МИНОБРНАУКИ РОССИИ
ФГБОУ ВО «СГУ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

УНИВЕРСАЛЬНЫЕ АЛГЕБРЫ И АЛГЕБРА ОТНОШЕНИЙ

ЛАБОРАТОРНАЯ РАБОТА

студента 3 курса 331 группы
направления 10.05.01 — Компьютерная безопасность
факультета КНиИТ
Токарева Никиты Сергеевича

Проверил
аспирант ______________ В. Н. Кутин

Саратов 2022

1 Постановка задачи

Цель работы – изучение основных понятий универсальной алгебры и операций над бинарными отношениями.

Порядок выполнения работы:

1. Рассмотреть понятие алгебраической операции и классификацию свойств операций. Разработать алгоритмы проверки свойств операций: ассоциативность, коммутативность, идемпотентность, обратимость, дистрибутивность.
2. Рассмотреть основные операции над бинарными отношениями. Разработать алгоритмы выполнения операции над бинарными отношениями.
3. Рассмотреть основные операции над матрицами. Разработать алгоритмы выполнения операций над матрицами.

2 Теоретические сведения по рассмотренным темам с их обоснованием

2.1 Понятие алгебраической операции

Отображение $f : A^n \to A$ называется алгебраической n-арной операцией или просто **алгебраической операцией** на множестве A. При этом n называется порядком или арностью алгебраической операции f.

Далее для бинарной операции f по возможности будем использовать мультипликативную запись с помощью символа «·», т.е.вместо $f(x, y)$ писать $x \cdot y$. При необходимости для бинарной операции f используется также аддитивная запись с помощью символа «+», т.е. вместо $f(x, y)$ записывается $x + y$.

2.2 Классификация свойств операций

Бинарная операция $\cdot$ на множестве A называется:

— идемпотентной, если $\forall x \in A$ выполняется равенство $x \cdot x = x$;
— коммутативной, если $\forall x, y \in A$ выполняется равенство $x \cdot y = y \cdot x$;
— ассоциативной, если $\forall x, y, z \in A$ выполняется равенство $x \cdot (y \cdot z) = (x \cdot y) \cdot z$;
— обратимой, если $\forall x, y \in A$, если уравнения $x \cdot a = y$ и $b \cdot x = y$ имеют решение, причем единственное;
— дистрибутивной относительно операции +, если $\forall x, y, z \in A$ выполняются равенства

$$x \cdot (y + z) = (x \cdot y) + (x \cdot z),$$
$$(y + z) \cdot x = (y \cdot x) + (z \cdot x);$$

2.3 Основные операции над бинарными отношениями

— Над бинарными отношениями можно выполнять любые теоретико-множественные операции, в частности операции объединения $\cup$, пересечения $\cap$ и дополнения $\neg$;
— Обратным для бинарного отношения $\rho \subset A \times B$ называется бинарное отношение $\rho^{-1} \subset B \times A$, определяющееся по формуле:

$$\rho^{-1} = \{(b, a) : (a, b) \in \rho\};$$

— Композицией бинарных отношений $\rho \subset A \times B$ и $\sigma \subset B \times C$ называется бинарное отношение $\rho \circ \sigma \subset A \times C$, определяющееся по формуле:

$$\rho \circ \sigma = \{(a, c) : (a, b) \in \rho \text{ и } (b, c) \in \sigma \text{ для некоторого } b \in B\};$$

2.4 Основные операции над матрицами

— Сложение и вычитание матриц.

Суммой $A + B$ матриц $A_{m\times n} = (a_{ij})$ и $B_{m\times n} = (b_{ij})$ называется матрица $C_{m\times n} = (c_{ij})$, где $c_{ij} = a_{ij} + b_{ij}$ для всех $i = \overline{1, m}$ и $j = \overline{1, n}$.

Разностью $A - B$ матриц $A_{m\times n} = (a_{ij})$ и $B_{m\times n} = (b_{ij})$ называется матрица $C_{m\times n} = (c_{ij})$, где $c_{ij} = a_{ij} - b_{ij}$ для всех $i = \overline{1, m}$ и $j = \overline{1, n}$.

— Умножение матрицы на число.

Произведением матрицы $A_{m\times n} = (a_{ij})$ на число α называется матрица $C_{m\times n} = (c_{ij})$, где $c_{ij} = \alpha a_{ij}$ для всех $i = \overline{1, m}$ и $j = \overline{1, n}$.

— Произведение двух матриц.

Произведением матриц $A_{m\times n} = (a_{ij})$ на матрицу $B_{m\times n} = (b_{ij})$ называется матрица $C_{m\times n} = (c_{ij})$, где $c_{ij} = \sum\limits_{k=1}^{n} a_{ik}b_{kj}$ для всех $i = \overline{1, m}$ и $j = \overline{1, n}$.

— Транспонирование матрицы.

Транспонированной по отношению к матрице $A_{m\times n} = (a_{ij})$ называется матрица $A^T_{n\times m} = (a^T_{ij})$ для элементов которой $a^T_{ij} = a_{ji}$.

— Обращение матрицы.

Обращение матрицы $A_{m\times n}$ - получение матрицы A^{-1}, обратной к исходной матрице A. Обратная матрица A^{-1} — такая, при умножении которой на исходную матрицу A получается единичная матрица E. Это такая матрица, которая удовлетворяет равенству

$$AA^{-1} = A^{-1}A = E$$

3 Результаты работы

3.1 Описание алгоритмов проверки свойств операций: ассоциативность, коммутативность, идемпотентность, обратимость, дистрибутивность

Алгоритм 1 – Проверка бинарной операции «·» на идемпотентность

Вход: Матрица $A = (a_{ij})$ бинарной операции «·» размерности $n \times n$, список элементов l множества X над операцией «·».

Выход: «Бинарная операция «·» обладает свойством идемпотентности» или «Бинарная операция «·» не обладает свойством идемпотентности».

Шаг 1. Инициализировать булеву переменную $is\_idempotent = true$.

Шаг 2. Пройти по всем элементам $l[i]$ множества X, где $0 \leq i < n$. Если хотя бы один элемент $a_{ii} \neq l[i]$, то присвоить $is\_idempotent = false$.

Шаг 3. Если после завершения цикла $is\_idempotent = true$, то вывести в консоль следующее сообщение: «Бинарная операция «·» обладает свойством идемпотентности». Если $is\_idempotent = false$ – вывести сообщение: «Бинарная операция «·» не обладает свойством идемпотентности».

Оценка сложности данного алгоритма равна $O(n)$.

Алгоритм 2 – Проверка бинарной операции «·» на коммутативность

Вход: Матрица $A = (a_{ij})$ бинарной операции «·» размерности $n \times n$.

Выход: «Бинарная операция «·» обладает свойством коммутативности» или «Бинарная операция «·» не обладает свойством коммутативности».

Шаг 1. Инициализировать булеву переменную $is\_commutative = true$.

Шаг 2. Пройти по всем элементам a_{ij} матрицы A, где $0 \leq i,j < n$. Если хотя бы один элемент $a_{ij} \neq a_{ji}$, то присвоить $is\_commutative = false$.

Шаг 3. Если после завершения цикла $is\_commutative = true$, то вывести в консоль следующее сообщение: «Бинарная операция «·» обладает свойством коммутативности». Если $is\_commutative = false$ – вывести сообщение: «Бинарная операция «·» не обладает свойством коммутативности».

