Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии

Департамент цифровых, робототехнических систем и электроники

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №4**

**дисциплины**

**«Искусственный интеллект в профессиональной сфере»**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Выполнила:  Михеева Елена Александровна  3 курс, группа ИВТ-б-о-22-1,  09.03.01 «Информатика и вычислительная техника», направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | | Проверил:  Воронкин Р.А.-доцент департамента цифровых, роботехнических систем и электроники института перспективной инженерии  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | |  | |

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_ Дата защиты\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ставрополь, 2024 г.

**ТЕМА: ИССЛЕДОВАНИЕ ПОИСКА С ОГРАНИЧЕНИЕМ ГЛУБИНЫ**

**Цель:** приобретение навыков по работе с поиском с ограничением глубины с помощью языка программирования Python версии 3.x

**Порядок выполнения работы:**

**Репозиторий:** [*https://github.com/helendddd/AI\_4.git*](https://github.com/helendddd/AI_4.git)

***Система навигации робота-пылесоса***

В рамках разработки системы навигации для робота-пылесоса была поставлена задача определить, существует ли путь к целевой комнате с учетом ограничения на глубину поиска. Робот передвигался по различным комнатам в доме, но из-за ограниченных ресурсов, таких как заряд батареи и время на уборку, требовалось эффективно выбирать маршрут.

Дано дерево, где каждый узел представлял комнату. Узлы соединялись, если существовала возможность перемещения между ними. Необходимо было выяснить, достигалась ли целевая комната из начальной (корня дерева) без превышения заданного лимита глубины.

