VI Международная конференция учащихся
«НАУЧНО-ТВОРЧЕСКИЙ ФОРУМ»

«Создание игры для школьников “Обучение программированию на Python”»

Автор работы: Бойков Артём Алексеевич
11“А” МБОУ СОШ №46 г. Калуги
Научный руководитель: Жандарова Лариса Борисовна
учитель физики, МБОУ СОШ №46

Калуга, 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Человечество быстро прогрессирует. Ещё в 1950-ых годах количество компьютеров не превышало 100, а сейчас почти у каждого человека в доме есть свой компьютер, ноутбук, телефон или же планшет. Каждый день открываются всё новые и новые возможности открытий в самых разных областях науки. Чтобы открыть новые технологии, учёные ставят перед собой множество задач. Для этого они проводят неисчислимое количество экспериментов. И вот результаты. На данный момент в мире разрабатывается огромное количество приложений, одной из их категорий как раз и являются игры. С течением времени появлялись новые языки программирования, на которых их можно было писать новые приложения. Наши школьники всяческими методами пытаются облегчить себе школьную жизнь. На протяжении долгих 11 лет обучения всё это большое количество информации просто не усваивается, поэтому у многих обучающихся возникают сложности с пониманием различных предметов. Все люди разные, кто-то быстрее, а кто-то медленнее запоминает и усваивает полученную информацию. Поэтому в интернете набрали популярность школьные приложения, которые предлагают помощь учащимся в обучении материала, трудного для запоминания, и не только. Так потихоньку начали появляться различные приложения и платформы не только в Интернет-ресурсах. Но, к сожалению, даже в таких способах обучения есть недостатки и главный из них это недопонимание. Большинство приложений и платформ, направленных на обучение, могут создать не только недопонимание, но и полное заблуждение в теме. Многие темы, изученные в школе, проверяются с помощью ОГЭ и ЕГЭ, поэтому в них нужно хорошо разбираться. На данный момент для изучение всех этих тем и их дальнейшего осознания, требуется куда больше времени, чем можно себе представить, также становится сложнее понимать задания и находить решения к ним, но быстро можно найти ответ в интернете, просто введя условие задачи и выбрать нужное, из множества предложенных вариантов, но этого может быть недостаточно для осознания сути решения задачи, поэтому часто дети заходят в специальным приложения с конспектами или вовсе используют приложения для пошагового решения задач и дальнейшего разбора.

Иногда дети пользуются книгами с решениями и видео-объяснениями, но они мало в чём помогают, а также усложняют решение, растягивая его очень сильно, когда можно сделать всё коротко и быстро. Всё это наталкивает на мысль, что обычных приложений и книжек для учёбы недостаточно.

Так мы плавно переходим к проблеме моего проекта – трудность обучения. Для решения данной проблемы я решил создать собственной приложение, в котором, чтобы открывалась теория, нужно пройти игровой уровень. В различных источниках, которыми человек пользуется зачастую трудно найти чёткое объяснение для решения какой-либо задачи, но для этого в своём приложении после каждого уровня будет предложено решение данной задачи, а также видео, в которых могут чётко объяснить способ решения задачи.

Главной целью данного проекта было создание собственного приложения-игры с библиотекой задач и теории внутри, доступной для понимания каждому, кто воспользуется моим продуктом, тот сможет найти новые полезные материалы для обучения. Также изучение космоса связано с установлением причинно-следственных связей, происходящих во вселенной. Для этого необходимо развивать логическое мышление на школьном уровне, чему способствует мой проект.

Глава 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 ЧТО ТАКОЕ PYTHON?

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью объектно-ориентированным. Необычной особенностью языка является выделение блоков кода отступами. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Сам же язык известен как интерпретируемый и используется в том числе для написания скриптов. Недостатками языка являются зачастую более низкая скорость работы и более высокое потребление памяти написанных на нём программ.

Стандартная библиотека включает большой набор полезных переносимых функций, начиная с возможностей для работы с текстом и заканчивая средствами для написания сетевых приложений. Дополнительные возможности, такие как математическое моделирование, работа с оборудованием, написание веб-приложений или разработка игр, могут реализовываться посредством обширного количества сторонних библиотек, а также интеграцией библиотек.

