**Федеральное государственное образовательное**

**бюджетное учреждение**

**высшего образования**

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ**

**ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ**

**ФЕДЕРАЦИИ»**

**(Финансовый университет)**

**Факультет**

**информационных технологий и анализа больших данных**

**Кафедра «Бизнес-информатика»**

**Домашнее задание № 3**

«Решение биматричных и антагонистических игр.»

Студенты группы БИ20-8:

Луканина Полина

Аверкин Никита

Филимонова Арина

Совин Владимир

Горшков Георгий

Киселева Евгения

Руководитель:

Аксенов Дмитрий Андреевич

**Москва 2022**

### Оглавление

[Оглавление 2](#_Toc100248082)

[1. Постановка задачи (физическая модель) 5](#_Toc100248083)

[2. Математическая модель 6](#_Toc100248084)

[2.1. Решение задачи о нахождении выигрышной стратегии для антагонистической игры 6](#_Toc100248085)

[2.2. Решение задачи о нахождении выигрышной стратегии для биматричной игры 9](#_Toc100248086)

[3. Алгоритмы решения задачи 11](#_Toc100248087)

[3.1. Решение задачи о нахождении выигрышной стратегии для 11](#_Toc100248088)

[антагонистической игры 11](#_Toc100248089)

[3.1.1. Описание входных данных. 11](#_Toc100248090)

[3.1.2. Описание алгоритма решения 12](#_Toc100248091)

[3.1.3. Описание выходных данных 13](#_Toc100248092)

[3.2. Решение задачи о нахождении выигрышной стратегии для биматричной игры 14](#_Toc100248093)

[3.2.1. Описание входных данных. 14](#_Toc100248094)

[3.1.2. Описание алгоритма решения 15](#_Toc100248095)

[3.1.3. Описание выходных данных 16](#_Toc100248096)

[4. Варианты использования системы 16](#_Toc100248097)

[4.1. Варианты использования в условиях антагонистической игры 17](#_Toc100248098)

[4.1.1. ВИ 1 17](#_Toc100248099)

[4.1.2. ВИ 2 18](#_Toc100248100)

[4.1.3. ВИ 3 19](#_Toc100248101)

[4.2. Варианты использования в условиях биматричной игры 21](#_Toc100248102)

[4.2.1. ВИ 1 21](#_Toc100248103)

[4.2.2. ВИ 2 23](#_Toc100248104)

[4.2.3. ВИ 3 25](#_Toc100248105)

[5. Архитектура решения 27](#_Toc100248106)

[5.1 Алгоритм 1. Для антагонистической стратегии 27](#_Toc100248107)

[5.1.1. Функции считывания информации 27](#_Toc100248108)

[5.1.2. Функции обработки информации 31](#_Toc100248109)

[5.1.3. Функции вывода информации 35](#_Toc100248110)

[5.2 Алгоритм 2. Для биматричной стратегии 38](#_Toc100248111)

[5.2.1. Функции считывания информации 38](#_Toc100248112)

[5.2.2. Функции обработки информации 44](#_Toc100248113)

[5.2.3. Функции вывода информации 50](#_Toc100248114)

[6 Тестирование 51](#_Toc100248115)

[6.1. Тестирование задачи о нахождении выигрышной стратегии для 52](#_Toc100248116)

[антагонистической игры 52](#_Toc100248117)

[6.1.1. Проверка №1 матрица 3x3: 52](#_Toc100248118)

[6.1.2. Проверка №2 матрица 3x3: 53](#_Toc100248119)

[6.1.3. Проверка №3 матрица 3x3: 54](#_Toc100248120)

[6.1.4. Проверка №4 матрица 4x4: 55](#_Toc100248121)

[6.1.5. Проверка №5 матрица 4x4: 57](#_Toc100248122)

[6.2. Тестирование задачи о нахождении выигрышной стратегии для 59](#_Toc100248123)

[биматричной игры 59](#_Toc100248124)

[6.2.1. Проверка №1 матрица 3x3 59](#_Toc100248125)

[6.2.1.2. Тест №1 онлайн-калькулятором: 60](#_Toc100248126)

[6.2.2. Проверка №2 матрица 3x3: 61](#_Toc100248127)

[6.2.3. Проверка №3 матрица 3x3: 63](#_Toc100248128)

[6.2.4. Проверка №4 матрица 4x4: 65](#_Toc100248129)

[6.2.5. Проверка №5 матрица 4x4: 67](#_Toc100248130)

[7. Заключение 69](#_Toc100248131)

[7.1. Заключение по антагонистической задаче 69](#_Toc100248132)

[7.2. Заключение по биматричной задаче 72](#_Toc100248133)

# Постановка задачи (физическая модель)

Хакер пытается взломать бд одной международной консалтинговой компании с целью кражи и дальнейшей перепродажи персональной информации о клиентах данной компании. Служба безопасности знает, что планируется атака на ее сервисы, но не знает когда и какими способами это произойдёт. У хакера есть ряд инструментов для попытки проникновения в систему: DDOS-атака, запуск трояна или поиск уязвимого места, через которое можно будет залезть в систему. У службы безопасности так же есть определённые протоколы, которые позволят отразить атаку: все внешние письма, ссылки и прочие файлы, которые прилетают сотрудникам по каналам связи проверять на вирусы; установка капчи на сайт для отражения ddos атак; усиленная проверка сотрудников на предмет вноса/выноса несанкционированной информации, смена паролей у сотрудников на более сложные. Исходя из этой информации СБ составила весовую матрицу:

Таблица 1. Весовая матрица заказчика

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Смена паролей на рабочих местах сотрудников | Тщательная проверка сотрудников на входе и выходе | Проверка входящих файлов | Капча на сайтах |
| Ddos-атака | 5 | 0 | 15 | 90 |
| Запуск вируса (троян) | 40 | 10 | 95 | 13 |
| Взлом рабочего места сотрудника | 90 | 20 | 99 | 7 |

# Математическая модель

Таблица 2. Исходные данные:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | B1 | B2 | B3 | B4 |
| A1 | 5 | 0 | 15 | 90 |
| A2 | 40 | 10 | 95 | 13 |
| A3 | 90 | 20 | 99 | 7 |

## 2.1. Решение задачи о нахождении выигрышной стратегии для антагонистической игры

1. Проверяем, имеет ли платежная матрица седловую точку. Если да, то выписываем решение игры в чистых стратегиях.

