2. ВТОРОЙ ЭТАП

Задачи второго этапа. Дополненная реальность

3.1. Основные понятия и определения технологии дополненной реальности

Задача 3.1.1. (1 балл)

Сопоставьте понятия и определения:

1. Результат введения в поле восприятия любых сенсорных данных с целью дополнения сведений об окружении и улучшения восприятия информации
2. Свойство технологической части среды, отражающее её возможности по вовлечению субъекта в систему отношений, определяемую содержанием среды
3. Технология 3D-захвата, которая позволяет создавать 3D-модели людей, сжимать их и передавать в любую точку мира в реальном времени
4. Технология, полностью погружающая человека в синтетическую среду
5. Является следствием объединения реального и виртуальных миров для создания новых окружений и визуализаций, где физический и цифровой объекты сосуществуют и взаимодействуют в реальном времени

а. Смешанная (гибридная) реальность

б. Голопортация

в. Иммерсивность

г. Дополненная реальность

д. Виртуальная реальность

Ответ: 1 - г, 2 - в, 3 - б, 4 - д, 5 - а.

Задача 3.1.2. (1 балл)

В иммерсивных технологиях сокращение MR означает...

1. More Reality
2. Measured Reality
3. Mixed Reality

4. Mirrored Reality

Ответ: 3.

Задача 3.1.3. (1 балл)

Как называется устройство для работы со смешанной реальностью от компании Microsoft?

1. Gear VR
2. MicroLens
3. HoloLens
4. VRLens

Ответ: 3.

Задача 3.1.4. (1 балл)

Что из перечисленного не является шлемом для виртуальной реальности (не существует)?

1. Gear VR
2. Vive
3. Xbox VR
4. Oculus

Ответ: 3.

Задача 3.1.5. (1 балл)

Сопоставьте понятия и определения:

1. Объект реального мира, являющийся поводом поставления в видеопоток камеры устройства пользователя дополнительной информации в виде виртуальных объектов
2. Информация, добавляемая в видеопоток камеры устройства пользователя при считывании маркера, распознавании 3Д-объекта, определения локации

а. Оверлей (аура)

б. Мишень (триггер, маркер и т.п.)

Ответ: 1 - б, 2 - а.

Задача 3.1.6. (1 балл)

Определите точку континуума реальность-виртуальность (тип реальности): участник-наблюдатель полностью погружен и взаимодействует с полностью искус-

ственным миром

1. Дополненная виртуальность
2. Гибридная реальность
3. Дополненная реальность
4. Виртуальная реальность

Ответ: 4.

Задача 3.1.7. (1 балл)

Определите точку континуума реальность-виртуальность (тип реальности): кто-то в виртуальной среде, зафиксированный камерой, выступающей в качестве точки зрения третьего лица

1. Дополненная виртуальность
2. Опосредованная реальность
3. Дополненная реальность
4. Виртуальная реальность

Ответ: 3.

Задача 3.1.8. (1 балл)

Определите точку континуума реальность-виртуальность (тип реальности): результат добавления к воспринимаемым в качестве элементов реального мира объектам мнимых объектов

1. Дополненная виртуальность
2. Опосредованная реальность
3. Дополненная реальность
4. Виртуальная реальность

Ответ: 3.

Задача 3.1.9. (1 балл)

Определите точку континуума реальность-виртуальность (тип реальности): способность добавлять, вычитать информацию или иным образом манипулировать своим восприятием реальности с помощью компьютера или ручного устройства, такого как смартфон. Визуальное восприятие среды пользователем при помощи кого-либо электронного устройств. Например, используется для улучшения зрительного восприятия в качестве вспомогательного средства для слабовидящих.

1. Дополненная виртуальность
2. Опосредованная реальность

3. Дополненная реальность
4. Виртуальная реальность

Ответ: 2.

Задача 3.1.10. (1 балл)

По типу взаимодействия с пользователем дополненная реальность бывает:

1. визуальная
2. мобильная
3. аудиовизуальная
4. интерактивная
5. аудио
6. автономная
7. геопозиционная
8. оптическая
9. стационарная

Ответ: 4, 6.

Задача 3.1.11. (1 балл)

По степени мобильности дополненная реальность бывает:

1. визуальная
2. мобильная
3. аудиовизуальная
4. интерактивная
5. аудио
6. автономная
7. геопозиционная
8. оптическая
9. стационарная

Ответ: 2, 9.

Задача 3.1.12. (1 балл)

По типу устройств, считывающих информацию, дополненная реальность бывает:

1. визуальная
2. мобильная

3. аудиовизуальная
4. интерактивная
5. аудио
6. автономная
7. геопозиционная
8. оптическая
9. стационарная

Ответ: 7, 8.

Задача 3.1.13. (1 балл)

По типу представления информации дополненная реальность бывает:

1. визуальная
2. мобильная
3. аудиовизуальная
4. интерактивная
5. аудио
6. автономная
7. геопозиционная
8. оптическая
9. стационарная

Ответ: 1, 3, 5.

Задача 3.1.14. (1 балл)

Прочитайте внимательно диалог специалистов бюро виртуальности «RealLife», представленный ниже. Сопоставьте героев диалога с их профессией.

13.56 ARCI-VR:

Андрей, наш заказчик - NASA, а не клуб любителей космической фантастики. По-твоему, так выглядит марсианский закат? Откуда здесь эти томные зеленоватые тона? Посмотри фотки с Opportunity – чистый голубой цвет. Тебе два дня, чтобы все исправить.

13.56 ANDREW:

OK. Хотя мне этот вариант кажется живописнее. Зато я допридумал грунт – теперь шаги будут звучать реалистично.

14.02 ARCI-VR:

И гравитацию подкрути, у тебя на прыжках картинка не совпадает.

14.05 ANDREW:

Еще с утра подкрутил. Тут Надя жалуется – набросала уже 20 стрессовых ситуаций для второго этапа подготовки колонистов, но до консультации с психологами мы не можем утверждать сценарии. Когда они уже результаты тестов пришлют?

1. ARCI-VR	а. дизайнер эмоций
2. ANDREW	б. дизайнер виртуальных миров
3. Надя	в. архитектор виртуальности

Ответ: 1 - в, 2 - б, 3 - а.

Задача 3.1.15. (1 балл)

К какой отрасли относятся следующие новые профессии: архитектор виртуальности, дизайнер виртуальных миров, дизайнер эмоций, игропрактик, редактор агрегаторов контента, инфостилист?

1. ИТ-сектор
2. социальная сфера
3. культура и искусство
4. медиа и развлечения
5. индустрия детских товаров и сервисов

Ответ: 4.

Задача 3.1.16. (1 балл)

Выберите факторы, препятствующие развитию и распространению дополненной реальности

1. недостаточный эффект погружения
2. высокая стоимость AR-устройств
3. отсутствие сред разработки

Ответ: 1, 2.

3.2. Общие сведения об инструментах, методах и алгоритмах AR

Задача 3.2.1. (1 балл)

Сопоставьте браузеры дополненной реальности с их иконками

1. 1
2. 2
3. 3
4. 4

а. HP Reveal
б. Wikitude
в. Layar
г. BlippAR

Ответ: 1 - а, 2 - б, 3 - г, 4 - в.

Задача 3.2.2. (1 балл)

Перед Вами стоит задача размещения во внешнем периодическом печатном издании рекламной статьи о своей компании, при этом в текст статьи должны быть зашиты ссылки на группы в социальных сетях и YouTube-канал, полноценный проморолик и другая интересная пользователю информация. Такие рекламные статьи Вы собираетесь публиковать в разных изданиях, а также выпускать буклеты, листовки и другую сувенирную продукцию. Вы работаете маркетологом, а разработчика ПО в Вашей компании нет. Какой инструмент для работы с дополненной реальностью Вы будете использовать?

1. HP Reveal
2. LayAR
3. Vuforia
4. EasyAR
5. ARToolKit

Ответ: 1.

Задача 3.2.3. (1 балл)

Вы являетесь высокотехнологичной компанией, осуществляющей разработку настольных игр. Добавляя дополненную реальность в свой продукт, Вы рассчитываете создать нативное брендированное великолепие, принадлежащее только вам, плюс иконка компании на рабочих столах мобильных устройств целевой аудитории, возможность обеспечения ряда уникальных функции по взаимодействию пользователя и виртуальных объектов в игре. Какой инструмент для работы с дополненной реальностью Вы будете использовать?

1. HP Reveal
2. Vuforia
3. EasyAR
4. ARToolKit

Ответ: 2, 3, 4.

Задача 3.2.4. (1 балл)

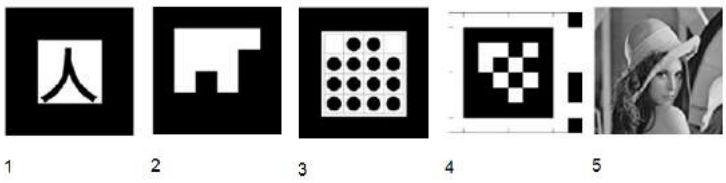
Вы работаете в визитно-информационном центре города. В рамках программы развития комфортного пребывания в вашем городе гостей Вы хотите создать приложение, предоставляющее пользователям информацию: о достопримечательностях, отелях, предприятиях общественного питания и т.п., в виде всплывающих на экранах смартфона объектах дополненной реальности, находящихся в непосредственной близости от клиента, в секторе угла обзора его гаджета, с возможностью проложить маршрут из точки в точку. Какой инструмент для работы с дополненной реальностью Вы будете использовать?