Оценка сложности данного алгоритма равна $O(n^2)$.

Алгоритм 3 – Проверка бинарной операции «·» на ассоциативность

Вход: Матрица $A = (a_{ij})$ бинарной операции «·» размерности $n \times n$, список элементов l множества X над операцией «·».

Выход: «Бинарная операция «·» обладает свойством ассоциативности» или «Бинарная операция «·» не обладает свойством ассоциативности».

Шаг 1. Инициализировать булеву переменную $is\_associative = true$.

Шаг 2. Необходимо пройти по всем элементам a_{ip} и a_{rk}, где p и r – индексы элементов $l[p] = a_{jk}$ и $l[r] = a_{ij}$ соответственно ($0 \leq i,j,k < n$). Если хотя бы один элемент $a_{ip} \neq a_{rk}$, то присвоить $is\_associative = false$.

Шаг 3. Если после завершения цикла $is\_associative = true$, то вывести в консоль следующее сообщение: «Бинарная операция «·» обладает свойством ассоциативности». Если $is\_associative = false$ – вывести сообщение: «Бинарная операция «·» не обладает свойством ассоциативности».

Оценка сложности данного алгоритма равна $O(n^3)$.

Алгоритм 4 – Проверка бинарной операции «·» на обратимость

Вход: Матрица $A = (a_{ij})$ бинарной операции «·» размерности $n \times n$, список элементов l множества X над операцией «·».

Выход: «Бинарная операция «·» обладает свойством обратимости» или «Бинарная операция «·» не обладает свойством обратимости».

Шаг 1. Инициализировать булеву переменную $is\_invertible = false$.

Шаг 2. Необходимо пройти по всем элементам a_{ij}, где $0 \leq i,j < n$ для того чтобы посчитать единицы на каждой j-й строке и i-м столбце. Если условие $a_{ij} = 1$, то увеличить счетчик $cnt_1 = cnt_1 + 1$, и если $a_{ji} = 1$, то увеличить счетчик $cnt_2 = cnt_2 + 1$. По завершении прохождения по текущей j-й строке и i-м столбце если $cnt_1 = cnt_2 = n$, то присвоить $is\_invertible = true$ и обнулить счетчики $cnt_1 = 0$ и $cnt_2 = 0$ для следующей итерации.

Шаг 3. Если после завершения цикла $is\_invertible = true$, то вывести в консоль следующее сообщение: «Бинарная операция «·» обладает свойством обратимости». Если $is\_invertible = false$ – вывести сообщение: «Бинарная операция «·» не обладает свойством обратимости».

Оценка сложности данного алгоритма равна $O(n^2)$.

Алгоритм 5 – Проверка бинарной операции «·» на дистрибутивность

Вход: Матрица $A = (a_{ij})$ бинарной операции «$\cdot$» размерности $n \times n$, матрица $B = (b_{ij})$ бинарной операции «$+$» размерности $n \times n$, список элементов l множества X над операцией «$\cdot$» и «$+$».
Выход: «Бинарная операция «$\cdot$» обладает свойством дистрибутивности» или «Бинарная операция «$\cdot$» не обладает свойством дистрибутивности».
Шаг 1. Инициализировать булеву переменную $is\_distributive = true$.
Шаг 2. Необходимо пройти по всем элементам a_{ip} и b_{rq}, где p, r и q – индексы элементов $l[p] = b_{jk}$, $l[r] = a_{ij}$, $l[q] = a_{ik}$ соответственно, а также по всем элементам a_{si} и b_{uv}, где s, u и v – индексы элементов $l[s] = b_{jk}$, $l[u] = a_{ji}$, $l[v] = a_{ki}$ соответственно ($0 \leq i,j,k < n$). Если хотя бы один элемент $a_{ip} \neq b_{rq}$ или $a_{si} \neq b_{uv}$, то присвоить $is\_distributive = false$.
Шаг 3. Если после завершения цикла $is\_distributive = true$, то вывести в консоль следующее сообщение: «Бинарная операция «$\cdot$» обладает свойством дистрибутивности». Если $is\_distributive = false$ – вывести сообщение: «Бинарная операция «$\cdot$» не обладает свойством дистрибутивности».

Оценка сложности данного алгоритма равна $O(n^3)$.

3.2 Описание алгоритмов выполнения операций над бинарными отношениями

Алгоритм 6 – Построение операции объединения бинарных отношений
Вход: Матрица $A = (a_{ij})$ бинарного отношения ρ размерности $n \times n$, матрица $B = (b_{ij})$ бинарного отношения σ размерности $n \times n$.
Выход: Матрица $C = (c_{ij})$ бинарного отношения $\rho \cup \sigma$ размерности $n \times n$.
Шаг 1. Элементы c_{ij} матрицы C находятся так что: $c_{ij} = a_{ij} + b_{ij}$, где $0 \leq i, j < n$.
Шаг 2. Вернуть матрицу C в качестве выхода функции.

Оценка сложности данного алгоритма равна $O(n^2)$.

Алгоритм 7 – Построение операции пересечения бинарных отношений
Вход: Матрица $A = (a_{ij})$ бинарного отношения ρ размерности $n \times n$, матрица $B = (b_{ij})$ бинарного отношения σ размерности $n \times n$.
Выход: Матрица $C = (c_{ij})$ бинарного отношения $\rho \cap \sigma$ размерности $n \times n$.
Шаг 1. Элементы c_{ij} матрицы C находятся так что: $c_{ij} = a_{ij} \cdot b_{ij}$, где $0 \leq i, j < n$.

Шаг 2. Вернуть матрицу C в качестве выхода функции.

Оценка сложности данного алгоритма равна $O(n^2)$.

Алгоритм 8 – Построение операции дополнения бинарного отношения

Вход: Матрица $A = (a_{ij})$ бинарного отношения ρ размерности $n \times n$.
Выход: Матрица $A^{'} = (a^{'}_{ij})$ бинарного отношения $\neg\rho$ размерности $n \times n$.
Шаг 1. Элементы $a^{'}_{ij}$ матрицы $A^{'}$ находятся так что: $a^{'}_{ij} = 1 - a_{ij}$, где $0 \leq i, j < n$.
Шаг 2. Вернуть матрицу $A^{'}$ в качестве выхода функции.

Оценка сложности данного алгоритма равна $O(n^2)$.

Алгоритм 9 – Построение обратной операции бинарных отношений

Вход: Матрица $A = (a_{ij})$ бинарного отношения ρ размерности $n \times n$.
Выход: Матрица $A^T = (a^T_{ij})$ бинарного отношения ρ^{-1}
Шаг 1. Необходимо получить матрицу $A^T = (a^T_{ij})$ путем транспонирования матрицы $A = (a_{ij})$. Элементы a^T_{ij} матрицы A^T находятся так что: $a^T_{ij} = a_{ji}$, где $0 \leq i, j < n$.
Шаг 2. Вернуть матрицу A^T бинарного отношения ρ^{-1} в качестве выхода функции.

Оценка сложности данного алгоритма равна $O(n^2)$.

Алгоритм 10 – Построение операции композиции бинарных отношений

Вход: Матрица $A = (a_{ij})$ бинарного отношения ρ размерности $n \times n$, матрица $B = (b_{ij})$ бинарного отношения σ размерности $n \times n$.
Выход: Матрица $C = (c_{ij})$ бинарного отношения $\rho \circ \sigma$ размерности $n \times n$.
Шаг 1. Элементы c_{ij} матрицы C находятся следующим образом:

$$c_{ij} = \sum_{k=0}^{l} a_{ik} \cdot b_{kj}, \text{ где } 0 \leq i, j < n.$$

Шаг 2. Вернуть матрицу C бинарного отношения $\rho \circ \sigma$ в качестве выхода функции.