Считаем, что игрок I выбирает свою стратегию так, чтобы получить максимальный свой выигрыш, а игрок II выбирает свою стратегию так, чтобы минимизировать выигрыш игрока I.

Таблица 3. Находим верхние и нижние границы игры

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Игроки | B1 | B2 | B3 | B4 | a = min(Ai) |
| A1 | 5 | 0 | 15 | 90 | 0 |
| A2 | 40 | 10 | 95 | 13 | 10 |
| A3 | 90 | 20 | 99 | 7 | 7 |
| b = max(Bi) | 90 | 20 | 99 | 90 |  |

Находим гарантированный выигрыш, определяемый нижней ценой игры a = max(ai) = 10, которая указывает на максимальную чистую стратегию A2.

Верхняя цена игры b = min(bj) = 20.

Что свидетельствует об отсутствии седловой точки, так как a ≠ b, тогда цена игры находится в пределах 10 ≤ y ≤ 20. Находим решение игры в смешанных стратегиях. Объясняется это тем, что игроки не могут объявить противнику свои чистые стратегии: им следует скрывать свои действия. Игру можно решить, если позволить игрокам выбирать свои стратегии случайным образом (смешивать чистые стратегии).

Так как игроки выбирают свои чистые стратегии случайным образом, то выигрыш игрока I будет случайной величиной. В этом случае игрок I должен выбрать свои смешанные стратегии так, чтобы получить максимальный средний выигрыш.

Аналогично, игрок II должен выбрать свои смешанные стратегии так, чтобы минимизировать математическое ожидание игрока I.

3. Находим решение игры в смешанных стратегиях.

Находим минимум функции (F(x) = x1+x2+x3 → min) при ограничениях (для игрока II):

5x1+40x2+90x3 ≥ 1

10x2+20x3 ≥ 1

15x1+95x2+99x3 ≥ 1

90x1+13x2+7x3 ≥ 1

Находим максимум функции (Z(y) = y1+y2+y3+y4 → max) при ограничениях (для игрока I):

5y1+15y3+90y4 ≤ 1

40y1+10y2+95y3+13y4 ≤ 1

90y1+20y2+99y3+7y4 ≤ 1

Решаем прямую задачу линейного программирования симплексным методом, с использованием симплексной таблицы:

Определим максимальное значение целевой функции Z(Y) = y1+y2+y3+y4 при следующих условиях-ограничений:

5y1+15y3+90y4≤1

40y1+10y2+95y3+13y4≤1

90y1+20y2+99y3+7y4≤1

4. Получим, что оптимальный план двойственной задачи равен:

x1 = 13/1800, x2 = 0, x3 = 1/20

F(X) = 1\*13/1800+1\*0+1\*1/20 = 103/1800

Цена игры будет равна g = 1/F(x), а вероятности применения стратегий игроков:

qi = g\*yi; pi = g\*xi.

Цена игры: g = 1: 103/1800 = 1800/103, где:

p1 = 1800/103\*13/1800 = 13/103

p2 = 1800/103\*0 = 0

p3 = 1800/103\*1/20 = 90/103

Оптимальная смешанная стратегия игрока I: P = (13/103; 0; 90/103), где:

q1 = 1800/103\*0 = 0

q2 = 1800/103\*83/1800 = 83/103

q3 = 1800/103\*0 = 0

q4 = 1800/103\*1/90 = 20/103

Оптимальная смешанная стратегия игрока II: Q = (0; 83/103; 0; 20/103)

Цена игры: v=1800/103

## 2.2. Решение задачи о нахождении выигрышной стратегии для биматричной игры

Рассмотрим конфликтную ситуацию, в которой каждый из двух участников имеет следующие возможности для выбора своей линии поведения:

1) игрок А – может выбрать любую из стратегий А1, …, Аm,

2) игрок В – любую из стратегий В1, …, Вn.

При этом их совместный выбор оценивается вполне определённо: если игрок А выбрал i-ю стратегию Аi, а игрок В – k-ю стратегию Вk, то в итоге выигрыш игрока А будет равен некоторому числу aik, а выигрыш игрока В некоторому, вообще говоря, другому числу bik.

Последовательно перебирая все стратегии игрока А и все стратегии игрока В, мы сможем заполнить их выигрышами две таблицы.

Первая из таблиц описывает выигрыш игрока А, а вторая – выигрыш игрока В. Имея одну из таблиц, введем еще одну, чтобы условие было полноценным. Преобразовав таблицы в матрицы, получим:

Таблица 4. Платежная матрица игрока А

|  |  |  |  |
| --- | --- | --- | --- |
| 5 | 0 | 15 | **90** |
| 40 | 10 | 95 | 13 |
| **90** | **20** | **99** | 7 |

Позиции максимумов в столбцах матрицы А: (3,1), (3,2), (3,3), (1,4)

Таблица 5. Платежная матрица игрока Б

|  |  |  |  |
| --- | --- | --- | --- |
| 15 | **55** | 25 | 14 |
| **90** | 5 | 0 | 35 |
| 40 | 20 | 30 | **70** |

Позиции максимумов в строках матрицы В: (1,2), (2,1), (3,4)

Если биматричная игра не имеет равновесных ситуаций в чистых стратегиях, то она неразрешима в чистых стратегиях. Поскольку здесь отсутствует равновесие по Нэшу, то задача не разрешима по чистой стратегии. Тогда можно искать решение в смешанных стратегиях.

Пусть игрок А выбирает стратегию А1, с вероятностью р1, А2 – р2, …, Аm – pm, причём:

P1 ≥ 0, p2 ≥ 0, …..., pm ≥ 0, ∑(m) pi = 1

Игрок В использует стратегию В1 с вероятностью q1, B2 – q2, …, Bn – qn, причём:

Q1 ≥ 0, q2 ≥ 0, …., qn ≥ 0, ∑(n) qj = 1

В качестве критерия "удачности" игры возьмём математические ожидания выигрыша игроков, которые вычисляются по формулам:

Va = ∑(m)∑(n) aij pi qj

Vb = ∑(n)∑(m) bij pi qj

Распределение вероятностей Р\* и Q\* (определяют равновесную ситуацию, если для любых других распределений P и Q одновременно выполнены следующие неравенства:

Va (P, Q\*) ≤  Va (P\*, Q\*), Vb (P\*, Q) ≤  Vb (P\*, Q\*)

Если равновесная ситуация существует, то отклонение от неё невыгодно самому игроку.