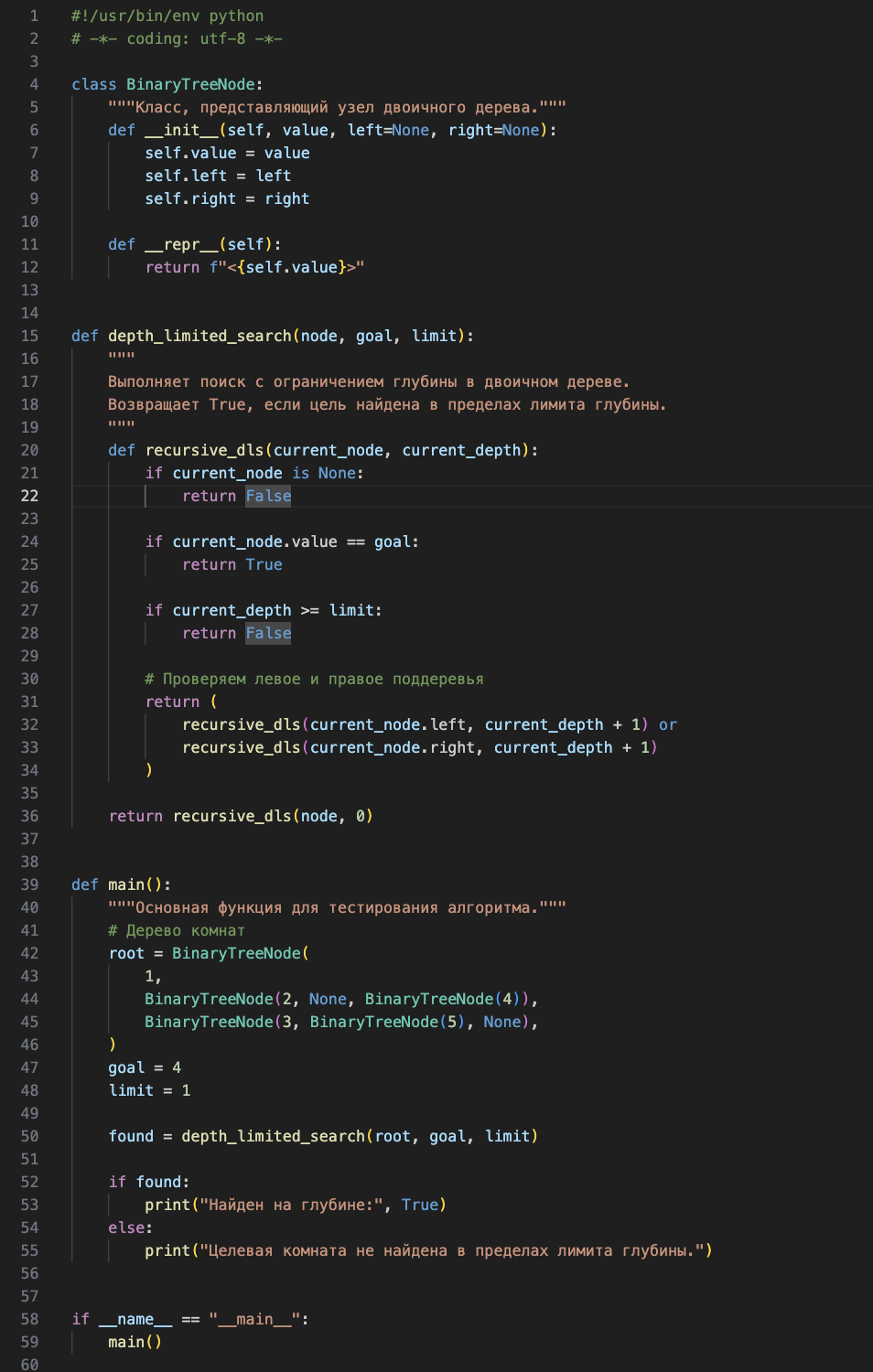


Рисунок 1. Код программы

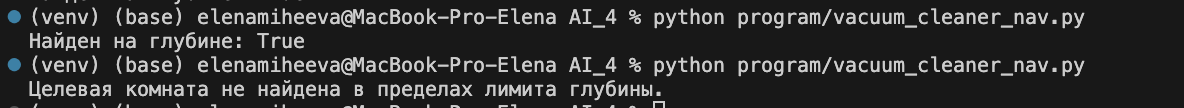


Рисунок 2. Результат работы программы для глубины 2 и 1

***Система управления складом.***

В процессе разработки системы управления складом была поставлена задача найти наименее затратный путь к товару в структуре, подобной двоичному дереву. Каждый узел дерева представлял место хранения, которое вело к левому и правому подразделу.

Для гарантии приемлемого времени поиска требовалось ограничить глубину анализа. Задача заключалась в определении оптимального маршрута к товару при заданном ограничении глубины.



Рисунок 3. Код программы



Рисунок 4. Результат работы программы

***Система автоматического управления инвестициями.***

Была разработана система для автоматического управления инвестициями, в которой дерево решений используется для отображения последовательности инвестиционных решений и их возможных результатов. Цель состояла в нахождении наилучшего исхода (максимальной прибыли) на заданной глубине принятия решений, принимая во внимание ограниченные ресурсы и время, доступное для анализа.