Оценка сложности данного алгоритма равна $O(n^3)$.

3.3 Описание алгоритмов выполнения операций над матрицами

Алгоритм 11 – Построение операции сложения двух матриц

Вход: Матрица $A = (a_{ij})$ размерности $n \times m$, матрица $B = (b_{ij})$ размерности $n \times m$.

Выход: Матрица $C = (c_{ij})$ размерности $n \times m$, полученная из суммы $A + B$.

Шаг 1. Элементы c_{ij} матрицы C находятся как $c_{ij} = a_{ij} + b_{ij}$, где $0 \le i < n$, $0 \le j < m$.

Шаг 2. Вернуть матрицу C в качестве выхода функции.

Оценка сложности данного алгоритма равна $O(n \cdot m)$.

Алгоритм 12 – Построение операции вычитания двух матриц

Вход: Матрица $A = (a_{ij})$ размерности $n \times m$, матрица $B = (b_{ij})$ размерности $n \times m$.

Выход: Матрица $C = (c_{ij})$ размерности $n \times m$, полученная из разности $A - B$.

Шаг 1. Элементы c_{ij} матрицы C находятся как $c_{ij} = a_{ij} - b_{ij}$, где $0 \le i < n$, $0 \le j < m$.

Шаг 2. Вернуть матрицу C в качестве выхода функции.

Оценка сложности данного алгоритма равна $O(n \cdot m)$.

Алгоритм 13 – Построение операции умножения матрицы на число

Вход: Матрица $A = (a_{ij})$ размерности $n \times m$, α – некоторое фиксированное число.

Выход: Матрица $\alpha A = (\alpha a_{ij})$ размерности $n \times m$, полученная из умножения матрицы A На число α.

Шаг 1. Необходимо пройти по всем элементам a_{ij} матрицы A, $0 \le i < n$, $0 \le j < m$. Тогда умножив каждый элемент a_{ij} на число α, то получим матрицу αA.

Шаг 2. Вернуть матрицу αA в качестве выхода функции.

Оценка сложности данного алгоритма равна $O(n \cdot m)$.

Алгоритм 14 – Построение операции произведения двух матриц

Вход: Матрица $A = (a_{ij})$ размерности $n \times l$, матрица $B = (b_{ij})$ размерности

$l \times m$.

Выход: Матрица $C = (c_{ij})$ размерности $n \times m$, полученная из произведения матриц A и B.

Шаг 1. Элементы c_{ij} матрицы C находятся следующим образом:

$$c_{ij} = \sum_{k=0}^{l} a_{ik} \cdot b_{kj}, \text{ где } 0 \leq i < n, 0 \leq j < m.$$

Шаг 2. Вернуть матрицу C в качестве выхода функции.

Оценка сложности данного алгоритма равна $O(n \cdot m \cdot l)$.

Алгоритм 15 – Построение операции транспонирования матрицы

Вход: Матрица $A = (a_{ij})$ размерности $n \times m$.

Выход: Матрица $A^T = (a^T_{ij})$ размерности $n \times m$, полученная путем транспонирования матрицы A.

Шаг 1. Элементы a^T_{ij} матрицы A^T находятся так что: $a^T_{ij} = a_{ji}$, где $0 \leq i < n$, $0 \leq j < m$.

Шаг 2. Вернуть матрицу A^T в качестве выхода функции.

Оценка сложности данного алгоритма равна $O(n \cdot m)$.

Алгоритм 16 – Нахождение обратной матрицы

Вход: Матрица $A = (a_{ij})$ размерности $n \times m$.

Выход: Матрица $A^{-1} = (a^{-1}_{ij})$ размерности $n \times m$, полученная путем транспонирования матрицы A.

Шаг 1. Необходимо найти определитель $|A|$.

Шаг 2. Запустив алгоритм 14, найти матрицу A^T.

Шаг 3. Посчитать матрицу алгебраических дополнений $A^{'}$. Пройдя двумя циклами по $0 \leq i < n$ и $0 \leq j < m$, нужно рассмотреть такие a^T_{ij}, которые не стоят на i-й строке и j-м столбце текущей итерации. Далее считается определитель из рассматриваемых элементов. Значение полученного определителя является элементом $a^{'}_{ij}$ матрицы $A^{'}$.

Шаг 4. Используя алгоритм 12, необходимо умножить матрицу $A^{'}$ на число, посчитанное на шаге 1. Таким образом, получим матрицу $A^{-1} = (a^{-1}_{ij})$.

Шаг 5. Вернуть матрицу A^{-1} в качестве выхода функции.

Оценка сложности данного алгоритма равна $O(n \cdot m \cdot k)$, где k – количество итераций при нахождении миноров.