Также справедлива теорема Дж. Нэша. Всякая биматричная игра имеет хотя бы одну равновесную ситуацию в смешанных стратегиях.

Итак, в первое неравенство системы последовательно подставляются все чистые стратегии игрока А, при предположении, что В придерживается своей оптимальной стратегии. Во второе неравенство подставляются все чистые стратегии игрока В, при предположении, что А придерживается своей оптимальной стратегии.

Получаем систему m+n неравенств, решение которой дает значение элементов оптимальных смешанных стратегий (P\*, Q\*) и платежи, получаемые игроками в точке равновесия.

Получим цены игр для матричных игр: va=1749/103, vb=283/4. Следовательно, функция полезности по Нэшу примет вид:  
U=(H1-1749/103)(H2-283/4)

# Алгоритмы решения задачи

Алгоритмы решения реализованы с помощью программного кода в Python.

## 3.1. Решение задачи о нахождении выигрышной стратегии для

## антагонистической игры

### 3.1.1. Описание входных данных.

Вид входных данных зависит от способа, которым будут вводиться данными. Данный алгоритм позволяет самим ввести данные, или ввести с помощью CSV файла.

Для ручного ввода входными данными являются:

* Изображение выглядит как текст

  Автоматически созданное описаниеКоличество стратегий для компании А, название стратегий для компании А, количество стратегий для компании В, название стратегий для компании Б, значения весовой матрицы.

Рисунок 1. Ручной способ ввода данных

* Изображение выглядит как текст

  Автоматически созданное описаниеКоличество стратегий игрока А.

Рисунок 2. Весовая матрица

Импорт входных данных из csv файла:

* Весовая матрица с названиями столбцов и строк.



Рисунок 3. 3 способ ввода информации в код

### 3.1.2. Описание алгоритма решения

После того как данные введены, программе необходимо преобразовать данные для дальнейшего использования.

Шаг 1: берем весовую матрицу, находим минимум каждой строки, а далее - максимум — это получилась цена игры для игрока А.

Шаг 2: находим максимум для каждого столбца весовой матрицы, и затем находим минимальное из них — это цена игры для игрока B.

Шаг 3: Переход к канонической форме задачи линейного программирования путем введения неотрицательных дополнительных балансовых (базисных) переменных. Запись задачи в симплекс-таблицу. Между системой ограничений задачи и симплекс-таблицей взаимно-однозначное соответствие. Строчек в таблице столько, сколько равенств в системе ограничений, а столбцов - столько, сколько свободных переменных. Базисные переменные заполняют первый столбец, свободные - верхнюю строку таблицы.

Шаг 4: Проверка опорного плана на оптимальность. Для этого необходимо анализировать строку целевой функции F. Если найдется хотя бы один коэффициент индексной строки меньше нуля, то план не оптимальный, и его необходимо улучшить.

Шаг 5: Улучшение опорного плана. Из отрицательных коэффициентов индексной строки выбирается наибольший по абсолютной величине. Затем элементы столбца свободных членов симплексной таблицы делит на элементы того же знака ведущего столбца. Далее идет построение нового опорного плана.

Шаг 6: Выписывание оптимального решения.

### 3.1.3. Описание выходных данных

В конце программа рассчитает оптимальную чистую стратегию, цену игры для игрока А при выборе чистой оптимальной стратегии, оптимальная чистая стратегия для игрока Б (текст), цена игры для игрока Б при выборе чистой оптимальной стратегии, таблица смешанных стратегий для игрока А, цена игры для игрока А при выборе смешанной оптимальной стратегии.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 4. Полученный ответ

## 3.2. Решение задачи о нахождении выигрышной стратегии для биматричной игры

### 3.2.1. Описание входных данных.

Вид входных данных зависит от способа, которым будут вводиться данными. Данный алгоритм позволяет самим ввести данные, или ввести с помощью CSV файла.

Для ручного ввода входными данными являются:

* Количество стратегий для компании А, название стратегий для компании А, количество стратегий для компании В, название стратегий для компании Б, значения весовой матрицы.

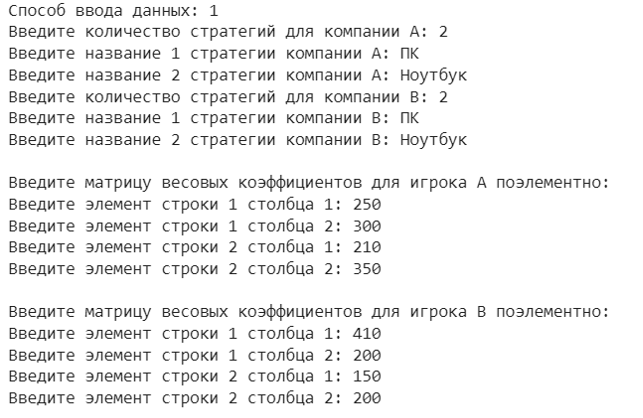


Рисунок 5. Терминал питона при 1 способе ввода

* Количество стратегий игрока А, количество стратегий игрока В



Рисунок 6. Терминал питона при 2 способе ввода данных

Импорт входных данных из csv файла:

* Весовая матрица с названиями столбцов и строк.



Рисунок 7. Терминал питона при 3 способе ввода данных

### 3.1.2. Описание алгоритма решения

После того как данные введены, программе необходимо преобразовать данные для дальнейшего использования.

Шаг 1: берем весовую матрицу, находим минимум каждой строки, а далее - максимум — это получилась цена игры для игрока А.

Шаг 2: находим максимум для каждого столбца весовой матрицы, и затем находим минимальное из них — это цена игры для игрока B.

Шаг 3: Переход к канонической форме задачи линейного программирования путем введения неотрицательных дополнительных балансовых (базисных) переменных. Запись задачи в симплекс-таблицу. Между системой ограничений задачи и симплекс-таблицей взаимно-однозначное соответствие. Строчек в таблице столько, сколько равенств в системе ограничений, а столбцов - столько, сколько свободных переменных. Базисные переменные заполняют первый столбец, свободные - верхнюю строку таблицы. Нижняя строка называется индексной, в ней записываются коэффициенты при переменных в целевой функции.

Шаг 4: Проверка опорного плана на оптимальность. Для этого необходимо анализировать строку целевой функции F. Если найдется хотя бы один коэффициент индексной строки меньше нуля, то план не оптимальный, и его необходимо улучшить.