Python стал одним из самых популярных языков, он используется в анализе данных, машинном обучении, веб-разработке, а также в других сферах, включая разработку игр. За счёт читабельности, простого синтаксиса и отсутствия необходимости в компиляции язык хорошо подходит для обучения программированию, позволяя концентрироваться на изучении алгоритмов, концептов и парадигм. Отладка же и экспериментирование в значительной степени облегчаются тем фактом, что язык является интерпретируемым. Применяется язык многими крупными компаниями, такими как Google или Facebook.

1.2 ЧТО ТАКОЕ ИГРОВОЙ ДВИЖОК?

Игровой движок — базовое программное обеспечение компьютерной игры. Разделение игры и игрового движка часто расплывчато, и не всегда студии проводят чёткую границу между ними. Но в общем случае термин «игровой движок» применяется для того программного обеспечения, которое пригодно для повторного использования и расширения, и тем самым может быть рассмотрено как основание для разработки множества различных игр без существенных изменений.

Большинство игровых движков разработано и настроено для того, чтобы запустить определённую игру на определённой платформе. И даже наиболее обобщённые много платформенные движки подходят для построения игр определённого жанра, например, шутеров первого лица или гонок. В данном контексте можно более аккуратно сказать, что игровой движок становится не оптимальным при его применении не для той игры или той платформы, для которой разработан. Данный эффект проявляется от того, что программное обеспечение представляет собой набор компромиссов, основанных на тех предположениях, какой должна быть игра. Например, проектирование рендеринга внутри зданий приведёт к тому, что движок, скорее всего, не будет таким же хорошим для открытых пространств. В первом случае движок может использовать BSP-дерево для отрисовки объектов, близких к камере. В то же время для открытых пространств могут использоваться менее точные способы, а также более активно применяются технологии отрисовки с разной степенью детализации, когда более далёкие объекты прорисовываются менее чётко, так как занимают меньшее количество пикселей.

1.3 ЧТО ТАКОЕ UNITY?

Unity — кроссплатформенная среда разработки компьютерных игр, разработанная американской компанией Unity Technologies. Unity позволяет создавать приложения, работающие на более чем 25 различных платформах, включающих персональные компьютеры, игровые консоли, мобильные устройства, интернет-приложения и другие.

Достоинства и недостатки Unity:

Как правило, игровой движок предоставляет множество функциональных возможностей, позволяющих их задействовать в различных играх, в которые входят моделирование физических сред, карты нормалей, динамические тени и многое другое. В отличие от многих игровых движков, у Unity имеется два основных преимущества:
наличие визуальной среды разработки и межплатформенная поддержка. Первый фактор включает не только инструментарий визуального моделирования, но и интегрированную среду, цепочку сборки, что направлено на повышение производительности разработчиков, в частности, этапов создания прототипов и тестирования.
Под межплатформенной поддержкой предоставляется не только места развёртывания, но и наличие инструментария разработки (интегрированная среда может использоваться под Windows.

Третьим преимуществом называется модульная система компонентов Unity, с помощью которой происходит конструирование игровых объектов, когда последние представляют собой комбинируемые пакеты функциональных элементов. В отличие от механизмов наследования, объекты в Unity создаются посредством объединения функциональных блоков, а не помещения в узлы дерева наследования. Такой подход облегчает создание прототипов, что актуально при разработке игр.

В качестве недостатков приводятся ограничение визуального редактора при работе с многокомпонентными схемами, когда в сложных сценах визуальная работа затрудняется. Вторым недостатком называется отсутствие поддержки Unity ссылок на внешние библиотеки, работу с

которыми программистам приходится настраивать самостоятельно, и это также затрудняет командную работу. Ещё один недостаток связан с использованием шаблонов экземпляров. С одной стороны, эта концепция Unity предлагает гибкий подход визуального редактирования объектов, но с другой стороны, редактирование таких шаблонов является сложным. Также, WebGL-версия движка, в силу специфики своей архитектуры имеет ряд нерешённых проблем с производительностью, потреблением памяти и работоспособностью на мобильных устройствах.