1. HP Reveal
2. Wikitude
3. Vuforia
4. EasyAR
5. ARToolKit
6. BlippAR
7. LayAR

Ответ: 2, 7.

Задача 3.2.5. (1 балл)

Сопоставьте название баз маркеров с их изображением



1. 1
2. 2
3. 3
4. 4
5. 5

а. Не существует соответствующей базы маркеров

б. Hoffman marker system (HOM) used by SCR and Framatome ANP

в. Siemens Corporate Research (SCR) marker system

г. Institut Graphische Datenverarbeitung (IGD) marker system

д. ArToolKit(ATK) marker system

Ответ: 1 - д, 2 - г, 3 - в, 4 - б, 5 - а.

Задача 3.2.6. (1 балл)

Соотнесите маркер дополненной реальности с его типом:

1. 1
2. 2
3. 3
4. 4

а. Multimarker
б. Image / NFT Marker
в. GPS Marker
г. Framemarker

Ответ: 1 - г, 2 - б, 3 - в, 4 - а.

Задача 3.2.7. (1 балл)

Согласны ли Вы с данным утверждением: «Для качественного трекинга достаточно использовать информацию с гироскопа, акселерометра и магнитометра смартфона»?

1. Да
2. Нет

Ответ: 2.

Задача 3.2.8. (1 балл)

Сопоставьте понятия и определения:

1. Эвристические алгоритмы поиска, используемые для решения задач оптимизации и моделирования путём случайного подбора, комбинирования и вариации искомых параметров с использованием механизмов, напоминающих биологическую эволюцию.
2. Методы, нацеленные на вычисление абстракций изображения и выделения на нем ключевых особенностей. Данные особенности могут быть как в виде изолированных точек, так и кривых или связанных областей.
3. Теория, являющаяся основополагающей для развития технологий дополненной реальности, и прежде всего в области использования маркеров

а. компьютерное зрение
б. feature detection
в. генетические алгоритмы

Ответ: 1 - в, 2 - б, 3 - а.

Задача 3.2.9. (1 балл)

Сопоставьте термины на английском языке с их развернутым значением на русском:

1. Некоторый участок картинки, который является отличительным для заданного изображения
2. Метод оценки положения и ориентации предмета, для которого собираются данные с датчиков смартфона и информация о визуальном трекинге
3. Процесс определения местоположения и ориентации объекта при помощи камеры

а. features points

б. odometriya

в. tracking

Ответ: 1 - а, 2 - б, 3 - в.

Задача 3.2.10. (1 балл)

Составьте правильную последовательность действий в работе алгоритма распознавания маркера:

1. Приводим в градации серого
2. Выделяем контуры
3. Выделяем углы маркера
4. Определение замкнутых областей
5. Бинаризация изображения (порог)
6. Преобразуем координаты

Ответ: 1, 5, 4, 2, 3, 6.

Задача 3.2.11. (1 балл)

В чем основное отличие между маркерной и безмаркерной дополненной реальностью?

1. В маркерной дополненной реальности используется алгоритм ORB, тогда как в безмаркерной используется SIFT.
2. В маркерной дополненной реальности маркер предопределен (задан заранее), тогда как в безмаркерной необязательно иметь полное представление о искомом объекте.
3. Маркерная дополненная реальность не использует ключевые точки и дескрипторы для распознавания объекта.

Ответ: 2.

Задача 3.2.12. (1 балл)

Согласны ли вы с утверждением, что в отличие от Vuforia, ARKit не распознает точки высокого контраста?

1. Да
2. Нет

Ответ: 2.

Задача 3.2.13. (1 балл)

Какие факторы, из перечисленных ниже, важно учитывать при разработке приложений с помощью ARKit?

1. Нельзя использовать контрастные поверхности
2. Рекомендуется использовать небликующие поверхности
3. Для приложений на ARKit рекомендуется механика с быстрыми перемещениями устройства
4. Необходимо отображать распознавание поверхности

Ответ: 2, 4.

Задача 3.2.14. (1 балл)

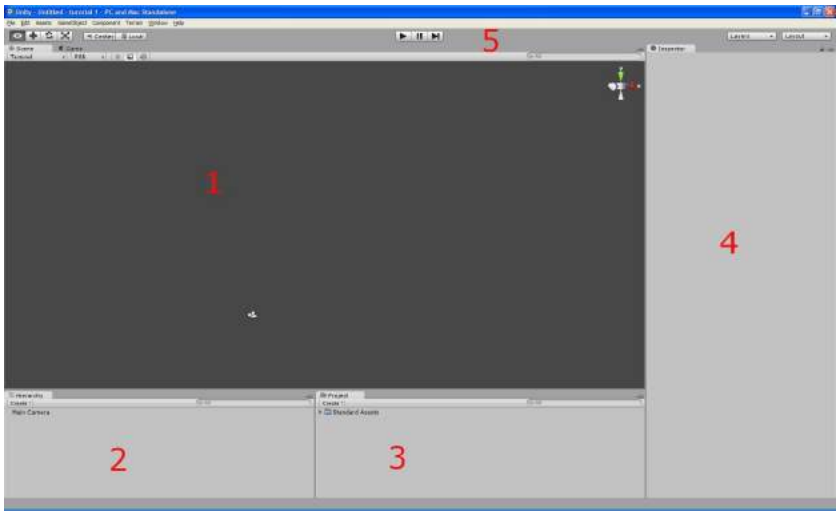
Что из перечисленного ниже распознает ARKit SDK?

1. горизонтальные поверхности
2. вертикальные поверхности
3. объекты произвольной формы
4. заранее отсканированные 3D объекты

Ответ: 1, 2.

3.3. Общее представление о SDK Unity

Задача 3.3.1. (1 балл)



Сопоставьте номера, отмечающие на рисунке ниже зоны различного функционального назначения в SDK Unity, с их кратким текстовым описанием

1. 1
2. 2
3. 3
4. 4
5. 5

а. Иерархия объектов на сцене
б. Инспектор префабов и ресурсов
в. Главное окно редактирования
г. Инспектор объектов
д. Кнопки управления

Ответ: 1 - в, 2 - а, 3 - б, 4 - г, 5 - д.

Задача 3.3.2. (1 балл)

Для чего нужна вкладка Hierarchy?

1. В ней располагаются объекты на сцене
2. В ней находятся все материалы к проекту
3. В нее выводятся различные ошибки и надписи в ходе игры
4. В ней располагаются свойства к объектам
5. В ней отображается игровая сцена

Ответ: 1.

Задача 3.3.3. (1 балл)

Для чего нужна вкладка Inspector?

1. В ней отображается игровая сцена
2. В нее выводятся различные ошибки и надписи в ходе игры
3. В ней располагаются объекты на сцене
4. В ней располагаются свойства к объектам
5. В ней находятся все материалы к проекту

Ответ: 4.

Задача 3.3.4. (1 балл)

Для чего нужна вкладка Scene?

1. В ней отображается игровая сцена
2. В ней располагаются свойства к объектам
3. В ней находятся все материалы к проекту
4. В ней располагаются объекты на сцене
5. В нее выводятся различные ошибки и надписи в ходе игры

Ответ: 1.

Задача 3.3.5. (1 балл)

Для чего нужна вкладка Project?

1. В ней отображается игровая сцена
2. В ней располагаются свойства к объектам
3. В ней находятся все материалы к проекту
4. В ней располагаются объекты на сцене
5. В нее выводятся различные ошибки и надписи в ходе игры

Ответ: 3.

Задача 3.3.6. (1 балл)

Где отображаются различные сообщения?

1. в инспекторе
2. во вкладке иерархии
3. во вкладке консоль
4. во вкладке проект

Ответ: 3.

Задача 3.3.7. (1 балл)

Как называется официальный магазин от Unity?

1. Projects Store
2. Unity Store
3. Packages Store
4. Asset Store
5. Plugins Store

Ответ: 4.

Задача 3.3.8. (1 балл)

Какие объекты добавлены по умолчанию при создании 3D-проекта в Unity?

1. Никаких объектов нет по умолчанию
2. Main Camera
3. Main Camera и Directional light
4. Directional light
5. Terrain

Ответ: 3.

Задача 3.3.9. (1 балл)

Какой компонент есть у каждого объекта?

1. Light
2. Rigidbody
3. Collider
4. Mesh
5. Transform

Ответ: 5.

Задача 3.3.10. (1 балл)

Можно ли вращать и передвигать камеру?

1. Можно, только если камера не основная
2. Можно всегда
3. Нет, нельзя

Ответ: 2.

Задача 3.3.11. (1 балл)

При импортировании файла какого формата в Unity в редактор передается только 3D-модель, а тесктуры и анимацию необходимо передавать отдельно?

1. fbx
2. obj

Ответ: 2.

Задача 3.3.12. (1 балл)

Выберите правила, которым должны отвечать 3D-модели для AR-приложений, создаваемых в Unity.

1. Соответствие текстурной карты размеру треугольника
2. Небольшое количество полигонов
3. Расположение текстур в рамках одного UV-пространства
4. Максимально возможное количество полигонов у модели

Ответ: 2, 3.

Задача 3.3.13. (1 балл)

Правило квадрата при создании 3D-моделей гласит о том, что каждая текстура должна четко вписываться в квадрат

1. Да
2. Нет

Ответ: 2.

Задача 3.3.14. (1 балл)

Чему равна одна единица измерения в Unity (юнит)?

1. 1 пикселу
2. 1 метру
3. 1 миллиметру
4. 1 сантиметру
5. 1 дециметру

Ответ: 2.

Задача 3.3.15. (1 балл)

Как в Unity называется объект, при изменении которого меняются все его объекты наследники?