Шаг 5: Улучшение опорного плана. Из отрицательных коэффициентов индексной строки выбирается наибольший по абсолютной величине. Затем элементы столбца свободных членов симплексной таблицы делит на элементы того же знака ведущего столбца. Далее идет построение нового опорного плана.

Шаг 6: Расчет равновесий по Нешу, для этого создается матрица с координатами максимальных значений по строке для первого игрока и для второго, а после этого сравниваем их на равенство.

### 3.1.3. Описание выходных данных

В конце программа рассчитает оптимальную чистую стратегию, цену игры для игрока А при выборе чистой оптимальной стратегии, оптимальная чистая стратегия для игрока Б (текст), цена игры для игрока Б при выборе чистой оптимальной стратегии, таблица смешанных стратегий для игрока А, цена игры для игрока А при выборе смешанной оптимальной стратегии.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 8. Результат выполнения кода

# Варианты использования системы

В нашей системе есть три варианта использования.

## 4.1. Варианты использования в условиях антагонистической игры

### 4.1.1. ВИ 1

Данный вариант использования включает в себя ручной ввод данных с клавиатуры. Для того, чтобы его активировать в графу «Каким способом вы хотите ввести значения?» надо ввести цифру «1».

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 9. Терминал кода при ВИ 1

После этого необходимо ввести количество стратегий для компании А цифрой. (Например, 2).

Затем нужно ввести названия стратегий, их количество зависит от того, сколько стратегий вы указали ранее. (Например, ноутбук).

Далее переходим к заполнению данных для второй компании. Мы также вводим количество стратегий для компании В, а также названия стратегий.

После этого вы должны ввести данные для заполнения весовой матрицы. Количество значений зависит от того, сколько стратегий вы указали ранее. (Например, 100).

После этого от пользователя требуется лишь нажатие клавиши «Enter» и на экране выведется оптимальная чистая стратегия, цена игры для игрока А при выборе чистой оптимальной стратегии, оптимальная чистая стратегия для игрока B, цена игры для игрока B при выборе чистой оптимальной стратегии, цена игры для игрока А при выборе смешанной оптимальной стратегии, таблица смешанных стратегий для игрока А, таблица смешанных стратегий для игрока B.

### 4.1.2. ВИ 2

Данный вариант позволяет сгенерировать весовую матрицу, для его выбора в первой строке необходимо ввести «2».

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 10. Автоматическая генерация весовой матрицы

Далее Вам необходимо ввести количество стратегий игрока А.

После этого от пользователя требуется лишь нажатие клавиши «Enter» и на экране выведется оптимальная чистая стратегия, цена игры для игрока А при выборе чистой оптимальной стратегии, оптимальная чистая стратегия для игрока B, цена игры для игрока B при выборе чистой оптимальной стратегии, цена игры для игрока А при выборе смешанной оптимальной стратегии, таблица смешанных стратегий для игрока А, таблица смешанных стратегий для игрока B.

### 4.1.3. ВИ 3

Данный вариант использования включает в себя ввод данных с помощью файла csv. Для того, чтобы его активировать в графу «Каким способом вы хотите ввести значения?» надо ввести цифру «3».

После этого появляется окно, в котором вводим путь к csv файлу. Например:



Рисунок 11. Выбор местонахождения csv файла



Рисунок 12. Данные следует вводить таким образом:

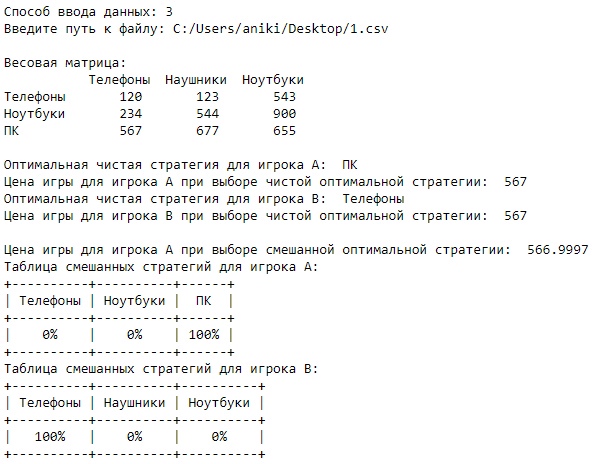
После этого получаем результат.

Рисунок 13. Пример:

## 4.2. Варианты использования в условиях биматричной игры

### 4.2.1. ВИ 1

Данный вариант использования включает в себя ручной ввод данных с клавиатуры. Для того, чтобы его активировать в графу «Каким способом вы хотите ввести значения?» надо ввести цифру «1».

Изображение выглядит как текст, стол

Автоматически созданное описание

Рисунок 14. Терминал при выборе 1 ВИ

После этого необходимо ввести количество стратегий для компании А цифрой. (Например, 2).

Затем нужно ввести названия стратегий, их количество зависит от того, сколько стратегий вы указали ранее. (Например, ноутбук).

Далее переходим к заполнению данных для второй компании. Мы также вводим количество стратегий для компании В, а также названия стратегий.

После этого вы должны ввести данные для заполнения весовой матрицы. Количество значений зависит от того, сколько стратегий вы указали ранее. (Например, 100).

После этого от пользователя требуется лишь нажатие клавиши «Enter» и на экране выведется оптимальная чистая стратегия, цена игры для игрока А при выборе чистой оптимальной стратегии, оптимальная чистая стратегия для игрока B, цена игры для игрока B при выборе чистой оптимальной стратегии, цена игры для игрока А при выборе смешанной оптимальной стратегии, таблица смешанных стратегий для игрока А, таблица смешанных стратегий для игрока B, количество равновесий по Нешу.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 15. Результат запроса

### 4.2.2. ВИ 2

Данный вариант позволяет сгенерировать весовую матрицу, для его выбора в первой строке необходимо ввести «2».

Изображение выглядит как текст, оранжевый, снимок экрана

Автоматически созданное описание

Рисунок 16. Терминал при 2 ВИ

Далее Вам необходимо ввести количество стратегий игрока А.