3.4 Коды программ, реализующей рассмотренные алгоритмы

```
import numpy as np


# Проверка операции на идемпотентность
def check_idempotence(set_list, a):
  is_idempotent = True
  for i in range(len(set_list)):
    if a[i][i] != set_list[i]:
      is_idempotent = False
      break
  if is_idempotent:
    print('Binary operation is idempotent')
  else:
    print('Binary operation is not idempotent')


# Проверка операции на коммутативность
def check_commutative(set_list, a):
  is_commutative = True
  n = len(set_list)
  for i in range(n):
    for j in range(n):
      if a[i][j] != a[j][i]:
        is_commutative = False
        break
  if is_commutative:
    print('Binary operation is commutative')
  else:
    print('Binary operation is not commutative')


# Проверка операции на ассоциативность
def check_associative(set_list, a):
  is_associative = True
  n = len(set_list)
  for i in range(n):
```

```
    for j in range(n):
      for k in range(n):
        if a[i][set_list.index(str(a[j][k]))] != \
          a[set_list.index(str(a[i][j]))][k]:
          is_associative = False
          break
  if is_associative:
    print('Binary operation is associative')
  else:
    print('Binary operation is not associative')


# Проверка операции на обратимость
def check_invertibility(set_list, a):
  is_invertible = False
  n = len(set_list)
  for i in range(n):
    cnt1 = 0
    cnt2 = 0
    for j in range(n):
      if a[i][j] == '1':
        cnt1 += 1
      if a[j][i] == '1':
        cnt2 += 1
    if cnt1 == cnt2 == n:
      is_invertible = True
  if is_invertible:
    print('Binary operation is invertible')
  else:
    print('Binary operation is not invertible')


# Проверка операции на дистрибутивность
def check_distributivity(set_list, a):
  print('Enter matrix values')
  b = []
  for i in range(len(set_list)):
    b.append([j for j in input().split()])

  is_distributive = True
  n = len(set_list)
```

```
  for i in range(n):
    for j in range(n):
      for k in range(n):
        if (a[i][set_list.index(b[j][k])] != \
           b[set_list.index(a[i][j])][set_list.index(a[i][k])]) \
           or (a[set_list.index(b[j][k])][i] != \
           b[set_list.index(a[j][i])][set_list.index(a[k][i])]):
          is_invertible = False
          break
  if is_distributive:
    print('Binary operation is distributive')
  else:
    print('Binary operation is not distributive')


# Построение объединение бинарных отношений
def get_union_bin_rel(a, b, n):
  c = []
  for i in range(n):
    tmp = []
    for j in range(len(a[i])):
      if a[i][j] and b[i][j]:
        tmp.append(1)
      else:
        tmp.append(a[i][j] + b[i][j])
    c.append(tmp)
  print('*******************************************')
  print_matrix(c, n)
  go_to_set(c, n)
  print('*******************************************')


# Построение пересечения бинарных отношений
def get_intersection_bin_rel(a, b, n):
  c = []
  for i in range(n):
    tmp = []
    for j in range(len(a[i])):
      tmp.append(a[i][j] * b[i][j])
    c.append(tmp)
  print('*******************************************')
```

```
  print_matrix(c, n)
  go_to_set(c, n)
  print('********************************************')


# Построение дополнения бинарного отношения
def get_addition_bin_rel(a, n):
  c = []
  for i in range(n):
    tmp = []
    for j in range(n):
      tmp.append(1 - a[i][j])
    c.append(tmp)
  print('********************************************')
  print_matrix(c, n)
  go_to_set(c, n)
  print('********************************************')


# Построение обратного бинарного отношения
def get_reverse_bin_rel(a, n):
  c = np.array(a)
  c = c.transpose()
  print('********************************************')
  print_matrix(c, n)
  go_to_set(c, n)
  print('********************************************')


# Построение композиции бинарных отношений
def get_composition_bin_rel(a, b, n):
  c = []
  for i in range(n):
    tmp = []
    for j in range(n):
      sum = 0
      for k in range(n):
        sum += a[i][k] * b[k][j]
        if sum > 1:
          sum = 1
      tmp.append(sum)
```

```
    c.append(tmp)
  print('*******************************************')
  print_matrix(c, n)
  go_to_set(c, n)
  print('*******************************************')

# Выполнение операции сложения матриц
def get_addition_operation(a, n):
  print('Enter values of another matrix')
  b = []
  for i in range(n):
    tmp = input().split()
    for j in range(len(tmp)):
      try:
        tmp[j] = float(tmp[j])
      except:
        k = 0
      else:
        tmp[j] = float(tmp[j])
    b.append(tmp)
  c = []
  for i in range(n):
    tmp = []
    for j in range(len(a[i])):
      tmp.append(a[i][j] + b[i][j])
    c.append(tmp)
  return c


# Выполнение операции вычитания матриц
def get_subtraction_operation(a, n):
  print('Enter values of another matrix')
  b = []
  for i in range(n):
    tmp = input().split()
    for j in range(len(tmp)):
      try:
        tmp[j] = float(tmp[j])
      except:
        k = 0
      else:
```

```python
                tmp[j] = float(tmp[j])
            b.append(tmp)
    c = []
    for i in range(n):
        tmp = []
        for j in range(len(a[i])):
            tmp.append(a[i][j] - b[i][j])
        c.append(tmp)
    return c


# Выполнение операции умножение матрицы на число
def get_multiplication_matrix_on_number(a):
    print('Enter number:')
    num = float(input())
    c = []
    for i in range(len(a)):
        tmp = []
        for j in range(len(a[i])):
            tmp.append(a[i][j] * num)
        c.append(tmp)
    return c


# Выполнение операции умножения матриц
def get_multiplication_operation(a):
    print('Enter the number of rows')
    n = int(input())
    print('Enter the number of columns')
    m = int(input())
    print('Enter values of another matrix')
    b = []
    for i in range(n):
        tmp = input().split()
        for j in range(m):
            try:
                tmp[j] = float(tmp[j])
            except:
                k = 0
            else:
                tmp[j] = float(tmp[j])
```

```
    b.append(tmp)
  c = []
  for i in range(len(a)):
    tmp = []
    for j in range(m):
      sum = 0
      for k in range(n):
        sum += a[i][k] * b[k][j]
      tmp.append(sum)
    c.append(tmp)
  return c


# Выполнение операции транспонирования матрицы
def get_transpose_operation(a):
  c = []
  n = len(a)
  m = len(a[0])
  for i in range(m):
    tmp = []
    for j in range(n):
      tmp.append(a[j][i])
    c.append(tmp)
  return c


# Нахождение обратной матрицы
def get_inverse_matrix(a):
  a = np.array(a)
  return np.linalg.inv(a)


# Вывод матрицы
def print_matrix(a, n):
    cnt = 0
    print('Your matrix:')
    for i in a:
        if (cnt < n):
            print(f'{i}' + ' ')
        else:
            print(f'{i}' + '\n')
```

```
            cnt = 0
        cnt += 1


# Вывод бинарного отношения
def print_binary_relation(br, n):
    print('Your binary relation')
    print('{ ', end='')
    for i in range(n):
        if i == n - 1:
            print('(' + str(br[i][0] + 1) + ', ' + str(br[i][1] + 1), end=')')
        else:
            print('(' + str(br[i][0] + 1) + ', ' + str(br[i][1] + 1), end='), ')
    print(' }')


# Построение бинарного отношения по матрице
def go_to_set(a, n):
    s = []
    for i in range(n):
        for j in range(n):
            if a[i][j] == 1:
                s.append((i, j))

    print_binary_relation(s, len(s))


# Проверка свойств (меню)
def check_properties_mode(set_list, a):
    print_matrix(a, len(a))
    print('Press 1 to check idempotence property')
    print('Press 2 to check commutative property')
    print('Press 3 to check associative property')
    print('Press 4 to check invertibility property')
    print('Press 5 to check distributivity property')
    print('Press 6 to exit')
    bl = input()
    if bl == '1':
      check_idempotence(set_list, a)
      check_properties_mode(set_list, a)
    elif bl == '2':
```

```
      check_commutative(set_list, a)
      check_properties_mode(set_list, a)
    elif bl == '3':
      check_associative(set_list, a)
      check_properties_mode(set_list, a)
    elif bl == '4':
      check_invertibility(set_list, a)
      check_properties_mode(set_list, a)
    elif bl == '5':
      check_distributivity(set_list, a)
      check_properties_mode(set_list, a)
    else:
      return choose_mode()


# Проверка свойств бинарных отношений
def check_all_bin_rel_properties(a, b=[]):
  if b == []:
    print_matrix(a, len(a))
    go_to_set(a, len(a))
    print('Choose operation:')
    print('Press 1 to get union of binary relation')
    print('Press 2 to get intersection of binary relation')
    print('Press 3 to get addition binary relation')
    print('Press 4 to get reverse binary relation')
    print('Press 5 to get composition of binary relation')
    bl = input()
    if bl == '1':
      get_union_bin_rel(a, a, len(a))
      check_all_bin_rel_properties(a)
    elif bl == '2':
      get_intersection_bin_rel(a, a, len(a))
      check_all_bin_rel_properties(a)
    elif bl == '3':
      get_addition_bin_rel(a, len(a))
      check_all_bin_rel_properties(a)
    elif bl == '4':
      get_reverse_bin_rel(a, len(a))
      check_all_bin_rel_properties(a)
    elif bl == '5':
      get_composition_bin_rel(a, a, len(a))
```

```
    check_all_bin_rel_properties(a)
  else:
    print('Exit...')
else:
  print('Your binary relations matrix:')
  print('First:')
  print_matrix(a, len(a))
  go_to_set(a, len(a))
  print('Second:')
  print_matrix(b, len(b))
  go_to_set(b, len(b))
  print('Choose operation:')
  print('Press 1 to get union of binary relations')
  print('Press 2 to get intersection of binary relations')
  print('Press 3 to get addition binary relations')
  print('Press 4 to get reverse binary relations')
  print('Press 5 to get composition of binary relations')
  bl = input()
  if bl == '1':
    get_union_bin_rel(a, b, len(a))
    check_all_bin_rel_properties(a, b)
  elif bl == '2':
    get_intersection_bin_rel(a, b, len(a))
    check_all_bin_rel_properties(a, b)
  elif bl == '3':
    print('First:', end='')
    get_addition_bin_rel(a, len(a))
    print('Second:', end='')
    get_addition_bin_rel(b, len(b))
    check_all_bin_rel_properties(a, b)
  elif bl == '4':
    print('First:', end='')
    get_reverse_bin_rel(a, len(a))
    print('Second:', end='')
    get_reverse_bin_rel(b, len(b))
    check_all_bin_rel_properties(a, b)
  elif bl == '5':
    get_composition_bin_rel(a, b, len(a))
    check_all_bin_rel_properties(a, b)
  else:
    print('Exit...')
```

```python
# Построение бинарных отношений
def construction_of_binary_relation():
    print('Enter number of binary relation elements')
    n = int(input())
    print('Enter values of binary relation matrix')
    a = []
    for i in range(n):
      tmp = input().split()
      for j in range(len(tmp)):
        tmp[j] = int(tmp[j])
      a.append(tmp)
    print('Do you want to enter other binary relation? (0 - no, 1 - yes):')
    bl = input()
    if bl == '1':
      print('Enter values of binary relation matrix')
      b = []
      for i in range(n):
        tmp = input().split()
        for j in range(len(tmp)):
          tmp[j] = int(tmp[j])
        b.append(tmp)
      check_all_bin_rel_properties(a, b)
    elif bl == '0':
      check_all_bin_rel_properties(a)

    return choose_mode()


# Построение матриц
def construction_of_matrix(a):
  print_matrix(a, len(a))
  n = len(a)
  print('Choose operation:')
  print('Press 1 to add another matrix')
  print('Press 2 to subtract another matrix')
  print('Press 3 to multiply this matrix on number')
  print('Press 4 to multiply on another matrix')
  print('Press 5 to transpose this matrix')
  print('Press 6 to find inverse matrix')
```

```
  bl = input()
  if bl == '1':
    a = get_addition_operation(a, n)
    construction_of_matrix(a)
  elif bl == '2':
    a = get_subtraction_operation(a, n)
    construction_of_matrix(a)
  elif bl == '3':
    a = get_multiplication_matrix_on_number(a)
    construction_of_matrix(a)
  elif bl == '4':
    a = get_multiplication_operation(a)
    construction_of_matrix(a)
  elif bl == '5':
    a = get_transpose_operation(a)
    construction_of_matrix(a)
  elif bl == '6':
    a = get_inverse_matrix(a)
    construction_of_matrix(a)
  else:
    return choose_mode()


# Главное меню
def choose_mode():
    print('Choose mode:')
    print('Press 1 to check properties')
    print('Press 2 to execute operations for binary relations')
    print('Press 3 to execute operations for matrix')
    print('Press 4 to exit')
    bl = input()
    if bl == '1':
        print('Enter numbers of set:')
        s = input()
        set_list = [i for i in s.split(' ')]
        print('Enter matrix values')
        a = []
        for i in range(len(set_list)):
          a.append([j for j in input().split()])
        check_properties_mode(set_list, a)
    elif bl == '2':
```

```
        construction_of_binary_relation()
    elif bl == '3':
        a = []
        print('Enter the number of rows')
        n = int(input())
        print('Enter the number of columns')
        m = int(input())
        print('Enter the matrix values')
        for i in range(n):
          tmp = input().split()
          for j in range(m):
              try:
                tmp[j] = float(tmp[j])
              except:
                k = 0
              else:
                tmp[j] = float(tmp[j])
          a.append(tmp)
        construction_of_matrix(a)
    elif bl == '4':
        return
    else:
        print('Incorrect output')
        return choose_mode()


choose_mode()
```