Первая версия Unity появилась в 2005 году, когда игровой движок был анонсирован на Worldwide Developers Conference. Изначально Unity предназначался исключительно для компьютеров Mac, а в августе вышло обновление, позволяющее работать под Windows. В следующих версиях постепенно добавлялись новые платформы и развёртывания: межплатформенный веб-плеер в 2006-м, iPhone в 2008-м, Android в 2010-м, и далее на игровых консолях Xbox и Playstation.

Есть возможность создавать приложения для запуска в браузерах с помощью специального подключаемого модуля Unity, а также с помощью реализации технологии WebGL. Ранее была экспериментальная поддержка реализации проектов в рамках модуля Adobe Flash Player, но позже команда разработчиков Unity приняла сложное решение по отказу от этого. В декабре 2009 года Gamasutra назвал Unity одним из самых значительных участников на рынке игровых компаний.

1.4 КАКИЕ ПРЕИМУЩЕСТВА СОЗДАНИЯ ИГР В UNITY?

Редактор Unity имеет простой интерфейс, состоящий из различных окон, благодаря чему можно производить отладку игры прямо в редакторе. Движок использует для написания скриптов C#. Ранее поддерживались также Boo и модификация JavaScript, известная как UnityScript. Расчёты физики производит физический движок PhysX от NVIDIA для 3D физики и Box2D для 2D физики. Графический API — DirectX

Проект в Unity делится на сцены — отдельные файлы, содержащие свои игровые миры со своим набором объектов, сценариев, и настроек. Сцены могут содержать в себе как, собственно, объекты, так и пустые игровые объекты — объекты, которые не имеют модели. Объекты, в свою очередь содержат наборы компонентов, с которыми и взаимодействуют скрипты. Также у объектов есть название, может быть тег и слой, на котором он должен отображаться. Так, у любого объекта на сцене обязательно присутствует компонент Transform — он хранит в себе координаты местоположения, поворота и размеров объекта по всем трём осям.
Также Unity поддерживает физику твёрдых тел и ткани.

В редакторе имеется система наследования объектов; подобъекты будут повторять все изменения позиции, поворота и масштаба родительского объекта. Скрипты в редакторе прикрепляются к объектам в виде отдельных компонентов.

При импорте текстуры в Unity можно сгенерировать alpha-канал, mip-уровни, normal-map, light-map, карту отражений, однако непосредственно на модель текстуру прикрепить нельзя — будет создан материал, которому будет назначен шейдер, и затем материал прикрепится к модели. Редактор Unity поддерживает написание и редактирование шейдеров. Редактор Unity имеет компонент для создания анимации, но также анимацию можно создать предварительно в 3D-редакторе и импортировать вместе с моделью, а затем разбить на файлы.

Unity 3D поддерживает систему Level Of Detail, суть которой заключается в том, что на дальнем расстоянии от игрока высоко детализированные модели заменяются на менее детализированные, и наоборот, а также систему Occlusion culling, суть которой в том, что у объектов, не попадающих в поле зрения камеры, не визуализируется геометрия и коллизия, что снижает нагрузку на центральный процессор и позволяет оптимизировать проект. При компиляции проекта создаётся исполняемый (.exe) файл игры (для Windows), а в отдельной папке — данные игры (включая все игровые уровни и динамически подключаемые библиотеки).

Движок поддерживает множество популярных форматов. Модели, звуки, текстуры, материалы, скрипты можно запаковывать в формат unitypackage и передавать другим разработчикам, или выкладывать в свободный доступ. Этот же формат используется во внутреннем магазине Unity Asset Store, в котором разработчики могут бесплатно и за деньги выкладывать в общий доступ различные элементы, нужные при создании игр. Чтобы использовать Unity Asset Store, необходимо иметь аккаунт разработчика Unity. Библиотека для реализации мультиплеера в играх на Unity была удалёна, начиная с версии 2018.4; решение «из коробки» для мультиплеера отсутствует. Также можно использовать подходящий пользователю способ контроля версий. К примеру, Tortoise SVN, Git или Source Gear.

В Unity входит Unity Asset Server — инструментарий для совместной разработки на базе Unity, являющийся дополнением, добавляющим контроль версий и ряд других серверных решений.