После этого от пользователя требуется лишь нажатие клавиши «Enter» и на экране выведется оптимальная чистая стратегия, цена игры для игрока А при выборе чистой оптимальной стратегии, оптимальная чистая стратегия для игрока B, цена игры для игрока B при выборе чистой оптимальной стратегии, цена игры для игрока А при выборе смешанной оптимальной стратегии, таблица смешанных стратегий для игрока А, таблица смешанных стратегий для игрока B, количество равновесий по Нешу.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 17. Оптимальная чистая стратегия для игрока А

### 4.2.3. ВИ 3

Данный вариант использования включает в себя ввод данных с помощью файла csv. Для того, чтобы его активировать в графу «Каким способом вы хотите ввести значения?» надо ввести цифру «3».

После этого появляется окно, в котором вводим путь к csv файлу.



Рисунок 18. Например:



Рисунок 19. Данные следует вводить таким образом:

# Архитектура решения

Для решения задачи использовались методы (функции), которые можно разделить на 3 принципиальных кода.

## Алгоритм 1. Для антагонистической стратегии

### 5.1.1. Функции считывания информации

После запуска программы необходимо определиться каким способом будет происходить ввод данных: 1 – ручной ввод; 2 – случайные числа; 3 – файл csv.

**Если введено «1»:**

**Входные данные:**

* Нет входных данных

**Выходные данные:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix – список значений весовой матрицы (тип данных: list);

**Переменные, затрагиваемые в ходе работы:**

* sum\_strategy\_A – количество стратегий для компании А (тип данных: int);
* sum\_strategy\_В – количество стратегий для компании В (тип данных: int);
* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix – список значений весовой матрицы (тип данных: list);
* name – название стратегий (тип данных: str);
* value – элементы весовой матрицы (тип данных: int);

**Если введено «2»:**

**Входные данные:**

* Нет входных данных

**Выходные данные:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix – список значений весовой матрицы (тип данных: list);

**Переменные, затрагиваемые в ходе работы:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix – список значений весовой матрицы (тип данных: list);
* size\_rows – количество стратегий игрока А (тип данных: int);
* х – генерируемые данные весовой матрицы (тип данных: int);

**Если ведено «3»:**

**Входные данные:**

* Нет входных данных

**Выходные данные:**

* matrix – список значений весовой матрицы (тип данных: list);
* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);

**Переменные, затрагиваемые в ходе работы:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix – список значений весовой матрицы (тип данных: list);
* file\_way – путь к файлу (тип данных: str).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 20. Фрагмент кода с данными переменными

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 21. Фрагмент кода с данными переменными

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 22. Фрагмент кода с данными переменными

### 5.1.2. Функции обработки информации

После того, как вы введете все необходимые данные, программа их получит и начнет первичную обработку.

При выборе ручного ввода все необходимые данные заносятся в словари, с помощью метода append ().

В случае случайной генерации, нужные данные генерируются с помощью random.randint (), при этом программа получает случайное целое число в заданном диапазоне. После этого данные также заносятся в пустые словари.

Если ввод данных осуществляется с помощью файла csv, то для начала файл необходимо открыть и прочитать программе, а затем разделить данные и также занести их в пустой словарь. Более того, программа образует список с названиями колонок матрицы, а также список с названиями строк. Не мало важен и перевод коэффициентов из типа данных str в тип int.

После всех необходимых операций с данными реализуются важные функции, рассчитывающие оптимальную чистую стратегию и оптимальную смешанную стратегию.

В функции pure\_strategy реализуется оптимальная чистая стратегия. Она представляет собой нахождение минимума в каждой строке, а также нахождения максимума в каждом столбце. Это находится с помощью функции min(), max().

В функции nash\_equilibrium реализуется оптимальная смешанная стратегия. С помощью функции linprog() находится оптимальное решение переменных с помощью которых мы находим процентное соотношение по стратегиям. Более того, здесь также составляются таблицы смешанных стратегий.

Далее следует подробное описание входных, выходных и переменных, затрагивается в ходе работы.

**Функция pure\_strategy:**

**Входные данные:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);

**Выходные данные:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix – список значений весовой матрицы (тип данных: list);

**Переменные, затрагиваемые в ходе работы:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* max\_min – максимальное значение по столбцу весовой матрицы (тип данных: list);
* min\_max – минимальное значение по строке весовой матрицы (тип данных: list);
* index\_row – индекс по строке весовой матрицы (тип данных: int);
* index\_column – индекс по столбцу весовой матрицы (тип данных: int).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 23. Фрагмент кода с данными переменными

**Функция nash\_equilibrium:**

**Входные данные:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);

**Выходные данные:**

* res — это массив с вероятностями стратегий игроков (тип данных: scipy.optimize.optimize.OptimizeResult );
* mytable – таблицы, которые выводят стратегии и проценты (тип данных: prettytable.prettytable.PrettyTable);

**Переменные, затрагиваемые в ходе работы:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* res - это массив с вероятностями стратегий игроков (тип данных: scipy.optimize.optimize.OptimizeResult );
* mytable – таблицы, которые выводят стратегии и проценты (тип данных: prettytable.prettytable.PrettyTable);
* a\_T — это транспонированная весовая матрица (тип данных: numpy.ndarray);
* mix\_strategy\_A — это вероятности стратегий в процентах игрока А (тип данных: list);
* mix\_strategy\_B — это вероятности стратегий в процентах игрока В (тип данных: list);

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 24. Фрагмент кода с данными переменными

### 5.1.3. Функции вывода информации

Метод вывода информации (он заключен внутри каждой функции)

Что делает: осуществляет вывод необходимой информации

Вывод информации осуществляется с помощью функции print ()

Затрагиваемые переменные:

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix – список значений весовой матрицы (тип данных: list);
* res — это массив с вероятностями стратегий игроков (тип данных: scipy.optimize.optimize.OptimizeResult );
* mytable – таблицы, которые выводят стратегии и проценты (тип данных: prettytable.prettytable.PrettyTable);

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 25. Вывод оптимальных стратегий для обоих игроков



Рисунок 26. Цена для игрока А



Рисунок 27. Вывод таблицы смешанных стратегий для игрока А



Рисунок 28. Вывод таблицы смешанных стратегий для игрока Б

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 29. Результат выполнения запроса при 1 ВИ

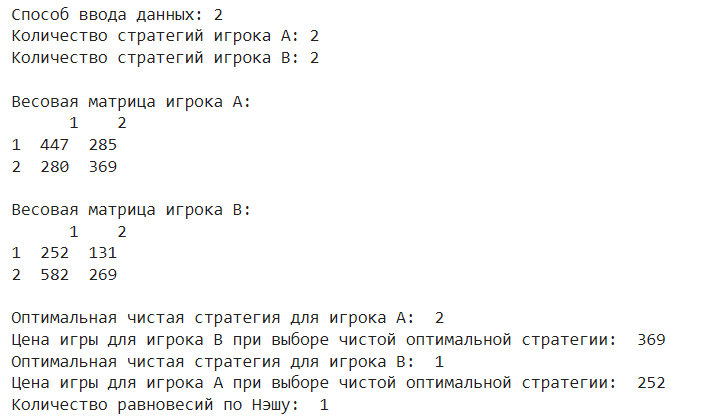


Рисунок 30. Результат выполнения запроса при 2 ВИ

## Алгоритм 2. Для биматричной стратегии

### 5.2.1. Функции считывания информации

После запуска программы необходимо определиться каким способом будет происходить ввод данных: 1 – ручной ввод; 2 – случайные числа; 3 – файл csv.