3.5 Результаты тестирования программ

Задание 1. С помощью теста Лайта исследуйте на ассоциативность операции умножения, заданные следующими таблицами Кэли:

·	a	b	c	d
a	a	b	a	b
b	a	b	a	b
c	a	b	c	d
d	a	b	c	d

Чтобы проверить операцию «·» на ассоциативность, необходимо рассмотреть следующие выражения:

$$
\begin{gathered}
a \cdot (a \cdot a) = (a \cdot a) \cdot a; \\
a \cdot (a \cdot b) = (a \cdot a) \cdot b; \\
a \cdot (a \cdot c) = (a \cdot a) \cdot c; \\
\ldots \\
d \cdot (d \cdot c) = (d \cdot d) \cdot c; \\
d \cdot (d \cdot d) = (d \cdot d) \cdot d.
\end{gathered}
$$

Рассмотрим первое выражение $a \cdot (a \cdot a) = (a \cdot a) \cdot a$. Левая часть согласно таблице Кэли будет равна: $a \cdot (a \cdot a) = a \cdot a = a$, тогда правая часть – $(a \cdot a) \cdot a = a \cdot a = a$. Данное выражение является равенством. Аналогично рассматриваются остальные выражения.

Допустим, если рассмотреть, например, выражение $a \cdot (b \cdot c) = (a \cdot b) \cdot c$. Левая часть по таблице Кэли будет равна: $a \cdot (b \cdot c) = a \cdot a = a$, аналогично рассматривается правая часть: $(a \cdot b) \cdot c = b \cdot c = a$. Данное выражение также является равенством.

На рисунке 1 показана проверка операции «·» на свойство ассоциативности, согласно применению алгоритма 3.

```
Choose mode:
Press 1 to check properties
Press 2 to execute operations for binary relations
Press 3 to execute operations for matrix
Press 4 to exit
1
Enter numbers of set:
a b c d
Enter matrix values
a b a b
a b a b
a b c d
a b c d
Your matrix:
['a', 'b', 'a', 'b']
['a', 'b', 'a', 'b']
['a', 'b', 'c', 'd']
['a', 'b', 'c', 'd']
Press 1 to check idempotence property
Press 2 to check commutative property
Press 3 to check associative property
Press 4 to check invertibility property
Press 5 to check distributivity property
Press 6 to exit
3
Binary operation is associative
Your matrix:
['a', 'b', 'a', 'b']
['a', 'b', 'a', 'b']
['a', 'b', 'c', 'd']
['a', 'b', 'c', 'd']
Press 1 to check idempotence property
Press 2 to check commutative property
Press 3 to check associative property
Press 4 to check invertibility property
Press 5 to check distributivity property
Press 6 to exit
```

Рисунок 1 – Проверка операции «·» на ассоциативность

Задание 2. Найти матрицу $A^2 + (10 - \lambda/2) \cdot A + \lambda/2 \cdot E$, где E – единичная матрица второго порядка, A = $\begin{pmatrix} 1 & -2 \\ -3 & \lambda \end{pmatrix}$, $\lambda = 6$.