2.1 ОПРОСЫ ДЛЯ ПРОВЕРКИ АКТУАЛЬНОСТИ ТЕМЫ ПРОЕКТА

а рис. 2.1 (приложения) изображены итоговые результаты опроса, у которого был следующий вопрос: “Вы готовитесь к ЕГЭ по информатике самостоятельно или с чьей-то помощью?”

Динамика развития исследуемого процесса (рис. 2.2 (приложения)) показывает, что около 7 человек из 10 при прохождении ЕГЭ по информатике сталкиваются с трудностями.

Динамика развития исследуемого процесса (рис. 2.3 (приложения)) показывает, что большая часть опрошенных при возможности использовала бы созданное мной приложение, а значит, актуальности моего проекта подтверждена.

2.2 АЛГОРИТМ СОЗДАНИЯ ПРИЛОЖЕНИЯ

1) Регистрация на платформе (Unity Hub)
2) Создание проекта (new project)
3) Создание сцены
4) Создание игрока
5) Написание кода для движения
6) Синхронизация кода и созданного игрока
7) Расширение сцены
8) Создание кода для рестарта уровня (в случае неправильного ответа)
9) Синхронизация кода с модельками объектов
10) Оформление сцены

2.3 ДЕМОНСТРАЦИЯ РАБОТЫ

#Код для осуществления прыжка игроком

```
using UnityEngine;
using UnityEngine.SceneManagement;

Public class Player  : MonoBehaviour
{
        public float jumpForce;
        public Rigidbody2D rb;

        private void Update()
        {
                If (Input.GetKeyDown(KeyCode.Space))
                {
                        Rb.AddForce((Vector)(transform.up*jumpForce), ForceMode2D.Impulse);
                }
        }
        private void OnCollisionEnter2D(Collision2D other)
        {
                If (other.gameObject.CompareTag(“PipePart”))
                {
                        SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);
                }
        }
}
```

Код для движения игрока

```
using UnityEngine;
using UnityEngine.SceneManagement;

public class Player : MonoBehaviour
{
```

```
    public float MovementSpeed = 5;
    public float JumpForce = 10;

    private Rigidbody2D _rigidbody;

    private void Start()
    {
        _rigidbody = GetComponent<Rigidbody2D>();
    }

    private void Update()
    {
        var movement = Input.GetAxis("Horizontal");
        transform.position += new Vector3(movement, 0, 0) * Time.deltaTime * MovementSpeed;

        if (Input.GetButtonDown("Jump") && Mathf.Abs(_rigidbody.velocity.y) < 0.001f)
        {
            _rigidbody.AddForce(new Vector2(0, JumpForce), ForceMode2D.Impulse);
        }
    }

    private void OnCollisionEnter2D(Collision2D other)
    {
        if (other.gameObject.CompareTag("Lava"))
        {
            SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);
        }
    }
}

# Код для закрепления камеры за игроком
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
```

```
public class CameraControl : MonoBehaviour
{
    public float dumping = 1.5f;
    public Vector2 offset = new Vector2(2f, 1f);
    public bool isLeft;

    private Transform player;
    private int lastX;

    [SerializeField]
     float leftLimit;
    [SerializeField]
     float rightLimit;
    [SerializeField]
     float bottomLimit;
    [SerializeField]
     float upperLimit;

    void Start()
    {
        offset = new Vector2(Mathf.Abs(offset.x), offset.y);
        FindPlayer(isLeft);
    }


    public void FindPlayer(bool playerIsLeft)
    {
       player = GameObject.FindGameObjectWithTag("Player").transform;
       lastX = Mathf.RoundToInt(player.position.x);
       if (playerIsLeft)
       {
          transform.position = new Vector3(player.position.x - offset.x, player.position.y - offset.y,
transform.position.z);
       }
```

```
        else
        {
            transform.position = new Vector3(player.position.x + offset.x, player.position.y +
offset.y, transform.position.z);
        }

    }

    void Update()
    {
        if (player)
        {
            int currentX = Mathf.RoundToInt(player.position.x);
            if (currentX > lastX) isLeft = false; else if (currentX < lastX) isLeft = true;
            lastX = Mathf.RoundToInt(player.position.x);

            Vector3 target;
            if (isLeft)
            {
                target = new Vector3(player.position.x - offset.x, player.position.y + offset.y,
transform.position.z);
            }
            else
            {
                target = new Vector3(player.position.x + offset.x, player.position.y + offset.y,
transform.position.z);
            }


            Vector3 currentPosition = Vector3.Lerp(transform.position, target, dumping *
Time.deltaTime);
            transform.position = currentPosition;

        }
```

```
        transform.position = new Vector3
        (
        Mathf.Clamp(transform.position.x, leftLimit, rightLimit),
        Mathf.Clamp(transform.position.y, bottomLimit, upperLimit),
        transform.position.z

        );
    }
}
```

