Министерство науки и высшего образования Российской Федерации
Калужский филиал
федерального государственного бюджетного
образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУК «Информатика и управление»

КАФЕДРА ИУК4 «Программное обеспечение ЭВМ, информационные технологии»

ЛАБОРАТОРНАЯ РАБОТА №1

«Основные операции над множествами»

ДИСЦИПЛИНА: «Дискретная математика»

Выполнил: студент гр. ИУК4-31Б _____ (Суриков Н. С.)
(подпись) (Ф.И.О.)

Проверил: _____ (Никитенко У. В.)
(подпись) (Ф.И.О.)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:
- Оценка:

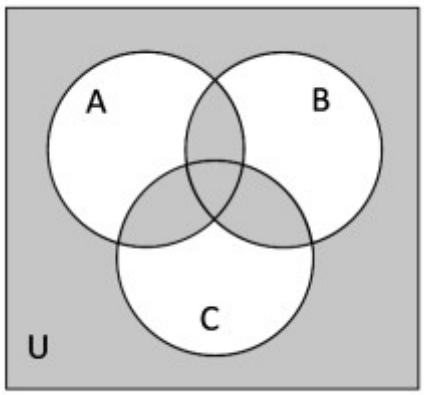
Цель: изучение способов задания множеств, приобретение практических навыков в выполнении операций над множествами, визуализация результатов.

Задачи:

1. Разработать приложение визуализации операций над множествами

Вариант 21

Задание: *Для конечных множеств, по заданному выражению строится диаграмма Эйлера-Венна, заштрихованная (выделенная определенным цветом) часть диаграммы должна соответствовать выражению.*
В качестве тестового примера использовать Задание 1 индивидуального варианта ЛР_1(I).



№ 21

Этапы выполнения:

Этап 1 — Выбор средств для решения задачи:

Для решения задачи был выбран язык Python и библиотека matplotlib_venn, так как синтаксис языка достаточно прост, а библиотека предоставляет готовые решения для быстрой разработки.

Установка: `pip install  matplotlib_venn`

Этап 2 — Формирование формулы:

Программа способна формировать диаграммы по введённой формуле, при условии что эта формула написана с использованием операторов и синтаксиса языка Python.

```
┌──(~/Semestr_3)
└─(20:24:09 on main ✹ ★)──> python -u "/home/syricoff/Semestr_3/ДМ/ЛР1/main.py"
Введите логическую формулу (используйте A, B, C):
```

Этап 3 — Обработка формулы:

Для обработки была написана функция обработчик которая генерирует регионы, которые должны быть закрашены.

```python
def list_regions(logical_formula) -> Generator[str, Any, None]:
    """
    Генерирует все регионы, удовлетворяющие заданной логической формуле.

    :param logical_formula:
    Логическая формула в виде строки, использующая переменные A, B, C.
    :yield: Регион в виде строки (например, "010").
    """
    for values in product([False, True], repeat=3):
        try:
            if eval(source/logical_formula, globals={}, locals=dict(zip(iter1/"ABC", iter2/values))):
                yield "".join(iterable/map(func/str, iter1/map(func/int, iter1/values)))
        except Exception as e:
            print(f"Ошибка при оценке формулы: {e}")
```

Этап 4 — Вывод диаграммы:

Для вывода диаграммы используются средства библиотеки `matplotlib_venn`, создание происходит в основной функции plot_diagram.