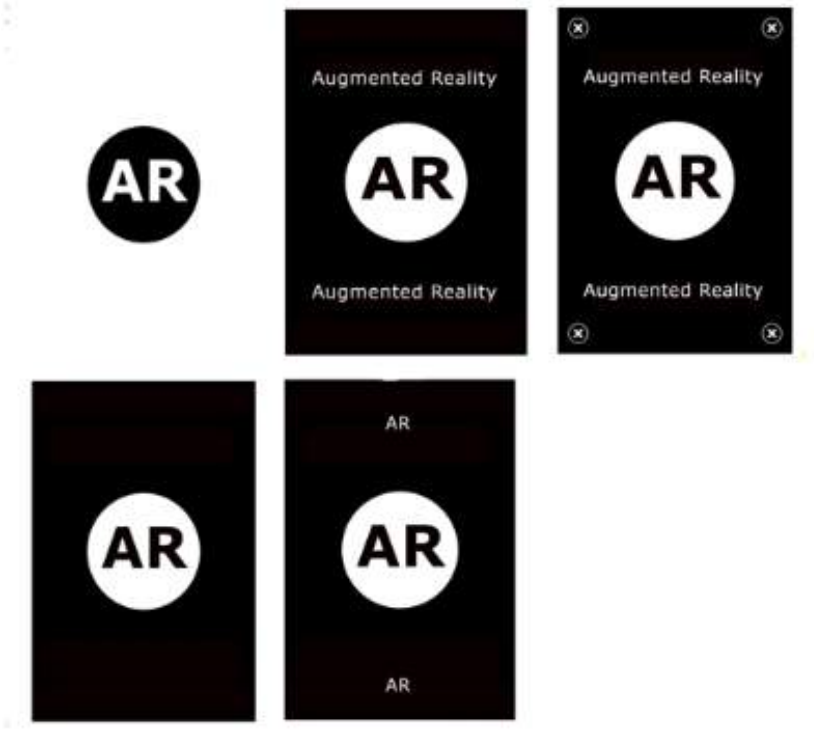
1. GameObject
2. Asset
3. Prefab

Ответ: 3.

3.4. Фремворк Vuforia

Задача 3.4.1. (1 балл)

Во фреймворке Vuforia для оценки качества изображения используется параметр Augmentable. Определите его возможное значение от 0 до 5 для следующих пяти вариантов маркеров, представленных на рисунке (изображение выполнено в формате A4).



Ответ запишите в виде числа, цифрами без пробелов. Например: 34521.

Ответ: 05512.

Задача 3.4.2. (1 балл)

Какое из приведенных на рисунке изображений, будет распознаваться лучше всего?

1. 1
2. 2
3. 3
4. 4

Ответ: 2.

Задача 3.4.3. (1 балл)

Какие помехи на изображении приводят к понижению значения параметра Augmentable во фреймворке Vuforia :

1. Неравномерно размещены ключевые точки (в каком-то месте изображения ключевых точек значительно больше)
2. Недостаточная контрастность изображения.
3. Ключевые точки размещены равномерно.
4. Изображение черно-белое, а алгоритм привередлив к цветам.

Ответ: 1, 2.

Задача 3.4.4. (1 балл)

Какие операции нужно проделать с изображением, представленным на рисунке, чтобы улучшить его распознавание, при разработке приложений дополненной реальности при помощи фреймворка Vuforia?

1. Это изображение распознаваться не будет
2. Увеличить контраст
3. Перевести в оттенки серого
4. Увеличить яркость

Ответ: 2, 4.

Задача 3.4.5. (1 балл)

Какие операции нужно проделать с изображением, представленным на рисунке, чтобы улучшить его распознавание, при разработке приложений дополненной реальности при помощи фреймворка Vuforia?

1. Это изображение распознаваться не будет
2. Увеличить контраст
3. Перевести в оттенки серого
4. Увеличить яркость

Ответ: 1.

Задача 3.4.6. (1 балл)

Представьте, что вам для AR-приложения, разрабатываемого при помощи фреймворка Vuforia, необходимо 3000 таргетов. Какой тип хранения лучше использовать?

1. Device
2. Cloud

Ответ: 2.

Задача 3.4.7. (1 балл)

Выберите форматы, подходящие для создания ImageTarget во фреймворке Vuforia

1. png
2. raw
3. gif
4. tiff
5. bmp
6. jpg

Ответ: 1, 6.

Задача 3.4.8. (1 балл)

При создании AR-приложения в Unity 3D с использованием фреймворка Vuforia для его стабильной работы в разлеле PlaySettings необходимо снять галочку с:

1. Android Game
2. Vuforia Augmented Reality Support
3. Android TV Compatibility

Ответ: 3.

Задача 3.4.9. (1 балл)

Что нельзя использовать в названии датабазы, создаваемой в Vuforia?

1. пробелы
2. цифры
3. прописные буквы
4. дополнительные символы

Ответ: 1, 4.

Задача 3.4.10. (1 балл)

Что необходимо указать при загрузке таргета в Vuforia Target Manager?

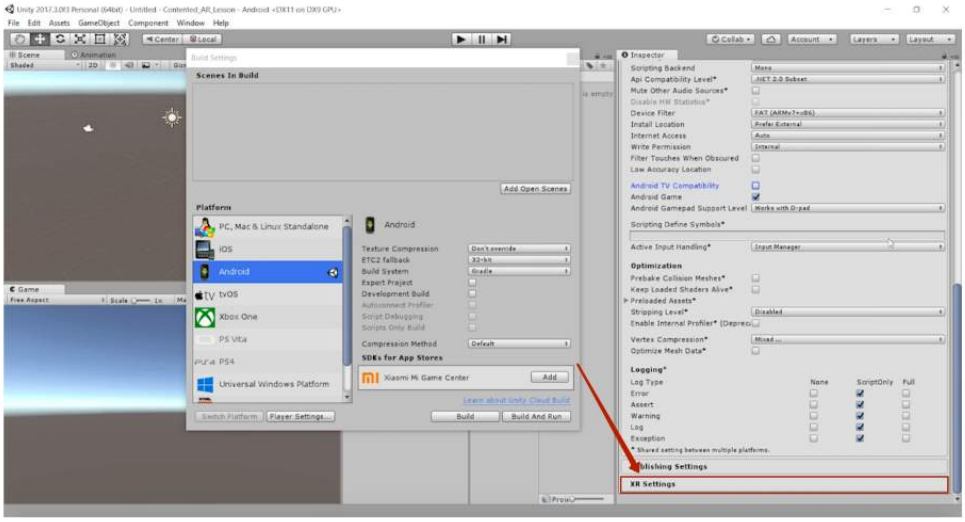
1. длину
2. ширину
3. никакие параметры указывать не надо, среда разработки все установит автоматически

Ответ: 2.

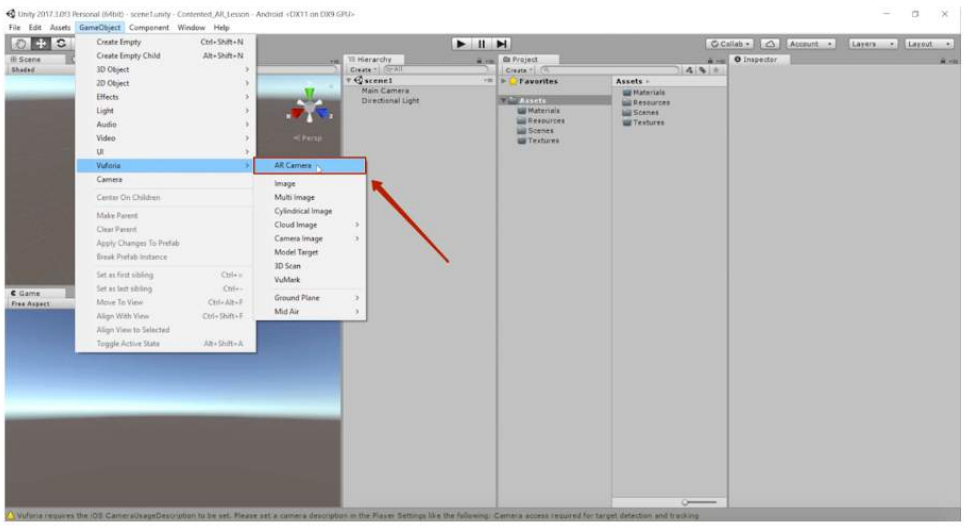
Задача 3.4.11. (1 балл)

Выберите скриншот, в котором указано, куда необходимо перейти, чтобы открыть настройки Vuforia

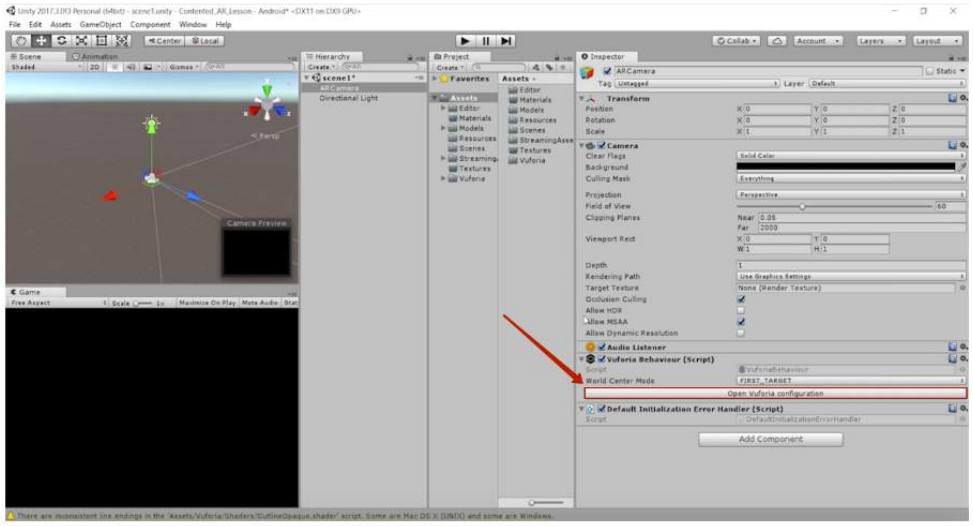
1.