На рисунках 2-5 показано выполнение задания 2. На рисунке 2 изображено нахождение матрицы (1-го слагаемого) A^2, где применяется алгоритм 14.

```
Choose mode:
Press 1 to check properties
Press 2 to execute operations for binary relations
Press 3 to execute operations for matrix
Press 4 to exit
3
Enter the number of rows
2
Enter the number of columns
2
Enter the matrix values
1 -2
-3 6
Your matrix:
[1.0, -2.0]
[-3.0, 6.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
4
Enter the number of rows
2
Enter the number of columns
2
Enter values of another matrix
1 -2
-3 6
Your matrix:
[7.0, -14.0]
[-21.0, 42.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
```

Рисунок 2 – Нахождение матрицы A^2

Проверка:

$$A^2 = \begin{pmatrix} 1 & -2 \\ -3 & 6 \end{pmatrix} \cdot \begin{pmatrix} 1 & -2 \\ -3 & 6 \end{pmatrix} = \begin{pmatrix} 1 \cdot 1 + (-2) \cdot (-3) & 1 \cdot (-2) + (-2) \cdot 6 \\ (-3) \cdot 1 + 6 \cdot (-3) & (-3) \cdot (-2) + 6 \cdot 6 \end{pmatrix} = \begin{pmatrix} 7 & -14 \\ -21 & 42 \end{pmatrix}.$$

На рисунке 3 изображено нахождение матрицы (2-го слагаемого) $(10 - \lambda/2) \cdot A = 7 \cdot A$, где применяется алгоритм 13.

```
Press 6 to find inverse matrix
0
Choose mode:
Press 1 to check properties
Press 2 to execute operations for binary relations
Press 3 to execute operations for matrix
Press 4 to exit
3
Enter the number of rows
2
Enter the number of columns
2
Enter the matrix values
1 -2
-3 6
Your matrix:
[1.0, -2.0]
[-3.0, 6.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
3
Enter number:
7
Your matrix:
[7.0, -14.0]
[-21.0, 42.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
```

Рисунок 3 – Нахождение матрицы $(10 - \lambda/2) \cdot A$

Проверка:

$$7 \cdot A = 7 \cdot \begin{pmatrix} 1 & -2 \\ -3 & 6 \end{pmatrix} = \begin{pmatrix} 7 & -14 \\ -21 & 42 \end{pmatrix}.$$

На рисунке 4 показано нахождение матрицы (3-го слагаемого) $\lambda/2 \cdot E = 3 \cdot E$, где применяется алгоритм 13.

```
Press 6 to find inverse matrix
0
Choose mode:
Press 1 to check properties
Press 2 to execute operations for binary relations
Press 3 to execute operations for matrix
Press 4 to exit
3
Enter the number of rows
2
Enter the number of columns
2
Enter the matrix values
1 0
0 1
Your matrix:
[1.0, 0.0]
[0.0, 1.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
3
Enter number:
3
Your matrix:
[3.0, 0.0]
[0.0, 3.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
```

Рисунок 4 – Нахождение матрицы $\lambda/2 \cdot E$

Проверка:

$$3 \cdot E = 3 \cdot \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 3 & 0 \\ 0 & 3 \end{pmatrix}.$$

Теперь посчитаем сумму 3-х слагаемых:

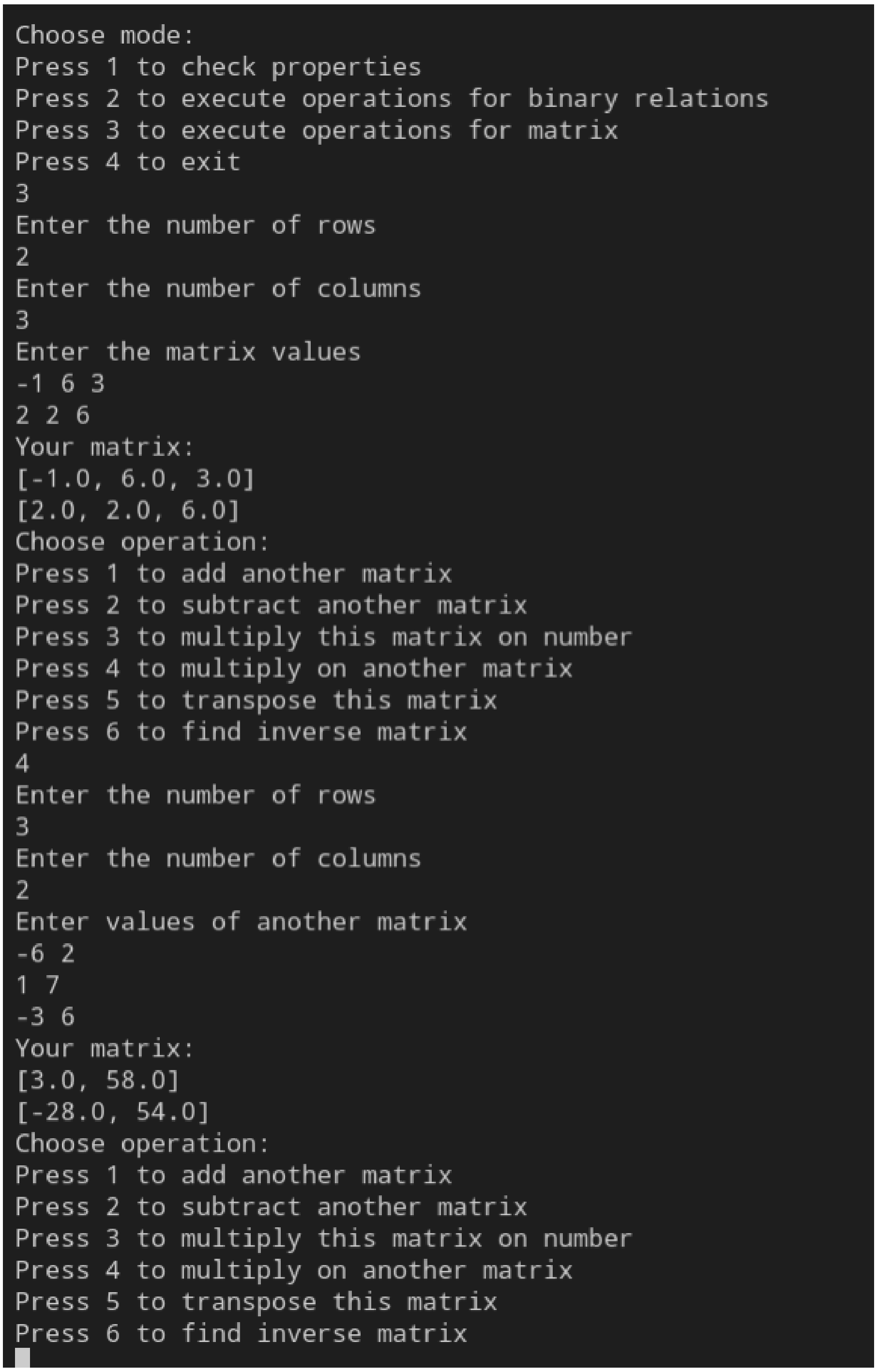
$$\begin{pmatrix} 7 & -14 \\ -21 & 42 \end{pmatrix} + \begin{pmatrix} 7 & -14 \\ -21 & 42 \end{pmatrix} + \begin{pmatrix} 3 & 0 \\ 0 & 3 \end{pmatrix} = \begin{pmatrix} 7+7+3 & -14-14+0 \\ -21-21+0 & 42+42+3 \end{pmatrix} = \begin{pmatrix} 17 & -28 \\ -42 & 87 \end{pmatrix}.$$