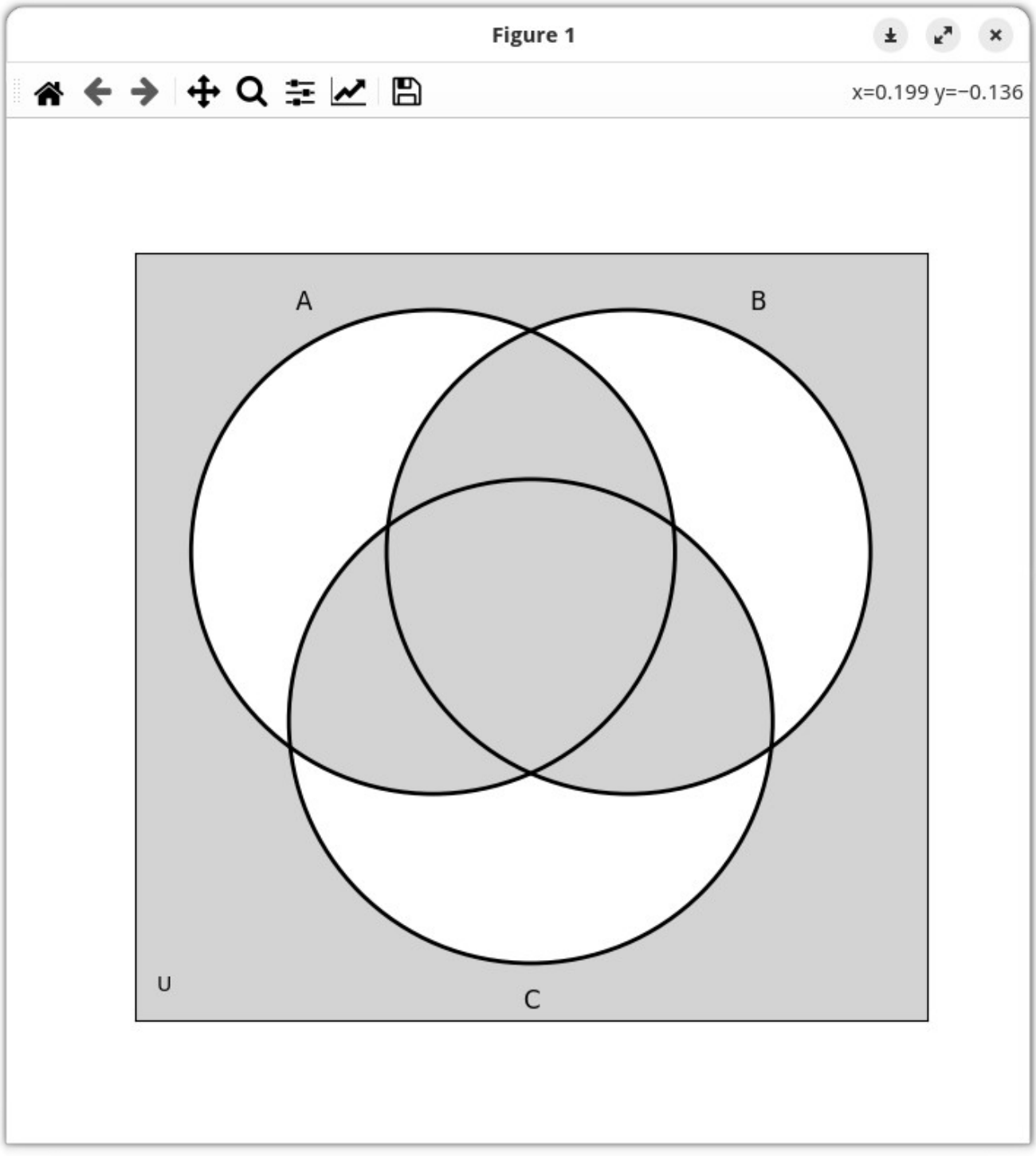
```python
def plot_diagram(logical_formula) -> None:
    """
    Создает диаграмму Венна по заданной логической формуле.

    :param logical_formula:
    Логическая формула в виде строки, использующая переменные A, B, C.
    """
    figure: Figure, ax: Any = plt.subplots(figsize=(7, 7))

    venn_diagram: VennDiagram = venn3(
        alpha=1.0,
        subsets=(1,) * 7,
        set_colors=("white",) * 3,
        subset_label_formatter=lambda x: "",
    )

    venn3_circles(subsets=[1] * 7, linestyle="solid")

    regions_to_color = list(iterable/list_regions(logical_formula=logical_formula))

    for region in regions_to_color:
        if region == "000":
            ax.set_facecolor("lightgrey")
        else:
            patch: Patch = venn_diagram.get_patch_by_id(id=region)
            if patch:
                patch.set_color(c="lightgrey")
            else:
                print(f"Регион {region} не найден")

    plt.annotate(text="U", xy=(-0.660, -0.675))
    plt.axis(arg="on")
    plt.show()
```

Результат работы программы:

Ввод:

```
Введите логическую формулу (используйте A, B, C): (A and B) or (A and C) or (B and C) or not(A or B or C)
```