1. **Если введено «1»:**

**Входные данные:**

* Нет входных данных

**Выходные данные:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix – список значений весовой матрицы (тип данных: list);

**Переменные, затрагиваемые в ходе работы:**

* sum\_strategy\_A – количество стратегий для компании А (тип данных: int);
* sum\_strategy\_В – количество стратегий для компании В (тип данных: int);
* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix\_А – список значений весовой матрицы для компании А (тип данных: list);
* matrix\_B – список значений весовой матрицы для компании B (тип данных: list);
* name – название стратегий (тип данных: str);
* value – элементы весовой матрицы (тип данных: int);

1. **Если введено «2»:**

**Входные данные:**

* Нет входных данных

**Выходные данные:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix\_А – список значений весовой матрицы для компании А (тип данных: list);
* matrix\_B – список значений весовой матрицы для компании B (тип данных: list);

**Переменные, затрагиваемые в ходе работы:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix\_А – список значений весовой матрицы для компании А (тип данных: list);
* matrix\_B – список значений весовой матрицы для компании B (тип данных: list);
* size\_rows – количество стратегий игрока А (тип данных: int);
* х – генерируемые данные весовой матрицы (тип данных: int);

1. **Если ведено «3»:**

**Входные данные:**

* Нет входных данных

**Выходные данные:**

* matrix\_А – список значений весовой матрицы для компании А (тип данных: list);
* matrix\_B – список значений весовой матрицы для компании B (тип данных: list);
* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);

**Переменные, затрагиваемые в ходе работы:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix\_А – список значений весовой матрицы для компании А (тип данных: list);
* matrix\_B – список значений весовой матрицы для компании B (тип данных: list);
* file\_way – путь к файлу (тип данных: str).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 31. Часть кода, в которой затрагиваются эти переменные

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 32. Часть кода, в которой затрагиваются эти переменные

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 33. Список для имен строк и столбцов матрицы

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 34. Часть кода, в которой затрагиваются эти переменные

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 35. Часть кода, в которой затрагиваются эти переменные

### 5.2.2. Функции обработки информации

После того, как вы введете все необходимые данные, программа их получит и начнет первичную обработку.

При выборе ручного ввода все необходимые данные заносятся в словари, с помощью метода append ().

В случае случайной генерации, нужные данные генерируются с помощью random.randint(), при этом программа получает случайное целое число в заданном диапазоне. После этого данные также заносятся в пустые словари.

Если ввод данных осуществляется с помощью файла csv, то для начала файл необходимо открыть и прочитать программе, а затем разделить данные и также занести их в пустой словарь. Более того, программа образует список с названиями колонок матрицы, а также список с названиями строк. Не мало важен и перевод коэффициентов из типа данных str в тип int.

После всех необходимых операций с данными реализуются важные функции, рассчитывающие оптимальную чистую стратегию и оптимальную смешанную стратегию.

В функции pure\_strategy реализуется оптимальная чистая стратегия. Она представляет собой нахождение минимума в каждой строке, а также нахождения максимума в каждом столбце. Это находится с помощью функции min (), max (). Более того, в этой функции рассчитывается количество равновесий по Нешу с помощью цикла for i in range.

В функции Mix\_strategy реализуется оптимальная смешанная стратегия. С помощью цикла происходит подсчет элементов и вычисляется значения оптимальных стратегий. Более того, здесь также составляются таблицы смешанных стратегий игрока А и игрока Б.

Далее следует подробное описание входных, выходных и переменных, затрагивается в ходе работы.

**Функция pure\_strategy:**

**Входные данные:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix\_А – список значений весовой матрицы для компании А (тип данных: list);
* matrix\_B – список значений весовой матрицы для компании B (тип данных: list);
* matrix\_А\_DF – список значений весовой матрицы для компании А (тип данных: list);
* matrix\_B\_DF – список значений весовой матрицы для компании B (тип данных: list);

**Выходные данные:**

* nash – количество равновесий по Нешу (тип данных: int);
* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* min\_max\_A – минимальное значение по строке весовой матрицы для игрока А (тип данных: list);
* min\_max\_B – минимальное значение по строке весовой матрицы для игрока В (тип данных: list);

**Переменные, затрагиваемые в ходе работы:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* min\_max\_A – минимальное значение по строке весовой матрицы для игрока А (тип данных: list);
* min\_max\_B – минимальное значение по строке весовой матрицы для игрока В (тип данных: list);
* index\_row – индекс по строке весовой матрицы (тип данных: int);
* index\_column – индекс по столбцу весовой матрицы (тип данных: int).
* matrix\_А – список значений весовой матрицы для компании А (тип данных: list);
* matrix\_B – список значений весовой матрицы для компании B (тип данных: list);
* matrix\_А\_DF – список значений весовой матрицы для компании А (тип данных: list);
* matrix\_B\_DF – список значений весовой матрицы для компании B (тип данных: list);
* nash\_A – список максимальных значений в матрице игрока А (тип данных: list);
* nash\_B - список максимальных значений в матрице игрока В (тип данных: list);
* nash – количество равновесий по Нешу (тип данных: int);

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 36. Часть кода, в которой затрагиваются эти переменные

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 37. Часть кода, в которой затрагиваются эти переменные

**Функция Mix\_strateg:**

**Входные данные:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix\_А – список значений весовой матрицы для компании А (тип данных: list);
* matrix\_B – список значений весовой матрицы для компании B (тип данных: list);