3.1 ОЦЕНКА ЭФФЕКТИВНОСТИ ПРОДУКТА

Для проверки эффективности продукта было проведено два теста на сайте РЕШУ ЕГЭ. Первый до работы в приложении, второй после работы. В этих тестах были собраны задания, к которым в моём приложении давался уже готовый код, единственное, что нужно было изменить в коде, это отдельные места, которые зависели от условия задачи. В тестировании принимали участие 10 человек, которые готовились сдавать ЕГЭ по информатике в 11 классе. По итогам тестирований были получены результаты:

Первое тестирование:

5 человек смогли решить правильно 2/3 или 3/3 заданий
5 человека смогли решить 0/3 или 1/3 заданий

Второе тестирование:

8 человек смогли решить правильно 2/3 или 3/3 заданий
2 человека смогли решить 0/3 или 1/3 заданий

3.2 ИНДИКАТОРЫ

Первое тестирование:

На рис. 3.1 (приложения) изображены итоговые результаты пробного тестирования. Из 3-ёх заданий выполнить 2 или 3 смогли только 50% учащихся (5 человек).

Второе тестирование:

Динамика развития исследуемого процесса (рис. 3.2 (приложения)) показывает, что процент решение задач повысился с 50% до 80%, значит мой продукт оказался полезным.

ЗАКЛЮЧЕНИЕ

Для решения поставленной проблемы, я изучил теоретический материал и реализовал свой продукт на практике.

Я смог решить все поставленные собой задачи, укладываясь в срок. Все задачи поочерёдно выполнялись и, при необходимости дорабатывались. После публичного тестирование была выявлена закономерность, что людям больше понравились сами готовые коды для решения задач ЕГЭ, а прохождение теории они посчитали лишним. В будущем я постараюсь решить данную проблему, чтобы заинтересовать людей в прохождении теоретической части тоже.

После выкладывания приложения в интернет оно станет доступно всем, кто хотел бы его скачать. В будущем, при развитии продукта, ученики смогут получить хорошее приложение для дополнительного образования при подготовке к ЕГЭ

Данное приложение является моим первым опытом в данной сфере. Я узнал много нового, я также приобрёл много новых умений и навыков для будущего развития как проекта, так и других моих деятельностей.

Впоследствии я планирую дорабатывать свой продукт, чтобы он помогал не только при решении определённых задач, но и всего варианта ЕГЭ в целом.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Unity от нуля до мастерства (основы) [Электронный ресурс]: https://unity.codame.online/1_modul
2. Практическая разработка игр в Unity 2022 [Электронный ресурс]: https://translated.turbopages.org/proxy_u/en-ru.ru.e195c769-660e3b81-d3d15072-
3. Кулинарная книга по разработке игр Unity: основы для каждой игры [Электронный ресурс]: https://translated.turbopages.org/proxy_u/en-ru.ru.33b6dcc1-660e3c00-3f36837b
4. Освоение разработки пользовательского интерфейса с Unity [Электронный ресурс]: https://www.gstu.by/sites/default/files/files/resources
5. «Си Шарп для чайников» [Электронный ресурс]: https://dzen.ru/a/yxq2h_c0nvvwrmia
6. Справочник UNITY-разработчика. Все, что нужно, под рукой Ларкович С. [Электронный ресурс]: https://www.chitai-gorod.ru/product/spravochnik-unity-razrabotchika-vse-chto-nuzhno-pod-rukoy-3017782
7. Unity в действии: разработка мультиплатформенных игр на C # Хокинг Д. [Электронный ресурс]: https://www.amazon.co.uk/dp/1617299332?tag=hackr-21&geniuslink=true
8. Разработка игр с Unity для .NET разработчиков Цзядун Ч. [Электронный ресурс]: https://www.packtpub.com/product/game-development-with-unity-for-net-developers/9781801078078
9. «Unity и C#. Геймдев от идеи до реализации» Гибсон Д. Б. [Электронный ресурс]: https://www.labirint.ru/reviews/goods/686355/?p=2155&admitad_uid=35d1d72b2bd9db19e7a0610a576a1729&publisher
10. «Unity. Полное руководство» Корнилова А. [Электронный ресурс]: https://www.bookvoed.ru/product/unity-polnoe-rukovodstvo-6106031?id=13392865&partner

