }  
  
 if (positionB == 0) {  
 sbResult.append(valdaResult + ", " + "A" + positionA);  
 } else {  
 sbResult.append(valdaResult + ", " + "A" + positionA)  
 .append(", " + "A" + positionB);  
 }  
  
 sbResult.append("]");  
 return sbResult.toString();  
}

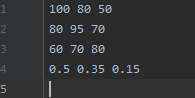


Рис.1. Файл із вхідними даними part1.txt.

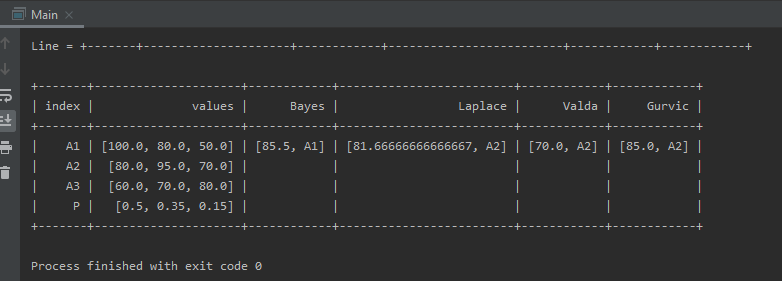


Рис.2. Результат виконання програми.

**Висновок:** Під час виконання даної лабораторної роботи, я одержав навички використання методів прийняття рішень в умовах невизначеності та ризику, реалізував на практиці критерії Баєса, Лапласа, Вальда та Гурвіца.