**Выходные данные:**

* res - это массив с вероятностями стратегий игроков (тип данных: scipy.optimize.optimize.OptimizeResult );
* mytable – таблицы, которые выводят стратегии и проценты (тип данных: prettytable.prettytable.PrettyTable);
* price\_game\_B – цена игры А (тип данных: int);
* price\_game\_A – цена игры В (тип данных: int);

**Переменные, затрагиваемые в ходе работы:**

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* matrix\_А – список значений весовой матрицы для компании А (тип данных: list);
* matrix\_B – список значений весовой матрицы для компании B (тип данных: list);
* price\_game\_B – цена игры А (тип данных: int);
* price\_game\_A – цена игры В (тип данных: int);
* mytable – таблицы, которые выводят стратегии и проценты (тип данных: prettytable.prettytable.PrettyTable);
* mix\_strategy\_A — это вероятности стратегий в процентах игрока А (тип данных: list);
* mix\_strategy\_B — это вероятности стратегий в процентах игрока В (тип данных: list);
* help\_matrix\_A – вспомогательная таблица матрицы А (тип данных: list);
* help\_matrix\_B - вспомогательная таблица матрицы А (тип данных: list);

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 38. Часть кода, в которой затрагиваются эти переменные

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 39. Часть кода, в которой затрагиваются эти переменные

### 5.2.3. Функции вывода информации

Метод вывода информации (он заключен внутри каждой функции)

Что делает: осуществляет вывод необходимой информации

Вывод информации осуществляется с помощью функции print ()

Затрагиваемые переменные:

* name\_rows – список названий стратегий компании А (тип данных: list);
* name\_columns – список названий стратегий компании В (тип данных: list);
* mytable – таблицы, которые выводят стратегии и проценты (тип данных: prettytable.prettytable.PrettyTable);
* matrix\_А – список значений весовой матрицы для компании А (тип данных: list);
* matrix\_B – список значений весовой матрицы для компании B (тип данных: list);
* price\_game\_B – цена игры А (тип данных: int);
* price\_game\_A – цена игры В (тип данных: int);
* min\_max\_A – минимальное значение по строке весовой матрицы для игрока А (тип данных: list);
* min\_max\_B – минимальное значение по строке весовой матрицы для игрока В (тип данных: list);

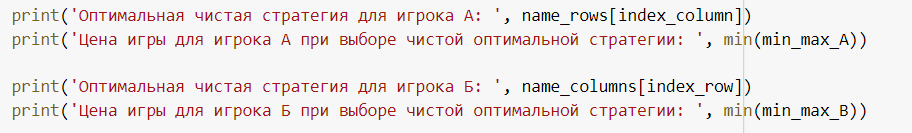


Рисунок 40. Вывод оптимальных чистых стратегий для обоих игроков



Рисунок 41. Вывод общей цены игры



Рисунок 42. Количество равновесий по Нэшу

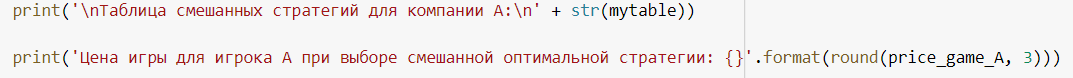


Рисунок 43. Вывод таблицы смешанных стратегий для игрока А

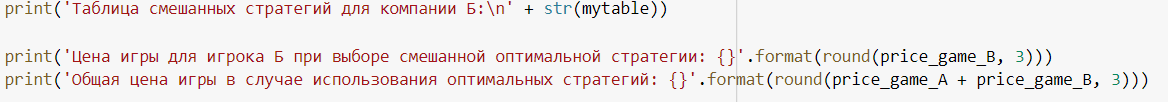


Рисунок 44. Вывод таблицы смешанных стратегий для игрока Б

# Тестирование

1. Проведём тестирование нашей программы и сравним полученные показатели, чтобы сделать вывод о предпочтительном варианте использования нашей программы или онлайн-калькулятора[[1]](#footnote-1) под условия заказчика.

## 6.1. Тестирование задачи о нахождении выигрышной стратегии для

## антагонистической игры

### 6.1.1. Проверка №1 матрица 3x3:

#### 6.1.1.1. Тест №1 кодом Python:

Выбираем метод ввода информации, количество стратегий (размер матрицы) и получаем оптимальную стратегию.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 45. Результат обработки запроса в питоне 1

#### 6.1.1.2. Тест №1 онлайн-калькулятором:

Выбираем размер матрицы и вводим необходимые данные в предназначенные поля. Метод решений: аналитический. Все сопутствующие «галочки» необходимо снять.

Изображение выглядит как текст, снимок экрана, внутренний

Автоматически созданное описание

Рисунок 46. Ответ в онлайн-калькуляторе 1

### 6.1.2. Проверка №2 матрица 3x3:

#### 6.1.2.1. Тест №2 кодом Python:

Выбираем метод ввода информации, количество стратегий (размер матрицы) и получаем оптимальную стратегию.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 47. Результат обработки запроса в питоне 2

#### 6.1.2.2. Тест №2 онлайн-калькулятором:

Выбираем размер матрицы и вводим необходимые данные в предназначенные поля. Метод решений: аналитический. Все сопутствующие «галочки» необходимо снять.

Изображение выглядит как текст, внутренний, снимок экрана

Автоматически созданное описание

Рисунок 48. Ответ в онлайн-калькуляторе 2

### 6.1.3. Проверка №3 матрица 3x3:

#### 6.1.3.1. Тест №3 кодом Python:

Выбираем метод ввода информации, количество стратегий (размер матрицы) и получаем оптимальную стратегию.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 49. Результат обработки запроса в питоне 3

#### 6.1.3.2. Тест №3 онлайн-калькулятором:

Выбираем размер матрицы и вводим необходимые данные в предназначенные поля. Метод решений: линейного программирования.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 50. Ответ в онлайн-калькуляторе 3

### 6.1.4. Проверка №4 матрица 4x4:

#### 6.1.4.1. Тест №4 кодом Python:

Выбираем метод ввода информации, количество стратегий (размер матрицы) и получаем оптимальную стратегию.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 51. Результат обработки запроса в питоне 4

#### 6.1.4.2. Тест №4 онлайн-калькулятором:

Выбираем размер матрицы и вводим необходимые данные в предназначенные поля. Метод решений: линейного программирования.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 52. Ответ в онлайн калькуляторе 4

### 6.1.5. Проверка №5 матрица 4x4:

#### 6.1.5.1. Тест №5 кодом Python:

Выбираем метод ввода информации, количество стратегий (размер матрицы) и получаем оптимальную стратегию.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 53. Результат обработки запроса в питоне 5

#### 6.1.5.2. Тест №5 онлайн-калькулятором:

Выбираем размер матрицы и вводим необходимые данные в предназначенные поля. Метод решений: линейного программирования.