2.



3.



1. 1
2. 2
3. 3

Ответ: 3.

Задача 3.4.12. (1 балл)

Что значит, если при сканировании объекта с помощью VuforiaObjectScanner полигоны сферы окрашиваются в зеленый цвет?

1. Запущено сканирование в этом фрагменте объекта
2. Сканирование в этом фрагменте объекта еще не начато
3. Сканирование в этом фрагменте объекта успешно завершено

Ответ: 3.

Задача 3.4.13. (1 балл)

Что из перечисленного подойдет в качестве 3D-объекта для трекинга?

1. Очки
2. Фломастер
3. Ведро

Ответ: 2.

Задача 3.4.14. (1 балл)

Какой должна быть поверхность для размещения на ней листа и объекта для сканирования при помощи VuforiaObjectScanner?

1. Блестящей
2. Небликующей
3. Цветной с яркими контрастными пятнами
4. Однородного цвета

Ответ: 2, 4.

Задача 3.4.15. (1 балл)

Какого размера должен быть лист для сканирования при помощи VuforiaObjectScanner по отношению к объекту?

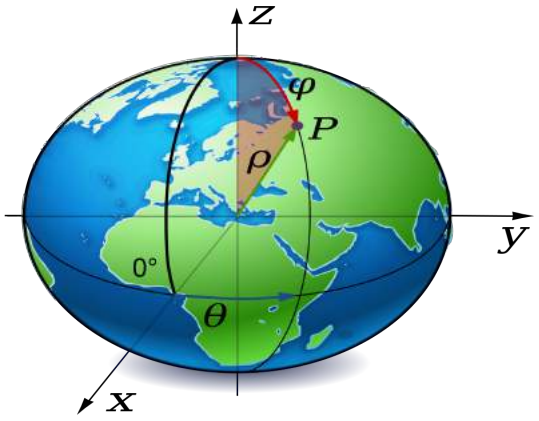
1. Объект может немного выходить за границы для его размещения
2. Объект должен полностью вписываться в область для его размещения
3. Объект должен влезать в область минимум на 1/3

Ответ: 2.

3.5. Основы работы с геоданными

Задача 3.5.1. (1 балл)

Основой для работы с геоданными является система координат. С начала XX века было предложено множество систем. Отметьте, какие из перечисленных систем подходят для определения координат на земной поверхности.



1. ПЗ-90 (Параметры Земли)
2. СК-42 (Система Координат)
3. WGS84 (World Geodetic System)

4. CCS (Celestial coordinate system)
5. EGS99 (Earth Global System)
6. ISO 6709 (Standard representation of geographic point location by coordinates)
7. (Ecl-CS) Ecliptic coordinate system
8. Geo URI (Uniform Resource Identifier)

Ответ: 1, 2, 6, 8.

Пояснения к ответу

Остальные варианты ответов не являются системами координат (вымышленные названия) или не подходят для земной поверхности

Задача 3.5.2. (1 балл)

Какая из систем геокоординат используется для определения расстояния между точками в ОС Android.

Подробности на сайте разработчика операционной системы (класс Location `https://developer.android.com/reference/android/location/Location`).

1. ISO 6709
2. EGS99
3. WGS84
4. Ecl-CS
5. СК-42
6. ПЗ-90

Пояснения к ответу

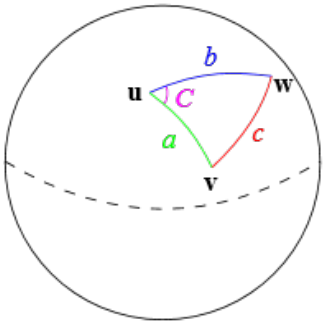
Подробности на сайте разработчика операционной системы (класс Location).

Достаточно почитать описание функций класса, например, getAltitude().

Ответ: 3.

Задача 3.5.3. (1 балл)

Перед вами фотографии двух памятников - паровозы на почётной стоянке у вокзалов. Определите поиском по картинке, в каком городе находится каждый памятник и определите их координаты. Рассчитайте в системе геокоординат расстояние между памятниками по дуге большого круга. Ответ вводится в километрах, допустимая погрешность - 2 км.



Пояснения к ответу

Поиском по картинкам в интернете определяем, что речь о паровозах-памятниках в Красноярске и Улан-Удэ. В онлайн-сервисах карт находим координаты каждого памятника. Рассчитать расстояние можно в сервисе карт или по координатам (`https://en.wikipedia.org/wiki/Haversine_formula`) с помощью программ или сервисов (например, `https://www.geodatasource.com/distance-calculator` или `https://www.movable-type.co.uk/scripts/latlong.html`).

Ответ: 863 км.

Задача 3.5.4. (1 балл)

Для описания объектов на картах часто используется формат GeoJSON. Какие типы геометрических объектов в нём используются?

1. MultiPolygon

2. Area
3. Rect
4. MultiLineString
5. Polygon
6. MultiPoint
7. Square
8. Circle
9. Point
10. LineString

Пояснения к ответу

В спецификации формата перечислены типы объектов.

Ответ: 1, 4, 5, 6, 9, 10.

Задача 3.5.5. (7 баллов)

Продолжаем знакомство с GeoJSON. Вам даны контуры нескольких стран в формате GeoJSON (архив по ссылке `https://github.com/ipetrushin/NTI-Contest/blob/master/8countries-geojson.zip`). С помощью онлайн-инструментов определите, какой из стран соответствует каждый из файлов и сопоставьте элементы списка ниже.

1. CountryA
2. CountryB
3. CountryC
4. CountryD
5. CountryE
6. CountryF
7. CountryG

а. Мальдивы
б. Италия
в. Россия
г. Мальта
д. Перу
е. Камбоджа
ж. Ватикан

Пояснения к ответу

Каждый из контуров в файле можно наложить на реальную географическую карту с помощью онлайн-сервиса, например `http://geojson.io`, по положению и подписи легко определить страну.

Ответ: 1 - б, 2 - е, 3 - а, 4 - ж, 5 - в, 6 - д, 7 - г.

Задача 3.5.6. (14 баллов)

Шаг 1. Скачайте файл, содержащий описание зданий, являющихся достопримечательностями города Иркутска. По описанию отыщите эти здания в сети интернет

и определите их географические координаты. Заполните таблицу с полями:

1. Координаты lon, lat
2. Описание
3. Название

Вам необходимо найти координаты ЦЕНТРА каждого здания, мы рекомендуем для этого воспользоваться сервисом 2ГИС (ВАЖНО!). Заполненная Вами таблица будет являться вспомогательным инструментом в следующих этапах задания.

Шаг 2. Правильное решение этого шага задачи оценивается в 14 баллов. Найдите координаты (lon, lat) центра окружности радиусом 2 км, находясь в котором пользователь увидит на экране смартфона максимальное количество достопримечательностей, описание которых дано в файле задания.

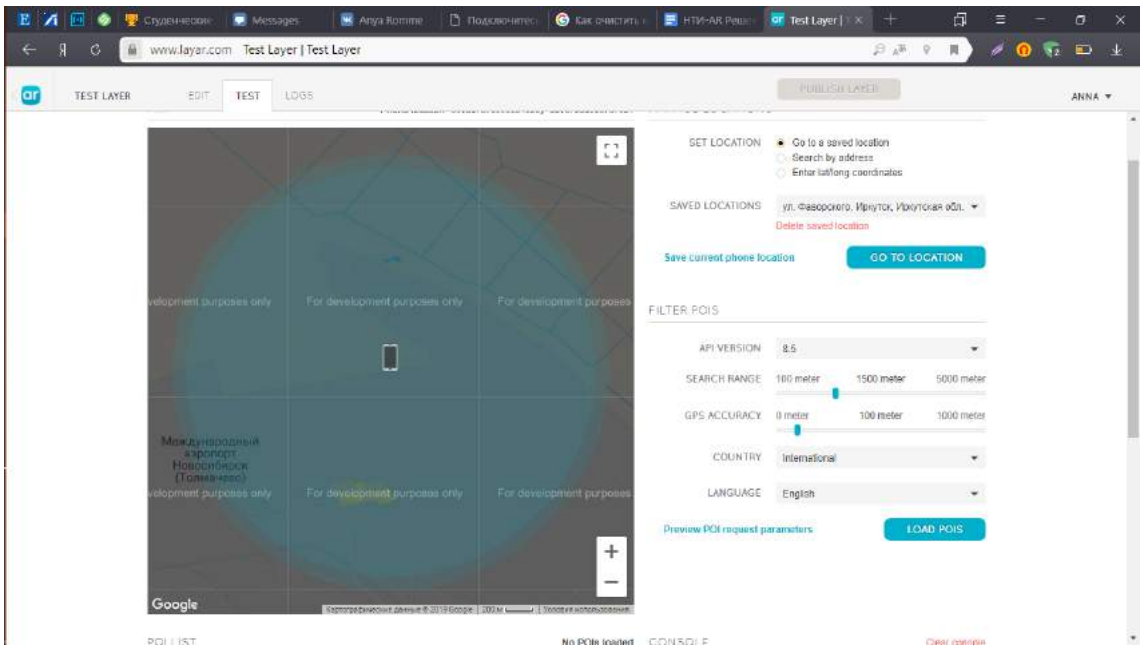
Для решения этой задачи мы рекомендуем воспользоваться одним из браузеров дополненной реальности: создать гео-слой с отображением на нем необходимых POI (points of interest, координат достопримечательностей, определенных вами в предыдущем шаге). Урок по созданию простого гео-слоя расположен по ссылке (`https://www.layar.com/documentation/browser/tutorials-tools/create-simple-geo-location-layer/`). Также вам может пригодиться мобильное приложение для подмены геопозиции вашего мобильного устройства.