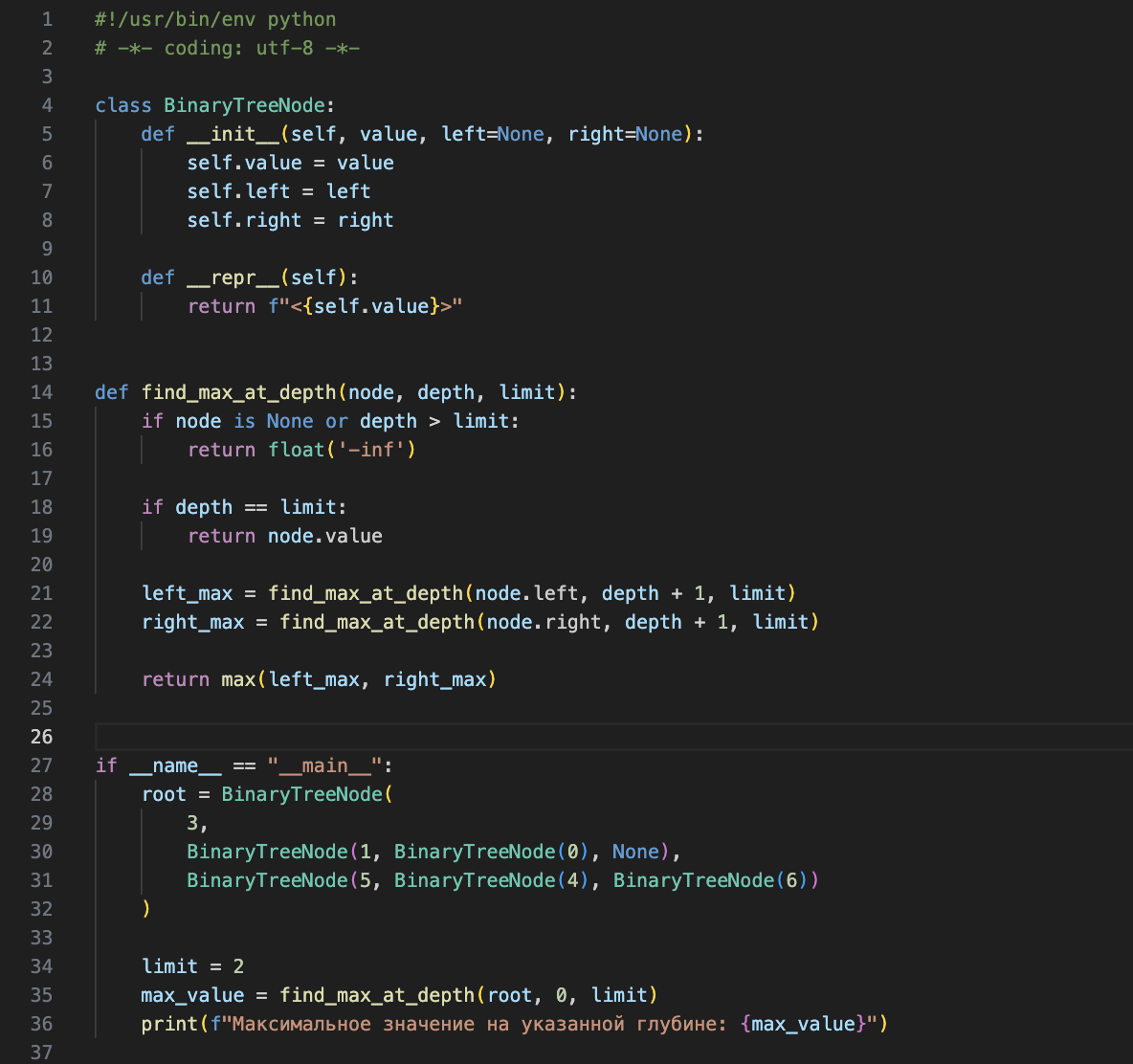


Рисунок 5. Код программы



Рисунок 6. Результат работы программы

***Нахождение минимального расстояния между начальными и конечными пунктами с использованием алгоритма поиска с ограничением глубины.***

Был построен граф из 20 населенных пунктов Италии. Узлы данного графа представляют населённые пункты, а рёбра — дороги, соединяющие их. Вес каждого ребра соответствует расстоянию между этими пунктами. В качестве начального населенного пункта выбран Кунео, а конечного – Верона.