Изображение выглядит как текст, стол

Автоматически созданное описание

Рисунок 54. Ответ онлайн-калькулятора 5

## 6.2. Тестирование задачи о нахождении выигрышной стратегии для

## биматричной игры

Проведём тестирование нашей программы и сравним полученные показатели, чтобы сделать вывод о предпочтительном варианте использования нашей программы или онлайн-калькулятора[[2]](#footnote-2) под условия заказчика.

### 6.2.1. Проверка №1 матрица 3x3

#### 6.2.1.1. Тест №1 кодом Python:

Выбираем метод ввода информации, размер матрицы и получаем оптимальные и смешанные стратегии для обоих игроков.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 55. Результат обработки запроса в питоне 1

### 6.2.1.2. Тест №1 онлайн-калькулятором:

Выбираем размер матрицы и вводим необходимые данные в предназначенные поля. Запускаем программу.

Изображение выглядит как текст, снимок экрана, внутренний

Автоматически созданное описание

Рисунок 56. Ответ в онлайн калькуляторе 1

### 6.2.2. Проверка №2 матрица 3x3:

#### 6.2.2.1. Тест №2 кодом Python:

Выбираем метод ввода информации, размер матрицы и получаем оптимальные и смешанные стратегии для обоих игроков.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 57. Результат обработки запроса в питоне 2

#### 6.2.2.2. Тест №2 онлайн-калькулятором:

Выбираем размер матрицы и вводим необходимые данные в предназначенные поля. Запускаем программу.

Изображение выглядит как текст, снимок экрана, внутренний

Автоматически созданное описание

Рисунок 58. Ответ в онлайн-калькуляторе 2

### 6.2.3. Проверка №3 матрица 3x3:

#### 6.2.3.1. Тест №3 кодом Python:

Выбираем метод ввода информации, размер матрицы и получаем оптимальные и смешанные стратегии для обоих игроков.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 59. Результат обработки запроса в питоне 3

#### 6.2.3.2. Тест №3 онлайн-калькулятором:

Выбираем размер матрицы и вводим необходимые данные в предназначенные поля. Запускаем программу.

Изображение выглядит как текст, снимок экрана, внутренний

Автоматически созданное описание

Рисунок 60. Ответы онлайн-калькулятора 3

### 6.2.4. Проверка №4 матрица 4x4:

#### 6.2.4.1. Тест №4 кодом Python:

Выбираем метод ввода информации, размер матрицы и получаем оптимальные и смешанные стратегии для обоих игроков.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 61. Результат обработки запроса в питоне 4

#### 6.2.4.2. Тест №4 онлайн-калькулятором:

Выбираем размер матрицы и вводим необходимые данные в предназначенные поля. Запускаем программу.

Изображение выглядит как текст, снимок экрана, внутренний

Автоматически созданное описание

Рисунок 62. Ответы онлайн-калькулятора 4

### 6.2.5. Проверка №5 матрица 4x4:

#### 6.2.5.1. Тест №5 кодом Python:

Выбираем метод ввода информации, размер матрицы и получаем оптимальные и смешанные стратегии для обоих игроков.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 63. Результат обработки запроса в питоне 5

#### 6.2.5.2. Тест №5 онлайн-калькулятором:

Выбираем размер матрицы и вводим необходимые данные в предназначенные поля. Запускаем программу.

Изображение выглядит как текст, снимок экрана, внутренний

Автоматически созданное описание

Рисунок 64. Ответ онлайн-калькулятора 5

# 7. Заключение

## 7.1. Заключение по антагонистической задаче

Наш представленный код решает поставленную задачу. На основании тестирования данного алгоритма можно сделать вывод о том, что Python выводит самое оптимальное решение достаточно быстро. Ниже представлено решение поставленной задачи через Python и проверка в онлайн-калькуляторе:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 65. Результат выполнения антагонистической задачи в питоне

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 66. Ответ на антагонистическую задачу в онлайн-калькуляторе

Теперь сравним два алгоритма по критериям: эффективности, скорости использования алгоритма, простоты использования, надёжности в разрезе человеческого фактора и точности предоставляемого решения.

Таблица 6. Сравнение онлайн-калькулятора и питона

|  |  |  |
| --- | --- | --- |
| **Критерий** | **Python** | **Онлайн-калькулятор** |
| Эффективность | Высокая | Высокая |
| Скорость использования алгоритма | Высокая | Средняя |
| Простота использования | Высокая | Средняя |
| Надёжность (человеческий фактор) | Высокая | Средняя |
| Точность | Высокая | Высокая |

Мы считаем, что представленный рукописный код на языке Python лучше, потому что он удобнее, быстрее и проще, а ещё имеет импорт .csv файлов и случайный ввод данных. Улучшением кода, к примеру, может послужить время выполнения запроса.

## 7.2. Заключение по биматричной задаче

Наш представленный код решает поставленную задачу. На основании тестирования данного алгоритма можно сделать вывод о том, что Python выводит самое оптимальное решение достаточно быстро.

Теперь сравним два алгоритма по критериям: эффективности, скорости использования алгоритма, простоты использования, надёжности в разрезе человеческого фактора и точности предоставляемого решения.

Таблица 7. Сравнение онлайн-калькулятора и питона

|  |  |  |
| --- | --- | --- |
| **Критерий** | **Python** | **Онлайн-калькулятор** |
| Эффективность | Высокая | Средняя |
| Скорость использования алгоритма | Высокая | Высокая |
| Простота использования | Высокая | Высокая |
| Надёжность (человеческий фактор) | Высокая | Высокая |
| Точность | Высокая | Высокая |

Мы считаем, что представленный рукописный код на языке Python лучше, потому что он удобнее, быстрее и проще, а ещё имеет импорт .csv файлов и случайный ввод данных. Улучшением кода, к примеру, может послужить время выполнения запроса.

1. <https://math.semestr.ru/games/antogonist.php> [↑](#footnote-ref-1)
2. <https://math.semestr.ru/games/bimatrix.php> [↑](#footnote-ref-2)