Приложения

Рис 2.1

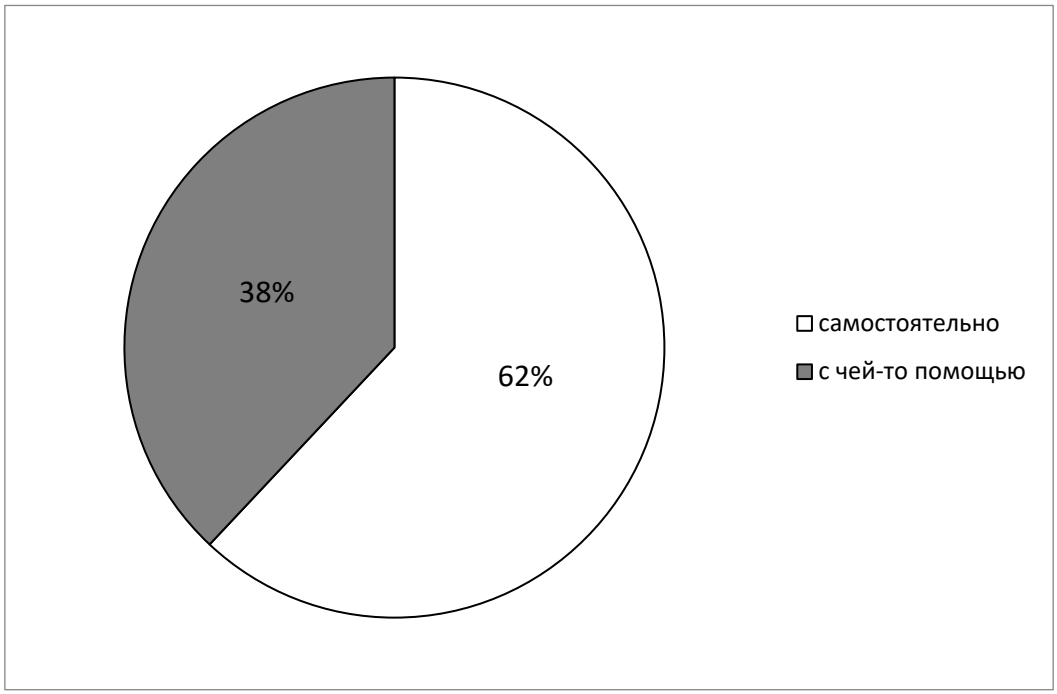

Рис 2.2