В качестве ответа необходимо прикрепить долготу и широту через пробел.

Пример ответа: 37.531276 55.702651

Решение

Есть несколько способов решения этой задачи. Самый простой - это перебор. Для этого можно при помощи урока (`https://www.layar.com/documentation/browser/tutorials-tools/create-simple-geo-location-layer/`) создать гео-слой (минус этого способа в том, что для этого потребуется хостинг) или вручную нанести на карту все найденные точки (если не хотите разбираться с хостингом, php и БД). Если вы используете Layar, то на сайте можно протестировать слой. Во время тестирования можно перетаскивать иконку телефона и смотреть, сколько точек попадают в окружность.



Второй способ - написать программу на любом языке программирования (например, Python), которая найдет необходимую точку. Для решения этим способом понадобится функция для расчета расстояния (в метрах) между двумя точками, заданными в формате географических координат (lon, lat). Для этой функции понадобится формула Гаверсинуса:

$$d = 2rarcsin\left(\sqrt{sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + cos(\phi_1)cos(\phi_2)sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)$$

Функция расчета этой формулы может выглядеть вот так:

```
def haversine(coord1, coord2):
    lon1 = coord1[0]
    lon2 = coord2[0]

    lat1 = coord1[1]
    lat2 = coord2[1]

    EARTH_RADIUS = 6371000

    lon1, lat1, lon2, lat2 = map(math.radians, [lon1, lat1, lon2, lat2])
    dlon = lon2 - lon1
    dlat = lat2 - lat1
    a = math.sin(dlat/2)**2 + math.cos(lat1) * math.cos(lat2) * math.sin(dlon/2)**2
    c = 2 * math.asin(math.sqrt(a))
    dec_m = EARTH_RADIUS * c
    return dec_m
```

Далее нужна функция, принимающая на вход координаты центра и массив точек. Эта функция будет возвращать количество точек, находящихся внутри окружности в заданном центре, радиусом 2000 м.

После необходимо обойти все возможные точки, начиная с минимальных lat и lon, в каждом шаге добавляя к ним по 0.001 (обход необходимо делать, используя вложенный цикл - для каждого lat каждый lon). Внутри циклов необходимо проверять, сколько точек попадает в окружность с центром в полученных lat и lon. Таким образом можно найти координату центра окружности, в которой будет максимальное количество точек.

Пример программы-решения

Ниже представлено решение на языке Python3

```
# -*- coding: utf-8 -*-
"""
Created on Sun Feb 24 00:33:33 2019
@author: DNS
"""

import math
#import matplotlib.pyplot as plt

import numpy as np

def unit_vector(vector):
    """ Returns the unit vector of the vector.  """
    return vector / np.linalg.norm(vector)

def angle_between(v1, v2):
    """ Returns the angle in degrees between vectors 'v1' and 'v2':
    """
    v1_u = unit_vector(v1)
    v2_u = unit_vector(v2)
    return math.degrees(np.arccos(np.clip(np.dot(v1_u, v2_u), -1.0, 1.0)))

def get_vectors(pA, pB, center):
    AO = [center[0] - pA[0], center[1] - pA[1]]
    BO = [center[0] - pB[0], center[1] - pB[1]]
    return AO, BO


def haversine(coord1, coord2):
    lon1 = coord1[0]
    lon2 = coord2[0]
    lat1 = coord1[1]
    lat2 = coord2[1]
    EARTH_RADIUS = 6371000

    lon1, lat1, lon2, lat2 = map(math.radians, [lon1, lat1, lon2, lat2])
    dlon = lon2 - lon1
    dlat = lat2 - lat1
    a = math.sin(dlat/2)**2 + math.cos(lat1) * math.cos(lat2) * math.sin(dlon/2)**2
    c = 2 * math.asin(math.sqrt(a))
    dec_m = EARTH_RADIUS * c
    return dec_m


def points_in_rad(center_coord, arr, radius = 2000):
    sorted_arr = sorted(arr)

    pointsInRadius = []

    for i in range(0, len(sorted_arr)):
        if(haversine((center_coord), sorted_arr[i]) <= radius):
            pointsInRadius.append(sorted_arr[i])
    return pointsInRadius


```

```
57
58
59  arr =  [(104.2607150000, 52.2509700000),
60          (104.3047200000, 52.2876890000),
61          (104.2963780000, 52.2908390000),
62          (104.2807670000, 52.2834890000),
63          (104.2899250000, 52.2738040000),
64          (104.2899250000, 52.2738040000),
65          (104.2920600000, 52.2732450000),
66          (104.2672810000, 52.2513130000),
67          (104.2921130000, 52.2742630000),
68          (104.2852950000, 52.2773480000),
69          (104.2805530000, 52.2779920000),
70          (104.2992640000, 52.2784380000),
71          (104.2987450000, 52.2785310000),
72          (104.2839000000, 52.2787930000),
73          (104.2839430000, 52.2784510000),
74          (104.2934170000, 52.2985000000),
75          (104.2866900000, 52.2892410000),
76          (104.2831170000, 52.2918110000),
77          (104.2945860000, 52.3006260000),
78          (104.2776020000, 52.2751160000),
79          (104.2884280000, 52.2935860000)]
80
81
82  sorted_arr = sorted(arr)
83  max_points_in_radius = 0
84
85  max_coord_r = ()
86
87
88  max_lon = max(list(zip(*arr))[0])
89
90
91  lon = min(list(zip(*arr))[0])
92  lat = min(list(zip(*arr))[1])
93  center_coord = (lon, lat)
94
95
96  for i in range(0, 1000):
97      if lon - max_lon > 0.1:
98          break
99      lon += 0.001
100     lat = 52.23
101     points_in_radius = 0
102     ps = 0
103     for j in range(0, 1000):
104         lat += 0.001
105         center_coord = (lon, lat)
106
107         if center_coord in arr:
108             pass
109         else:
110             points_in_radius = len(points_in_rad(center_coord, arr))
111
112         if points_in_radius > max_points_in_radius:
113             max_points_in_radius = points_in_radius
114             max_coord_r = center_coord
115
116
```

```
117 max_lon = max(list(zip(*arr))[0])
118 max_lat = max(list(zip(*arr))[1])
119
120 print(max_coord_r)
121 print(max_points_in_radius)
```

Задача 3.5.7. (17 баллов)

Шаг 1. Если вы выполнили предыдущее задание, пропустите этот шаг. Иначе, скачайте файл, содержащий описание зданий, являющихся достопримечательностями города Иркутска. По описанию отыщите эти здания в сети интернет и определите их географические координаты. Заполните таблицу с полями:

1. Координаты lon, lat
2. Описание
3. Название

Вам необходимо найти координаты ЦЕНТРА каждого здания, мы рекомендуем для этого воспользоваться сервисом 2ГИС (ВАЖНО!). Заполненная Вами таблица будет являться вспомогательным инструментом в следующих этапах задания.

Шаг 2. Правильное решение шага задачи оценивается в 17 баллов. Вам необходимо определить координаты (lon, lat) расположения пользователя AR-браузера, находясь в котором он сможет видеть максимальное количество достопримечательностей из файла задания, при угле обзора 90 градусов и радиусе до 2 км.

Для решения этой задачи мы рекомендуем воспользоваться одним из браузеров дополненной реальности: создать гео-слой с отображением на нем необходимых POI (points of interest, координат достопримечательностей, определенных вами в предыдущем шаге). Урок по созданию простого гео-слоя расположен по ссылке (`https://www.layar.com/documentation/browser/tutorials-tools/create-simple-geo-location-layer/`). Также вам может пригодиться мобильное приложение для подмены геопозиции вашего мобильного устройства.

В качестве ответа необходимо прикрепить долготу и широту через пробел.

Пример: 37.531276 55.702651

Решение

Для решения этой задачи можно взять за основу алгоритм перебора из предыдущей задачи. Но, в этом случае, необходимо также проверять каждую найденную точку - взять от окружности, центром которой является эта точка, сектор в 90 градусов и посчитать количество достопримечательностей, попадающих в этот сектор. После чего развернуть сектор на 1 градус и повторить подсчет достопримечательностей уже в новом секторе. Ответом будет являться та точка, которая является центром окружности, содержащей сектор с наибольшим количеством достопримечательностей в нем. Стоит отметить, что тут приведено не оптимальное решение, занимающее значительное процессорное время. Это решение можно оптимизировать, используя различные алгоритмы.