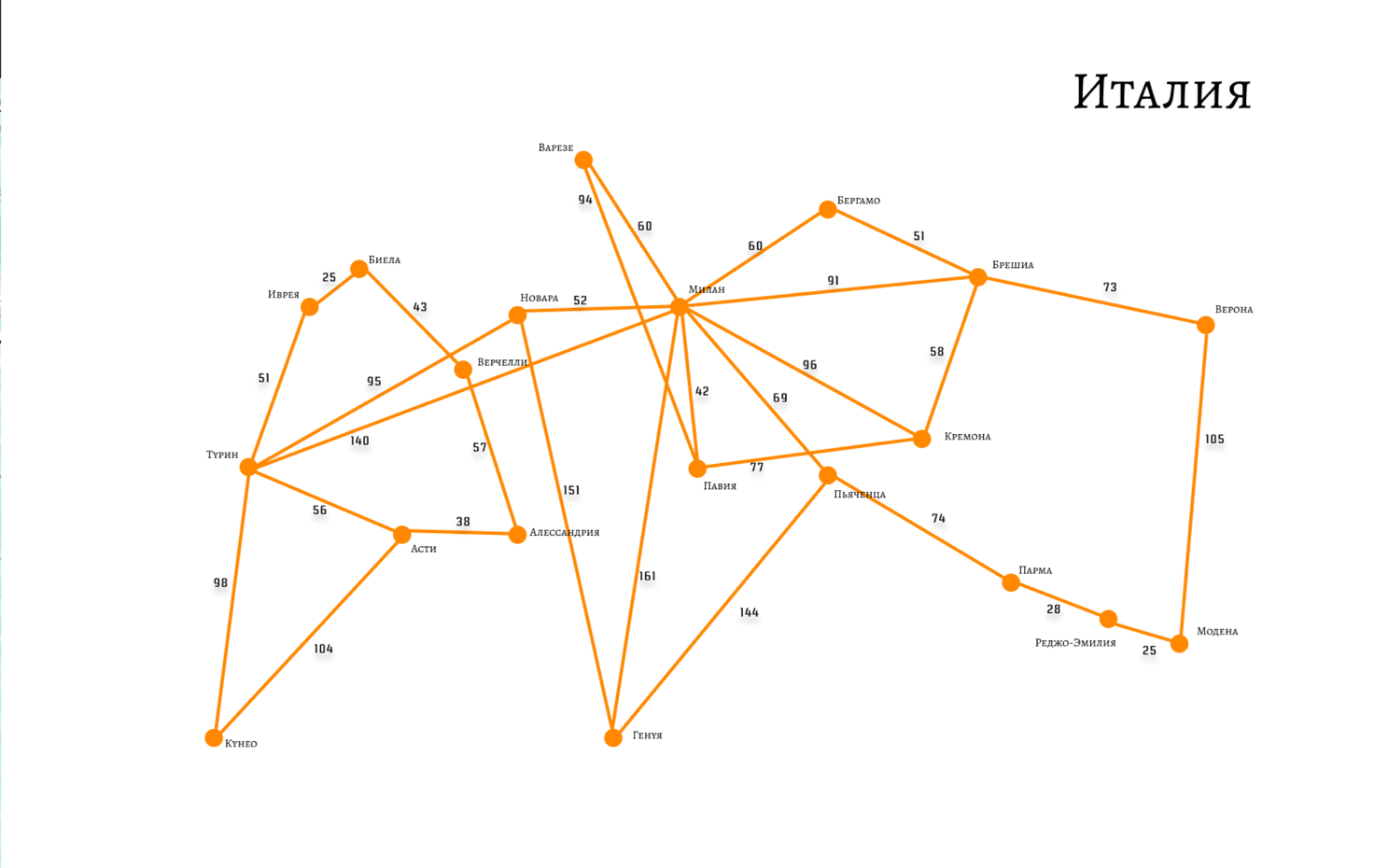


Рисунок 7. Граф из населенных пунктов

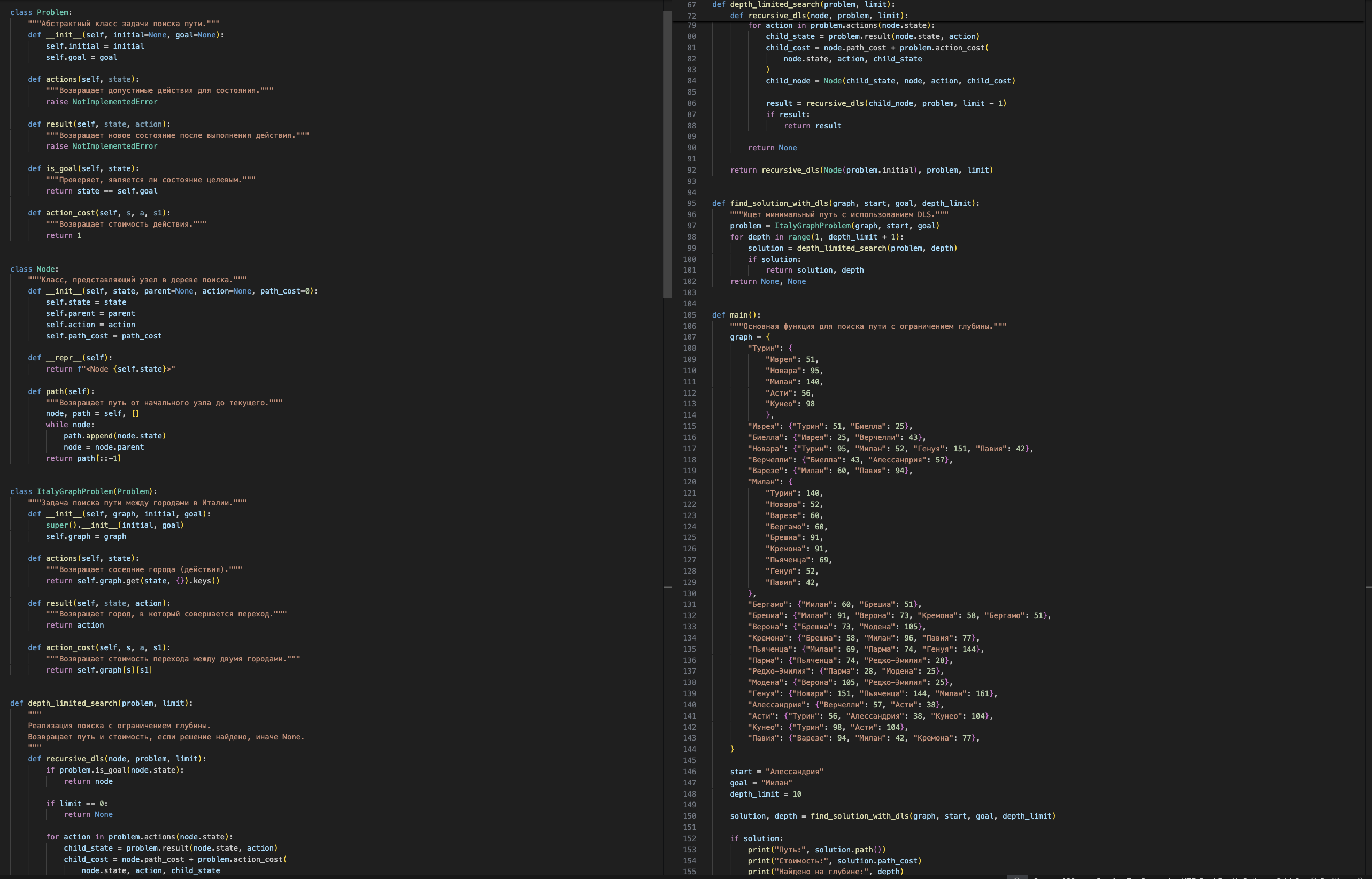
******

Рисунок 8. Код программы для задания

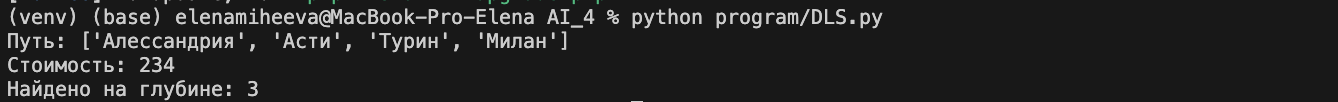


Рисунок 9. Результат работы программы

**Ответы на контрольные вопросы:**

1. Что такое поиск с ограничением глубины, и как он решает проблему бесконечных ветвей?

Поиск с ограничением глубины — это метод, который ограничивает максимальную глубину поиска. Он решает проблему бесконечных ветвей, прерывая их обработку на заранее определённой глубине.

1. Какова основная цель ограничения глубины в данном методе поиска?

Основная цель — предотвратить застревание в бесконечных ветвях и ограничить количество исполнившихся шагов, что экономит время и ресурсы.

1. В чем разница между поиском в глубину и поиском с ограничением глубины?

Поиск в глубину продолжает исследовать дерево без ограничений, в то время как поиск с ограничением глубины останавливается на заданной глубине, избегая чрезмерного углубления.

1. Какую роль играет проверка глубины узла в псевдокоде поиска с ограничением глубины?

Проверка глубины узла обеспечивает ограничение на количество шагов, которое может быть сделано в поиске, предотвращая застревание на бесконечных путях.