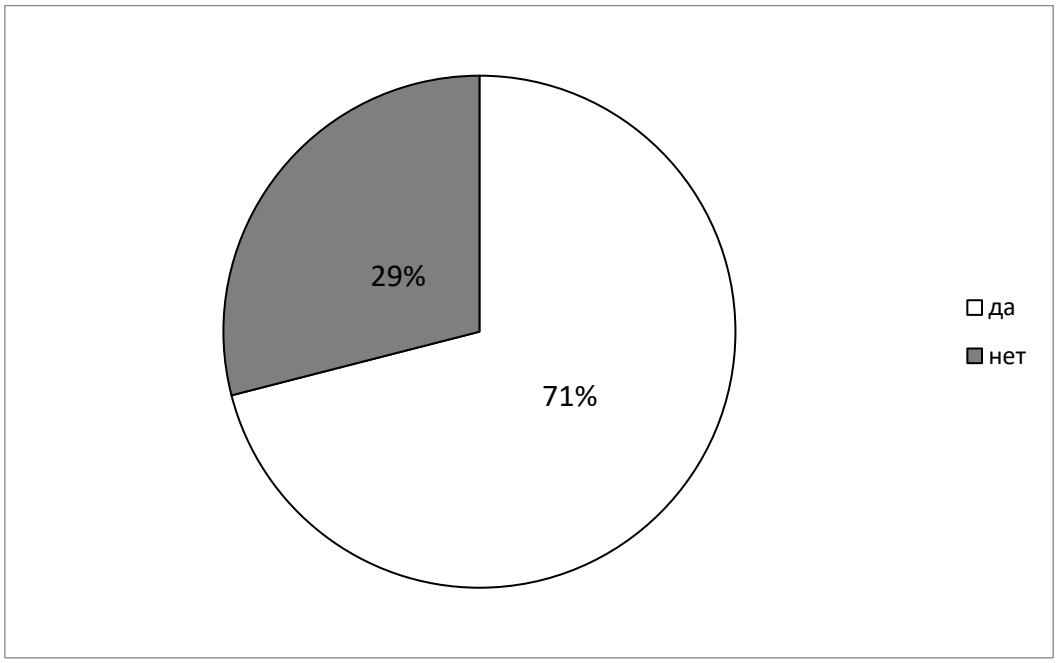

Рис 2.3

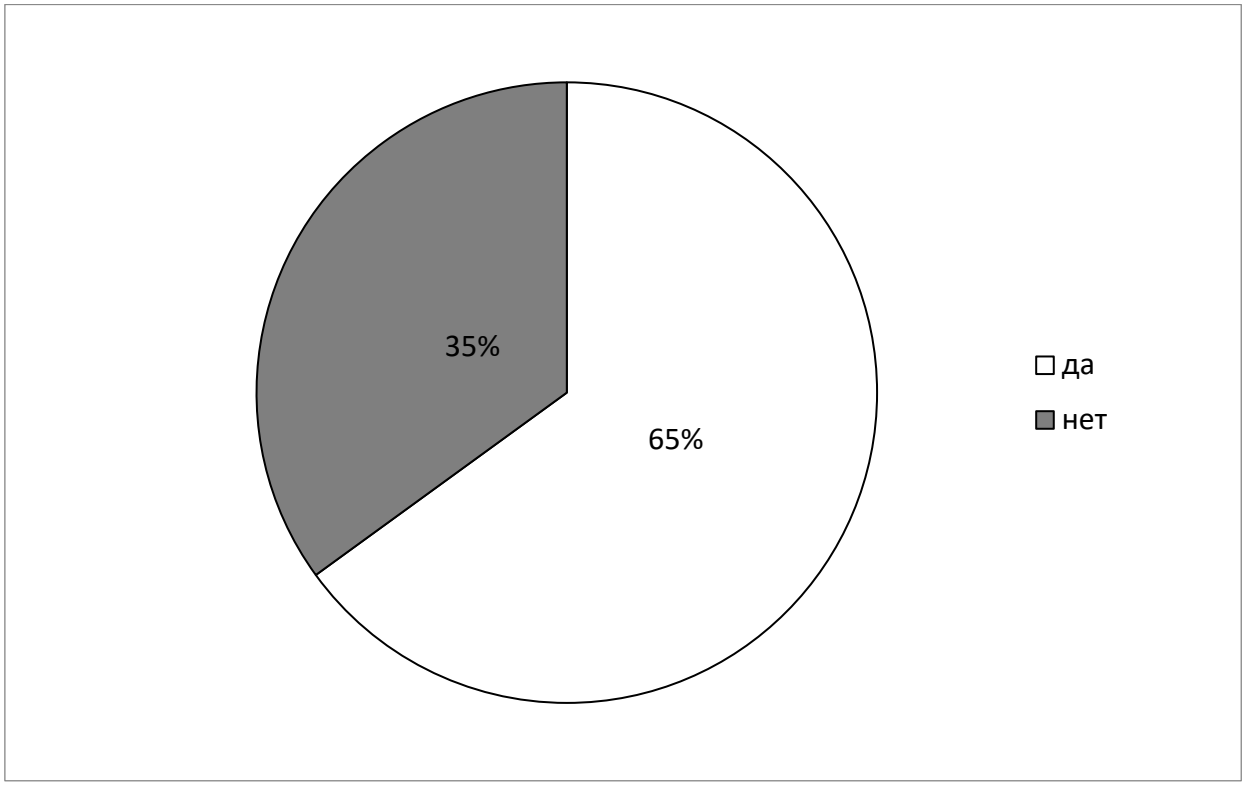