Пример программы-решения

Ниже представлено решение на языке Python3

```python
import math
# import matplotlib.pyplot as plt

import numpy as np


def haversine(coord1, coord2):
    lon1 = coord1[0]
    lon2 = coord2[0]
    lat1 = coord1[1]
    lat2 = coord2[1]
    EARTH_RADIUS = 6371000

    lon1, lat1, lon2, lat2 = map(math.radians, [lon1, lat1, lon2, lat2])
    dlon = lon2 - lon1
    dlat = lat2 - lat1
    a = math.sin(dlat / 2) ** 2 + math.cos(lat1)
    * math.cos(lat2) * math.sin(dlon / 2) ** 2
    c = 2 * math.asin(math.sqrt(a))
    dec_m = EARTH_RADIUS * c
    return dec_m


def points_in_rad(center_coord, arr, radius=2000):
    """
    Получение всех точек в определенном радиусе от центральной
    :param center_coord: центральная точка
    :param arr: массив точек
    :param radius: радиус в метрах
    :return: список точек
    """
    sorted_arr = sorted(arr)

    pointsInRadius = []

    for i in range(0, len(sorted_arr)):
        if haversine((center_coord), sorted_arr[i]) <= radius:
            pointsInRadius.append(sorted_arr[i])
    return pointsInRadius


def angle_between_points(left, center, right):
    """
    Расчет угла между тремя точками на поверхности Земли в радианах
    """
    a = haversine(center, right)
    b = haversine(center, left)
    c = haversine(left, right)
    return math.acos((a ** 2 + b ** 2 - c ** 2) / (2 * a * b))


def destination_point(center_coord, distance, angle):
    """
    Вычисляет точку на поверхности Земли, удаленную от определенной на расстояние
    :param center_coord: центральная точка, относительно которой мы ищем
    :param distance: расстояние в метрах
    :param angle: угол в градусах
    :return: точку, удаленную на distance с углом angle
    """
```

```
60      distance /= 6371000
61      angle = math.radians(angle)
62      lng1, lat1 = map(math.radians, center_coord)
63      lat2 = math.asin(math.sin(lat1) * math.cos(distance) +
64                       math.cos(lat1) * math.sin(distance) * math.cos(angle))
65      lng2 = lng1 + math.atan2(math.sin(angle) * math.sin(distance) * math.cos(lat1),
66                               math.cos(distance) - math.sin(lat1) * math.sin(lat2))
67      return math.degrees(lng2), math.degrees(lat2)
68
69
70  def points_in_sector(center_coord, arr, radius=2000, angle=90, step=1):
71      """
72      Получаем максимальное количество точек, находящиеся в секторе
73      :param center_coord: центральная точка
74      :param arr: массив точек
75      :param radius: радиус в метрах
76      :param angle: угол в градусах
77      :param step: шаг для определения сектора
78      :return: список точек
79      """
80      points = points_in_rad(center_coord, arr, radius)
81
82      max_points = []
83
84      angle = math.radians(angle)
85
86      start_points = []
87      start_angle = 0.0
88      while start_angle < 360:
89          start_points.append(destination_point(center_coord, radius, start_angle))
90          start_angle += step
91
92      for start_point in start_points:
93          tmp_points = []
94          for point in points:
95              if angle_between_points(start_point, center_coord, point) <= angle / 2:
96                  tmp_points.append(point)
97          if len(tmp_points) > len(max_points):
98              max_points = tmp_points
99      # print('LOOOOOOOOL')
100     return max_points
101
102
103 arr = [(104.2607150000, 52.2509700000),
104        (104.3047200000, 52.2876890000),
105        (104.2963780000, 52.2908390000),
106        (104.2807670000, 52.2834890000),
107        (104.2899250000, 52.2738040000),
108        (104.2899250000, 52.2738040000),
109        (104.2920600000, 52.2732450000),
110        (104.2672810000, 52.2513130000),
111        (104.2921130000, 52.2742630000),
112        (104.2852950000, 52.2773480000),
113        (104.2805530000, 52.2779920000),
114        (104.2992640000, 52.2784380000),
115        (104.2987450000, 52.2785310000),
116        (104.2839000000, 52.2787930000),
117        (104.2839430000, 52.2784510000),
118        (104.2934170000, 52.2985000000),
119        (104.2866900000, 52.2892410000),
```

```
        (104.2831170000, 52.2918110000),
        (104.2945860000, 52.3006260000),
        (104.2776020000, 52.2751160000),
        (104.2884280000, 52.2935860000)]

sorted_arr = sorted(arr)
max_points_in_radius = 0

max_coord_r = ()

max_lon = max(list(zip(*arr))[0])

lon = min(list(zip(*arr))[0])
lat = min(list(zip(*arr))[1])
center_coord = (lon, lat)

for i in range(0, 1000):
    if lon - max_lon > 0.1:
        break
    lon += 0.01
    lat = 52.23
    points_in_radius = 0
    ps = 0
    for j in range(0, 1000):
        lat += 0.01
        center_coord = (lon, lat)

        if center_coord in arr:
            pass
        else:
            points_in_radius = len(points_in_sector(center_coord, arr))

        if points_in_radius > max_points_in_radius:
            max_points_in_radius = points_in_radius
            max_coord_r = center_coord

max_lon = max(list(zip(*arr))[0])
max_lat = max(list(zip(*arr))[1])

print(max_coord_r)
print(max_points_in_radius)

# Проверка правильности работы алгоритма расположения точек по окружности
# start_points = []
# start_angle = 0.0
# while start_angle < 360:
#     start_points.append(
#    destination_point((104.2805530000, 52.2779920000), 2000, start_angle))
#     start_angle += 10
#
# print('\n'.join(map(str, map(lambda p: ','.join(map(str, p[::-1])), start_points))))
```

3.6. Основы 3D-моделирования

Задача 3.6.1. (20 баллов)

Вам необходимо прорешать задачи курса "Основы инженерного 3D-моделирования", доступного по ссылке: `https://stepik.org/course/47687/syllabus`.

Система оценки

Баллы будут начислены после закрытия модуля пропорционально баллам, набранным в указанном курсе, то есть если участник команды решит 70%, будет выставлено 14 баллов.

3.7. «Задача Рэдрика Шухарта»

Рэдрик Шухарт* возглавляет отдел по изучению Зон Посещения Института (`https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D0%BA%D0%BD%D0%B8%D0%BA_%D0%BD%D0%B0_%D0%BE%D0%B1%D0%BE%D1%87%D0%B8%D0%BD%D0%B5`) внеземных культур. Теперь подвергать людей риску нет никакой необходимости, вместо сталкеров на территорию контакта с внеземной цивилизацией отправляются мобильные роботы. Они передают в Институт всю необходимую информацию об аномалиях. Информация отображается на плоской цифровой сетке-табло, имитирующей поверхность одного из многочисленных квадратов Зоны. На краях сетки установлены маркеры, отмечающие границы изучаемой территории. Красным внутри сетки отмечается квадрат, в котором находится «пустышка»** с содержимым (сущностью внутри), синий квадрат – места нахождения пустышки без содержания, зеленый – отделившаяся от пустышек сущность - виртуальный объект, «призрак», плавающий над поверхностью Зоны. Камера фиксирует состояние табло в течение дня в виде серии снимков, которые передаются в отдел для анализа.

Стажерам-исследователям, принятым в отдел, необходимо:

разработать программу, позволяющую распознать снимки и вывести в файл координаты перемещения объектов внутри сетки за день; создать AR-приложение, позволяющее отобразить виртуальную сетку и объекты на ней в произвольном месте и масштабе За полное и верное решение каждой части задачи участник получают 60 баллов. Кроме того, оценивается частичное решение каждой задачи: первой в зависимости от процента верно распознанных изображений, второй – от части реализованной демонстрационной модели. Описание частей – в следующих шагах курса.

Примечания:

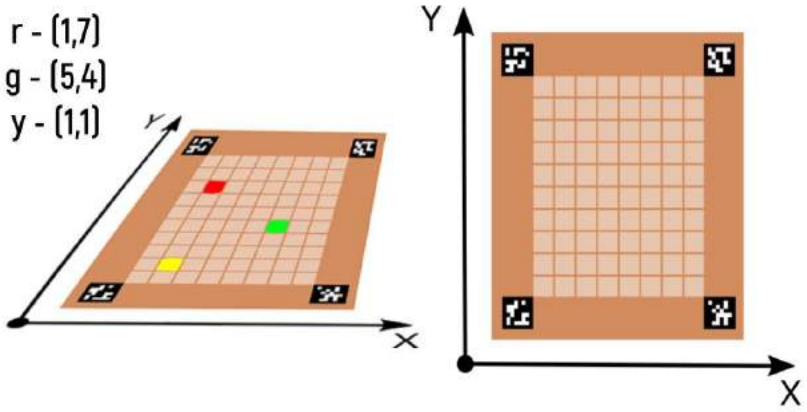
*Рэдрик Шухарт – сталкер, главный герой фантастической повести Аркадия и Бориса Стругацких «Пикник на обочине» (`https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D0%BA%D0%BD%D0%B8%D0%BA_%D0%BD%D0%B0_%D0%BE%D0%B1%D0%BE%D1%87%D0%B8%D0%BD%D0%B5`).

** «Пустышка» представляет собой два диска, между которыми находится пространство, то есть механически они не связаны. При этом диски нельзя сдвинуть с места относительно друг друга. В наши дни стало известно, что внутри пустышки

может находиться сущность – некий виртуальный объект, наблюдать который можно через специальные устройства

Задача 3.7.1. Часть 1 (60 баллов)

В папке dataset (архив на Google Диске `https://goo.gl/whS5jm`) находятся фотографии электронного табло, отображающего состояние одного из территориальных квадратов Зоны в течение дня. Все они имеют название cadrN.png где N - номер полученного кадра.