Как видно из рисунка 5, программа работает корректно.

```
Your matrix:
[7.0, -14.0]
[-21.0, 42.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
1
Enter values of another matrix
7 -14
-21 42
Your matrix:
[14.0, -28.0]
[-42.0, 84.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
1
Enter values of another matrix
3 0
0 3
Your matrix:
[17.0, -28.0]
[-42.0, 87.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
```

Рисунок 5 – Результат всего выражения

Задание 3. Вычислить произведение $A \cdot B$ матриц $A = \begin{pmatrix} -1 & \lambda & 3 \\ \lambda/3 & 2 & 8-\lambda/3 \end{pmatrix}$,
$B = \begin{pmatrix} -\lambda & 2 \\ 1 & 10-\lambda/2 \\ -3 & \lambda \end{pmatrix}$, где $\lambda = 6$.



```
Choose mode:
Press 1 to check properties
Press 2 to execute operations for binary relations
Press 3 to execute operations for matrix
Press 4 to exit
3
Enter the number of rows
2
Enter the number of columns
3
Enter the matrix values
-1 6 3
2 2 6
Your matrix:
[-1.0, 6.0, 3.0]
[2.0, 2.0, 6.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
4
Enter the number of rows
3
Enter the number of columns
2
Enter values of another matrix
-6 2
1 7
-3 6
Your matrix:
[3.0, 58.0]
[-28.0, 54.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
```

Рисунок 6 – Произведение двух матриц

Проверка:

$$A \cdot B = \begin{pmatrix} -1 & 6 & 3 \\ 2 & 2 & 6 \end{pmatrix} = \begin{pmatrix} -6 & 2 \\ 1 & 7 \\ -3 & 6 \end{pmatrix} =$$

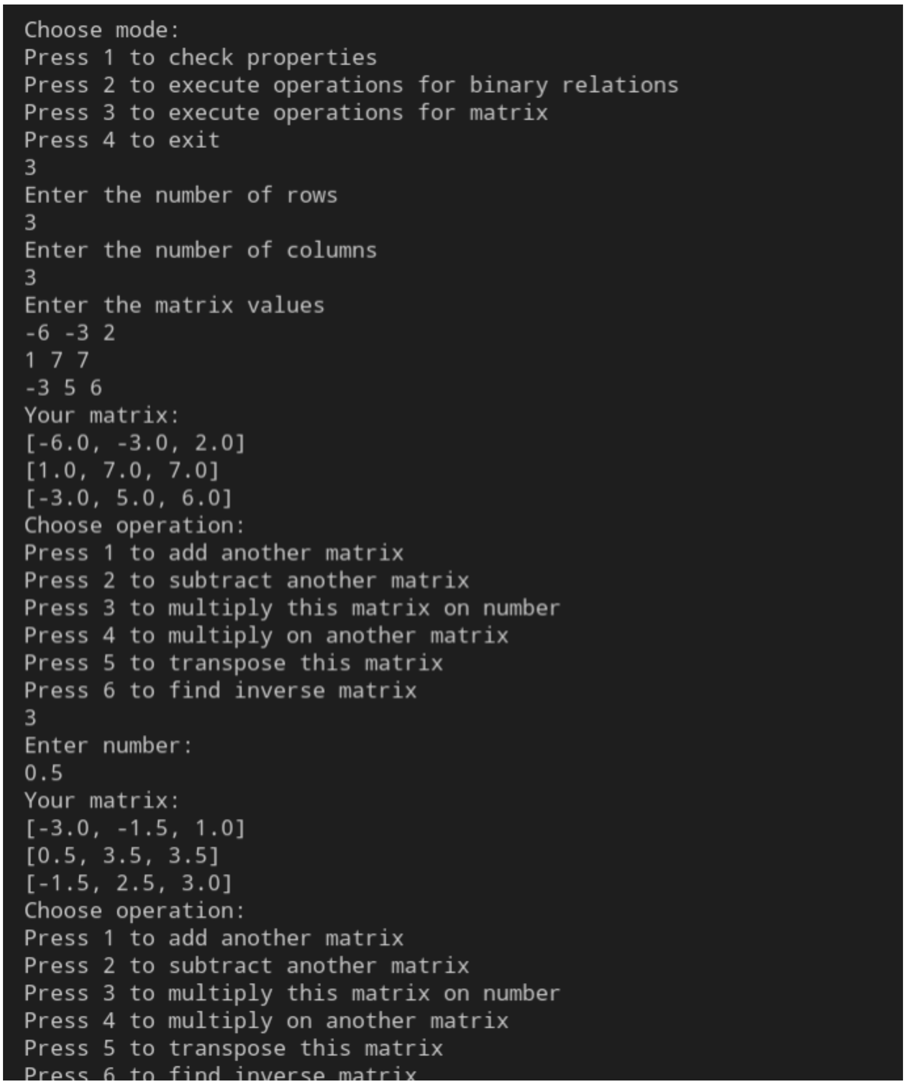
$$\begin{pmatrix} (-1)\cdot(-6) + 6\cdot 1 + 3\cdot(-3) & (-1)\cdot 2 + 6\cdot 7 + 3\cdot 6 \\ 2\cdot(-6) + 2\cdot 1 + 6\cdot(-3) & 2\cdot 2 + 2\cdot 7 + 6\cdot 6 \end{pmatrix} = \begin{pmatrix} 3 & 58 \\ -28 & 54 \end{pmatrix}.$$

Как видно из рисунка 6, программа работает корректно.

Задание 4. Решить матричное уравнение $2 \cdot X + 6 \cdot A = B$ для матриц $A = \begin{pmatrix} -1 & \lambda & 3 \\ -\lambda & 4 & -1 \\ \lambda/3 & 2 & 8-\lambda/3 \end{pmatrix}$, $B = \begin{pmatrix} -\lambda & -3 & 2 \\ 1 & \lambda+1 & 10-\lambda/2 \\ -3 & 5 & \lambda \end{pmatrix}$, где λ = 6.

Проведем следующие преобразования данного уравнения: $2 \cdot X + 6 \cdot A = B \to 2 \cdot X = B - 6 \cdot A \to X = B \cdot \frac{1}{2} - 3 \cdot A$.

На рисунке 7 показана матрица $B_1 = B \cdot \frac{1}{2}$, где применяется алгоритм 13.