1. Почему в случае достижения лимита глубины функция возвращает «обрезание»?

Это необходимо, чтобы остановить дальнейшее расширение узлов на глубине, которая превышает установленное ограничение.

1. В каких случаях поиск с ограничением глубины может не найти решение, даже если оно существует?

Если решение находится на глубине, превышающей установленный лимит, поиск не сможет его найти.

1. Как поиск в ширину и в глубину отличаются при реализации с использованием очереди?

В поиске в ширину используется очередь FIFO (первым пришёл — первым ушёл), а в поиске в глубину — очередь LIFO (последним пришёл — первым ушёл).

1. Почему поиск с ограничением глубины не является оптимальным?

Он не гарантирует нахождение наименьшего пути, так как не проверяет более глубокие решения, которые могут быть более эффективными.

1. Как итеративное углубление улучшает стандартный поиск с ограничением глубины?

Итеративное углубление постепенно увеличивает глубину, улучшая вероятность нахождения решения при минимальных затратах памяти и времени.

1. В каких случаях итеративное углубление становится эффективнее простого поиска в ширину?

Когда глубина решения неизвестна и нужно сбалансировать память и время работы. Итеративное углубление использует меньше памяти, чем поиск в ширину.

1. Какова основная цель использования алгоритма поиска с ограничением глубины?

Основная цель — эффективно искать решение в ограниченном пространстве, избегая избыточных вычислений и предотвращая бесконечные вычисления.

1. Какие параметры принимает функция depth\_limited\_search, и каково их назначение?

Параметры: problem (задача, для которой проводится поиск), node (текущий узел), limit (максимальная глубина). Они определяют, что и как будет искаться.

1. Какое значение по умолчанию имеет параметр limit в функции depth\_limited\_search?

По умолчанию значение параметра limit равно None, что означает отсутствие ограничения по глубине.

1. Что представляет собой переменная frontier, и как она используется в алгоритме?

frontier — это структура данных для хранения узлов, которые нужно обработать. В алгоритме используется очередь для управления поисковыми узлами.

1. Какую структуру данных представляет LIFOQueue, и почему она используется в этом алгоритме?

LIFOQueue представляет собой стек (Last In, First Out), который используется в поиске в глубину, чтобы обрабатывать узлы в обратном порядке их добавления.

1. Каково значение переменной result при инициализации, и что оно означает?

result инициализируется как None и означает, что решение ещё не найдено.

1. Какое условие завершает цикл while в алгоритме поиска?

Цикл завершается, когда frontier становится пустым или найдено решение.

1. Какой узел извлекается с помощью frontier.pop() и почему?

Извлекается последний добавленный узел, так как используется структура данных LIFO.

1. Что происходит, если найден узел, удовлетворяющий условию цели (условие problem.is\_goal(node.state))?

Если узел удовлетворяет условию цели, возвращается результат и поиск завершается.

1. Какую проверку выполняет условие elif len(node) >= limit, и что означает его выполнение?

Проверяется, не превышена ли максимальная глубина. Если да, поиск прекращается для этого узла.

1. Что произойдёт, если текущий узел достигнет ограничения по глубине поиска?

Если узел достигнет ограничения, его дальнейшее расширение не будет происходить, и функция вернёт "обрезание".

1. Какую роль выполняет проверка на циклы elif not is\_cycle(node) в алгоритме?

Проверка на циклы предотвращает застревание в бесконечных циклах, исключая повторное посещение одинаковых узлов.

1. Что происходит с дочерними узлами, полученными с помощью функции expand(problem, node)?

Дочерние узлы добавляются в frontier для дальнейшей обработки.

1. Какое значение возвращается функцией, если целевой узел не был найден?

Возвращается значение None или failure, если решение не найдено.

1. В чем разница между результатами failure и cutoff в контексте данного алгоритма?

failure означает, что решение невозможно, а cutoff — что решение не найдено из-за достижения лимита глубины, но оно может быть найдено на более глубокой глубине.

**Вывод:** былиприобретены навыки по работе с поиском ограничением глубины с помощью языка программирования Python версии 3.x