Необходимо обработать все фотографии в соответствии с их очередностью и в результате получить файл, каждая строчка которого соответствует положению сначала красного, зеленого и желтого объекта на каждом отдельном фото.

Для разделения кадров использовать — ——-"(10 знаков "минус").

Т.е. для примера одному кадру, будет соответствовать файл:

r - (7,5)

g - (4,3)

y - (7,6)

———-

А для двух фотографий, файл будет выглядеть так (и так далее):

r - (7,5)

g - (4,3)

y - (7,6)

———-

r - (8,6)

g - (3,3)

y - (3,0)

———-

Пример выходного файла для 1561 фотографии - aruko-markers-output-test.txt (`https://github.com/ipetrushin/NTI-Contest/blob/master/aruko-markers-output-test.txt`).

РЕКОМЕНДАЦИИ:

В файле random_point.py (`https://github.com/ipetrushin/NTI-Contest/blob/master/random_point.py`) пример того, как формируется выходной файл на языке python.

Для упрощения работы возможно использование библиотеки OpenCV (`https://docs.opencv.org`) и конкретного модуля в ней cv2.aruco (`https://docs.opencv.org/3.1.0/d5/dae/tutorial_aruco_detection.html`).

В первую очередь нужно детектировать сетку и преобразовать ее к нормальной форме (к прямоугольнику) с помощью аффинных преобразований. Для распознавания цвета объектов рекомендуется использование цветовой модели HSV.

Решение

1. На фотографии находятся ARuko маркеры.
2. По внутренним углам маркеров находится сетка.
3. В зависимости от положения маркеров сетка разворачивается таким образом, чтобы начало координат было в нижнем левом углу.
4. Получаются все цветные клетки сетки и определятся их цвета.

Рассмотрим по отдельности каждый пункт.

Для того, чтобы не писать самим алгоритмы распознования нужно установить библиотеку OpenCV и отдельный модуль из нее для распознавания маркеров aruko.

Далее подключаем эти библиотеки.

```
import cv2
from cv2 import aruco
```

Главная функция программы

```
def detect(filename):
    # код, который находит маркеры
    frame =cv2.imread(filename)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    aruco_dict = aruco.Dictionary_get(aruco.DICT_6X6_250)
    parameters =  aruco.DetectorParameters_create()
    aruko_paraps = aruco.detectMarkers(gray, aruco_dict,
                                                    parameters=parameters)


    frame =cv2.imread(filename)
    #получение сетки между маркерами
    warped_img = warp_by_4markers(frame, aruko_paraps)

    #получение объектов (цветных клеток) на сетке
    objects = get_objects(warped_img)

    #подписываем объекты на сетке
    for obj in objects:
        font = cv2.FONT_HERSHEY_SIMPLEX
        cv2.putText(warped_img, obj[0], obj[1], font, 1,(255,255,255),2,cv2.LINE_AA)
    plt.subplot(121),plt.imshow(warped_img),plt.title('Input')
```

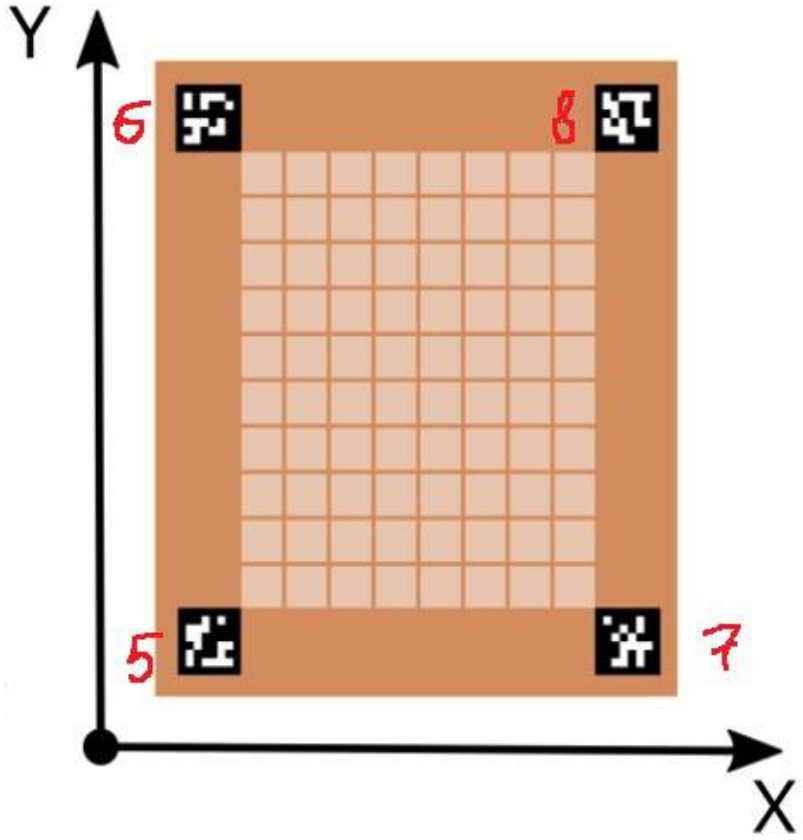
```
23
24      return objects
```

Получение сетки между маркерами

Алгоритм работы данного блока:

1. Находим внутренние углы маркеров.
2. Для этого нужно найти у каждого маркера те углы, которые ближе всего к центру. Далее сортируем их по правильному порядку следования Aruko маркеров, как на картинке.

С помощью афинных преобразований преобразуем часть изображения между внутренними углами к прямому.



Получение внутренних углов

```
1  def find_field_edges(frame, aruko_params):
2      center = (frame.shape[1]//2, frame.shape[0]//2)
3      corners, ids, rejectedImgPoints = aruko_params
4
5      ids = [id_[0] for id_ in ids]
6      ids_coros = dict(zip(ids, corners))
7      #углы к которым мы преобразуем изображение
8      warped_size = [[0,0], [0, 10*30-1], [30*8 - 1, 10*30-1], [30*8-1, 0]]
9      # очередь Aruko маркеров, для того чтобы в дальнейшем правильно развернуть сетку.
10     marker_id_order = [5,6,8,7]
11
```

```
    coords_towarp = []
    field_edges = []
    frame = {}
    bag = 0
    for k, i in enumerate(marker_id_order):
        try:
            smcorner = sorted(ids_coros[i][0], key = lambda coor: dist(coor, center))
            frame[i] = (smcorner[0][0],smcorner[0][1])
            coords_towarp.append(warped_size[k])
            field_edges.append(list(frame[i]))
        except:
            bag = 1
            continue
    return (len(list(frame.keys())), field_edges, coords_towarp)
```

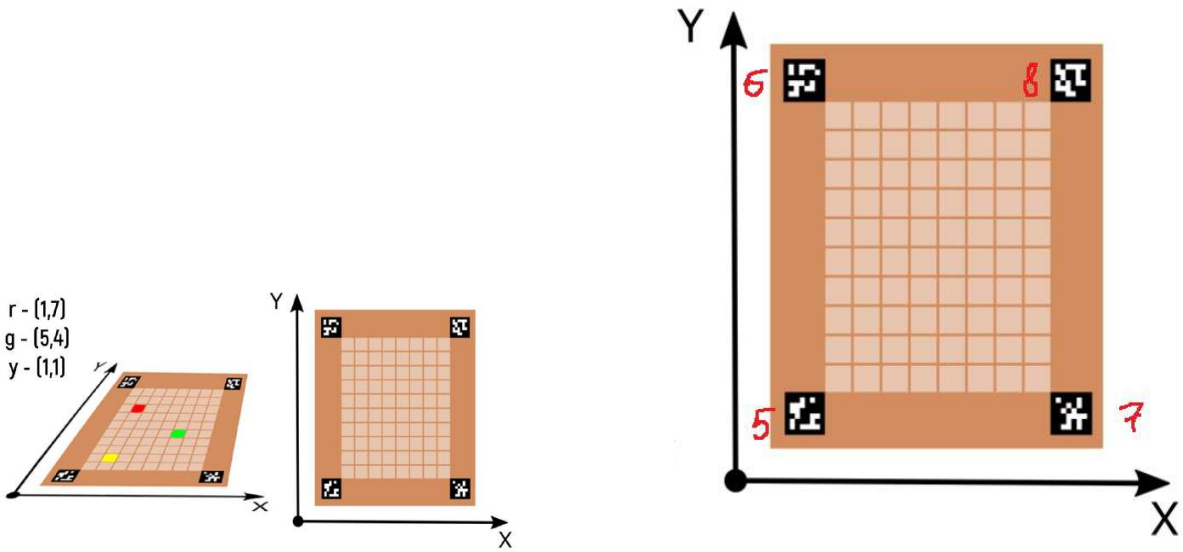
Код для афинных преобразований

```
def warp_by_4markers(frame, aruko_params):
    angle_count, field_edges, coords_towarp = find_field_edges(frame, aruko_params)
    rows, cols = (10*30-1, 30*8-1)

    if angle_count == 4:
        coords_towarp = np.float32([coords_towarp[:]])
        field_edges = np.float32([field_edges[:]])
        M = cv2.getPerspectiveTransform(field_edges, coords_towarp)
        warped_img = cv2.warpPerspective(frame,M,(cols,rows))
        return warped_img

    return warped_img
```

Результат работы кода:

Рисунок 1 - оригинальное фото

Рисунок 2 - вырезанная и преобразованная сетка



Получение объектов (цвеных клеток) на сетке

Для начала нам нужно просто определить нахождение всех цветных квадратиков на сектке. Для этого можно бинаризировать изображение по яркости (объекты

на сетке светлее, чем окружающие их клетки). Чтобы определить степень яркости переведем изображение из модели RGB в модель HSV.