Рис 3.1

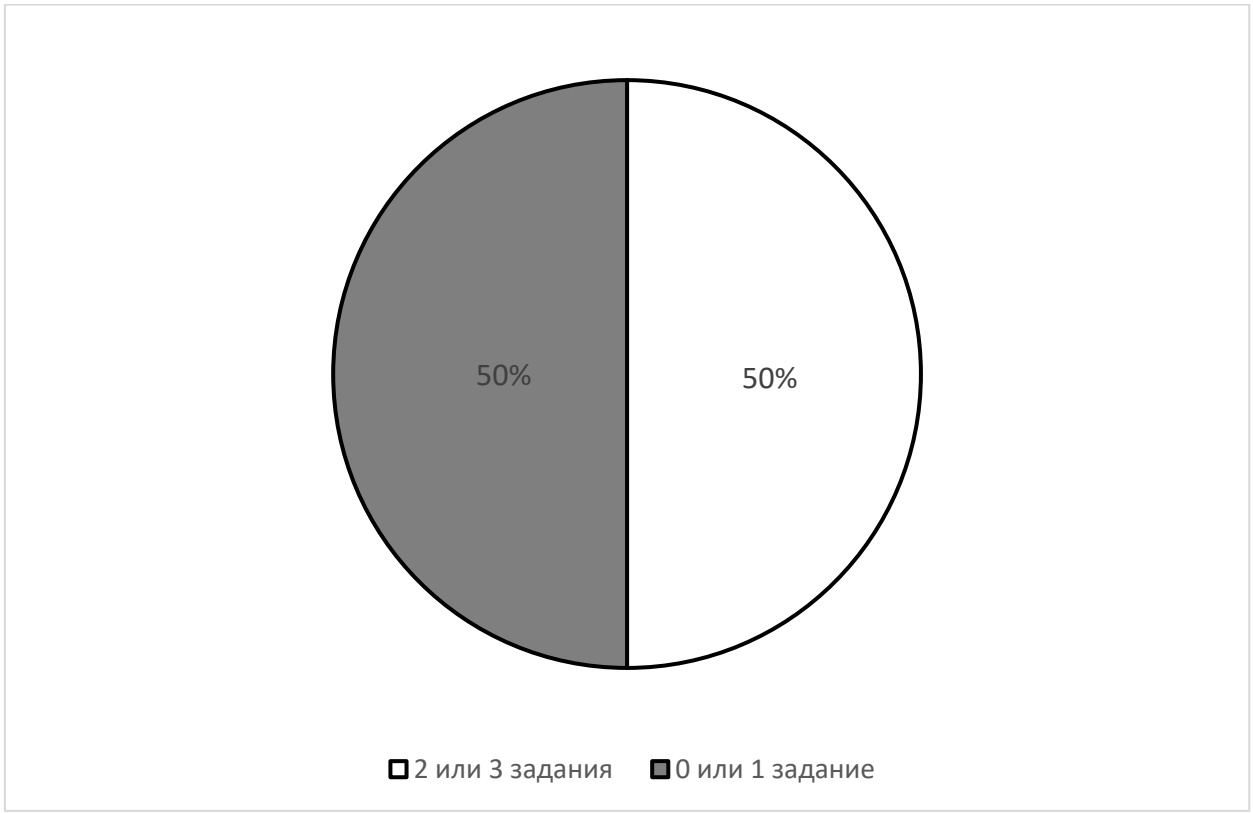

Рис 3.2