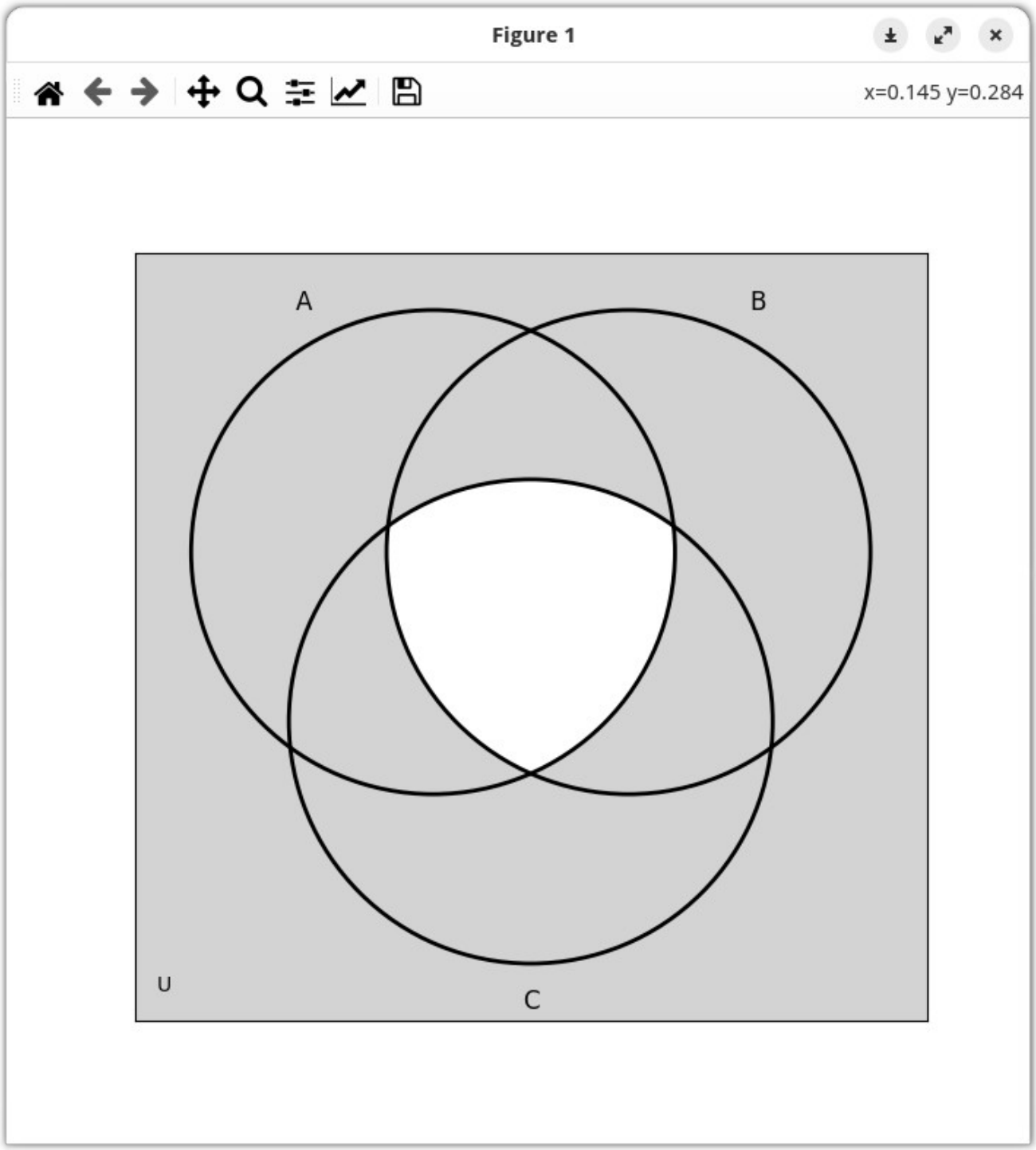
Вывод:



Ввод:

```
Введите логическую формулу (используйте A, B, C): not(A and B and C)
```

Вывод:



Листинг программы:

```
from matplotlib import pyplot as plt
from matplotlib_venn import venn3, venn3_circles
from itertools import product


def get_logical_formula():
    """
    Запрашивает у пользователя ввод логической формулы.

    :return: Логическая формула в виде строки.
    """
    formula = input("Введите логическую формулу (используйте A, B, C): ")
    return formula


def list_regions(logical_formula):
    """
    Генерирует все регионы, удовлетворяющие заданной логической формуле.

    :param logical_formula:
    Логическая формула в виде строки, использующая переменные A, B, C.
    :yield: Регион в виде строки (например, "010").
    """
    for values in product([False, True], repeat=3):
        try:
            if eval(logical_formula, {}, dict(zip("ABC", values))):
                yield "".join(map(str, map(int, values)))
        except Exception as e:
            print(f"Ошибка при оценке формулы: {e}")


def plot_diagram(logical_formula):
    """
    Создает диаграмму Венна по заданной логической формуле.

    :param logical_formula:
    Логическая формула в виде строки, использующая переменные A, B, C.
    """
```

```
    figure, ax = plt.subplots(figsize=(7, 7))

    venn_diagram = venn3(
        alpha=1.0,
        subsets=(1,) * 7,
        set_colors=("white",) * 3,
        subset_label_formatter=lambda x: "",
    )

    venn3_circles(subsets=[1] * 7, linestyle="solid")

    regions_to_color = list(list_regions(logical_formula))

    for region in regions_to_color:
        if region == "000":
            ax.set_facecolor("lightgrey")
        else:
            patch = venn_diagram.get_patch_by_id(region)
            if patch:
                patch.set_color("lightgrey")
            else:
                print(f"Регион {region} не найден")

    plt.annotate("U", (-0.660, -0.675))
    plt.axis("on")
    plt.show()


if __name__ == "__main__":
    logical_formula = get_logical_formula()
    plot_diagram(logical_formula)
```

Вывод: В ходе работы были изучены способы задания множеств, приобретены практические навыки в выполнении операций над множествами.