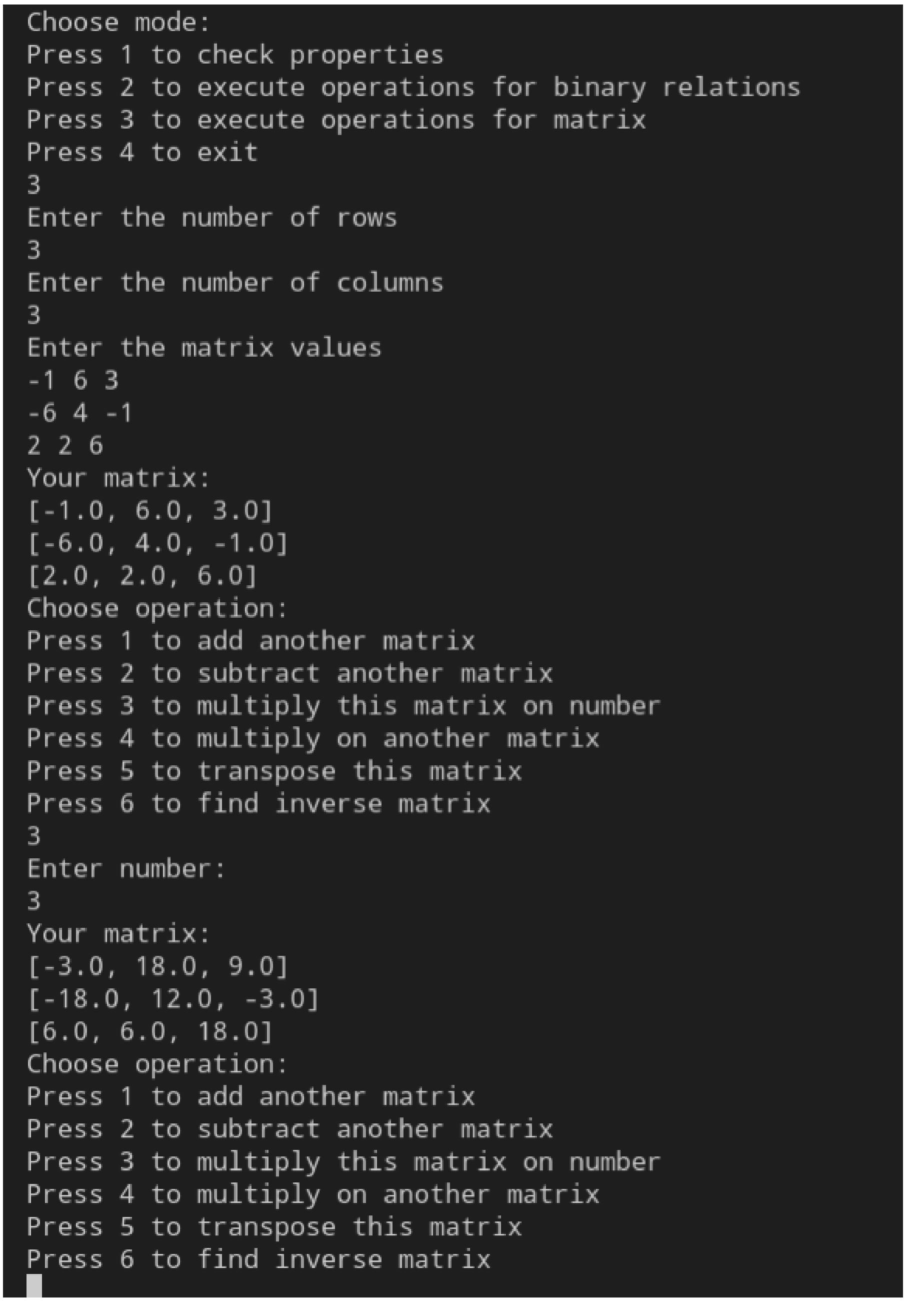
```
Choose mode:
Press 1 to check properties
Press 2 to execute operations for binary relations
Press 3 to execute operations for matrix
Press 4 to exit
3
Enter the number of rows
3
Enter the number of columns
3
Enter the matrix values
-6 -3 2
1 7 7
-3 5 6
Your matrix:
[-6.0, -3.0, 2.0]
[1.0, 7.0, 7.0]
[-3.0, 5.0, 6.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
3
Enter number:
0.5
Your matrix:
[-3.0, -1.5, 1.0]
[0.5, 3.5, 3.5]
[-1.5, 2.5, 3.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
```

Рисунок 7 – Нахождение матрицы B_1

Проверка:

$$B_1 = B \cdot \tfrac{1}{2} = \begin{pmatrix} -6 & -3 & 2 \\ 1 & 7 & 7 \\ -3 & 5 & 6 \end{pmatrix} \cdot \tfrac{1}{2} = \begin{pmatrix} -3 & -1.5 & 1 \\ 0.5 & 3.5 & 3.5 \\ -1.5 & 2.5 & 3 \end{pmatrix}.$$

На рисунке 8 показана матрица $A_1 = 3 \cdot A$, где применяется алгоритм 13.



```
Choose mode:
Press 1 to check properties
Press 2 to execute operations for binary relations
Press 3 to execute operations for matrix
Press 4 to exit
3
Enter the number of rows
3
Enter the number of columns
3
Enter the matrix values
-1 6 3
-6 4 -1
2 2 6
Your matrix:
[-1.0, 6.0, 3.0]
[-6.0, 4.0, -1.0]
[2.0, 2.0, 6.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
3
Enter number:
3
Your matrix:
[-3.0, 18.0, 9.0]
[-18.0, 12.0, -3.0]
[6.0, 6.0, 18.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
```

Рисунок 8 – Нахождение матрицы A_1

Проверка:

$$3 \cdot A = 3 \cdot \begin{pmatrix} -1 & 6 & 3 \\ -6 & 4 & -1 \\ 2 & 2 & 6 \end{pmatrix} = \begin{pmatrix} -3 & 18 & 9 \\ -18 & 12 & -3 \\ 6 & 6 & 18 \end{pmatrix}.$$

На рисунке 9 показано нахождение матрицы $X = B_1 - A_1$, где применяется алгоритм 12.

```
Choose mode:
Press 1 to check properties
Press 2 to execute operations for binary relations
Press 3 to execute operations for matrix
Press 4 to exit
3
Enter the number of rows
3
Enter the number of columns
3
Enter the matrix values
-3 -1.5 1
0.5 3.5 3.5
-1.5 2.5 3
Your matrix:
[-3.0, -1.5, 1.0]
[0.5, 3.5, 3.5]
[-1.5, 2.5, 3.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
2
Enter values of another matrix
-3 18 9
-18 12 -3
6 6 18
Your matrix:
[0.0, -19.5, -8.0]
[18.5, -8.5, 6.5]
[-7.5, -3.5, -15.0]
Choose operation:
Press 1 to add another matrix
Press 2 to subtract another matrix
Press 3 to multiply this matrix on number
Press 4 to multiply on another matrix
Press 5 to transpose this matrix
Press 6 to find inverse matrix
```

Рисунок 9 – Нахождение матрицы X

Проверка:

$$X = B_1 - A_1 = \begin{pmatrix} -3 & -1.5 & 1 \\ 0.5 & 3.5 & 3.5 \\ -1.5 & 2.5 & 3 \end{pmatrix} - \begin{pmatrix} -3 & 18 & 9 \\ -18 & 12 & -3 \\ 6 & 6 & 18 \end{pmatrix} =$$

$$\begin{pmatrix} -3+3 & -1.5-18 & 1-9 \\ 0.5+18 & 3.5-12 & 3.5+3 \\ -1.5-6 & 2.5-6 & 3-18 \end{pmatrix} = \begin{pmatrix} 0 & 19.5 & -8 \\ 18.5 & -8.5 & 6.5 \\ -7.5 & -3.5 & -15 \end{pmatrix}.$$

ЗАКЛЮЧЕНИЕ

В результате лабораторной работы были рассмотрены теоретические сведения об алгебраической операции и классификации свойств операций, основные операции над бинарными отношениями, а также основные операции над матрицами. Опираясь на изложенную выше теорию, были разработаны алгоритмы проверки свойств операций: ассоциативность, коммутативность, идемпотентность, обратимость, дистрибутивность, алгоритмы выполнения операции над бинарными отношениями, а также алгоритмы выполнения операций над матрицами. Была произведена оценка сложности каждого из построенных алгоритмов. Была реализована программа, написанная на языке Python с использованием библиотеки Numpy для преобразования матриц.