Чтобы определить какого цвета каждый объект, отсортируем их по параметру H(цветовому тону). Соответсвенно, вначале списка тогда окажется красный, потом желтый затем зеленый.

Алгоритм работы данного блока

1. Переводим изображение в HSV
2. Бинаризируем изображения по яркости и насыщенности. Применяем размытие гаусса, чтобы убрать лишние проблески.
3. Для того, чтобы отделить объекты между собой, находим на изображении контуры.
4. Находим центральный пиксель каждого контура, определяем его цвет в модели HSV. Формируем список по типу: [(значение пикселя в HSV, X, Y) , (значение пикселя в HSV, X, Y)]
5. Сортируем по параметру H.
6. Переводим координаты объектов из координат на изображении к координатам на сетке.

Получение координат объектов на изображении

```
def color_coords(img):
    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

    #Бинаризаця
    lower_blue = np.array([0,0,200])
    upper_blue = np.array([180,255,255])

    mask = cv2.inRange(hsv, lower_blue, upper_blue)
    kernel = np.ones((11,11),np.uint8)
    mask = cv2.erode(mask,kernel,iterations = 1)
    res = cv2.bitwise_and(img, img, mask= mask)
    res = cv2.GaussianBlur(res,(7,7),0)

    gray = cv2.cvtColor(res, cv2.COLOR_BGR2GRAY)
    ret,thresh = cv2.threshold(gray,127,255,0)
    #получение контуров
    im2, contours, hierarchy = cv2.findContours(thresh,cv2.RETR_TREE,
    cv2.CHAIN_APPROX_SIMPLE)

    centers = []
    clr = ["r","y","g"]
    for cnt in contours:
        #желтый , зеленый, красный
        M = cv2.moments(cnt)
        cX = int(M["m10"] / M["m00"])
        cY= int(M["m01"] / M["m00"])
        #формирование списка центров
        centers.append((cX,cY, hsv[cY, cX][0]))

    #Сортировка по значению Ние
    centers = sorted(centers, key = lambda lst: lst[2])
    colors = [(center[0],center[1], color) for center, color in zip(centers, clr)]
    return colors
```

Преобразование к координатам на сетке

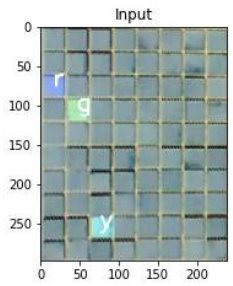
```
def get_objects(img):
    field_width, field_height = img.shape[:2]
    cell_shape = field_width/10,  field_height/8

    colors = color_coords(img)
    objects = []
    for color in colors:
        img_coor = color[:-1]
        field_coor = change_coord_sys(img_coor, cell_shape)
        objects.append([color[-1], img_coor, field_coor])
```

Функция преобразования для одной клетки

```
def change_coord_sys(obj, cell_shape):
    x = obj[0]//cell_shape[0]
    y = obj[1]//cell_shape[1]
    return (x,y)
```

Результат

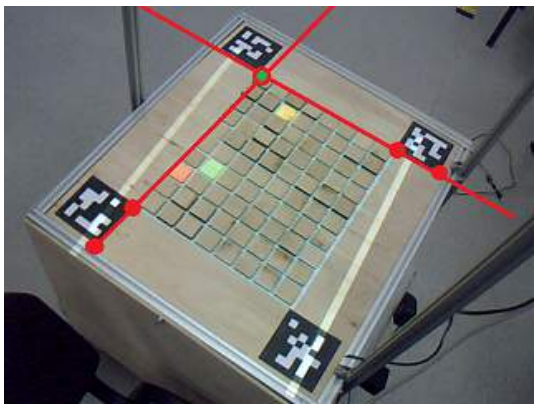
После применения главной функции detect, будет получен следующий результат:

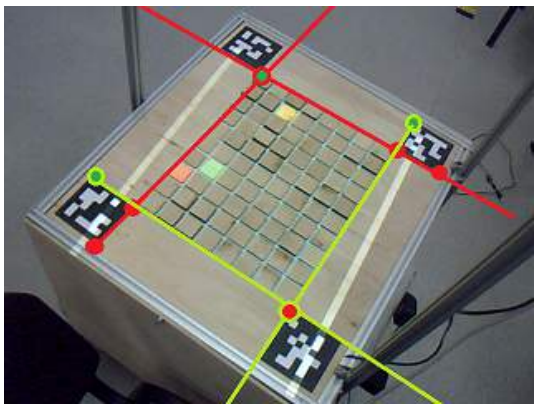


Примечание: на данный момент для того, чтобы вырезать сетку нужно находить все 4 маркера. Однако на чати фотографий всего 3 маркера из 4х. В такой ситуации можно восстановить сетку по следующему алгоритму:

Алгоритм:

1. Находим диагональные маркеры (к примеру, верхний правый и нижний левый).
2. Строим прямую по нижним координатам верхнего маркера и внутренним правым нижнего. Пересечение двух прямых это верхний угол сетки (См. рис1).
3. Соответственные действия производим для нахождения нижнего угла сетки (См. рис2) .
4. По полученным углам восстанавливаем сетку.

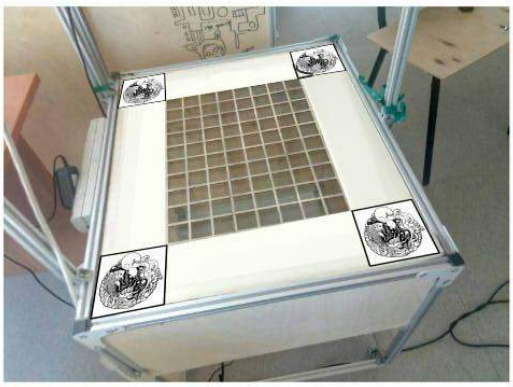


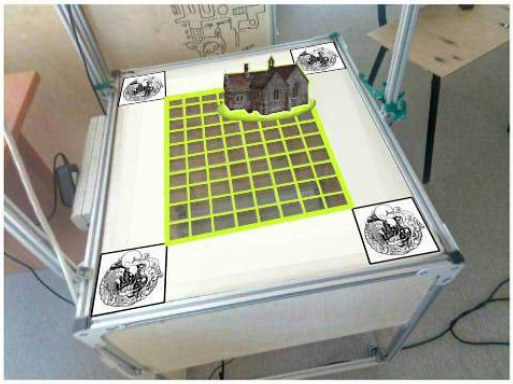


Задача 3.7.2. Часть 2 (60 баллов)

1. Создайте AR-приложение для мобильного устройства, которое позволяет высвечивать линии градуировки сетки, расположенной между четырьмя маркерами при наведении экрана смартфона на них. Размер сетки: количество ячеек по длине и ширине задается в приложении, в соответствии с этим, автоматически, определяется размер одной ячейки. Работающее приложение оценивается в 30 баллов. Если в приложении реализована статическая сетка (не растягивается при изменении положения маркеров), то работа оценивается в 15 баллов.
2. Реализуйте в приложении, созданного в предыдущем шаге, возможность визуализировать виртуальные объекты, появляющиеся над поверхностью эмуляции сетки, при вводе соответствующих координат в диалоговое окно или текстовые поля приложения (+15 баллов).
3. Модифицируйте приложение таким образом, чтобы координаты добавляемого виртуального объекта на сетку задавались не в окне приложения, а касанием соответствующей виртуальной ячейке на экране смартфона, либо наведением фокуса приложения (крестика/прицела) на соответствующую клетку, над которой будет появляться виртуальный объект (+15 баллов).

На рисунке изображена сетка и результат работы приложения.





Необходимо предоставить приложение, выполненное для мобильного устройства, работающего под управлением ОС Android, маркеры (используемые для обозначения углов сетки) и короткий видеоролик, демонстрирующий функциональные возможности AR-приложения в работе.

В ответ на задание приложите ссылку на архив с приложением и остальными материалами.

Оценка за приложение выставляется экспертом вручную на следующем шаге.

РЕКОМЕНДАЦИИ:

Есть несколько подходов к решению этой части задачи Института внеземных культур. Один из них совпадает с предыдущей задачей, только вместо маркеров aruco будут следует использовать обычные маркеры. Если вы используете библиотеку Vuforia, то она позволяет получить "в коде"описание положения маркера (угол наклона, координаты).

Отображать над сеткой в соответствующей координате можно, как плоские, так и трехмерные модели объектов, как с анимацией, так и без, как по одному объекту, так и все три: «пустышка» с содержимым, «пустышка» без содержимого, объект-призрак. Однако стоит учитывать, что все это влияет на формирование итогового балла по данному шагу задачи.

Решение

Данную задачу можно свести к задаче генерации и масштабированию объектов. Для решения можно воспользоваться следующим подходом:

1. Создать единицу сетки “клетку” из четырех кубов. И объединить все объекты с помощью одного EmptyObject, чтобы мы могли работать с ними как с одним объектом. Переименуем EmptyObject в Cell
2. Создадим класс GridMaker. Он будет генерировать поле заданного размера клеток из объекта Cell.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;


public class GridMaker : MonoBehaviour {

        public GameObject cell;

        static float widthf;
        static float heightf;

        public InputField xcount;
        public InputField ycount;
        // Use this for initialization
        void Start () {

        }

        // Update is called once per frame
        void Update () {

        }

    public void reload()
    {
        Application.LoadLevel("main");
```

```
29      }
30          public void clear(){
31
32                  GameObject grid = GameObject.Find("Grid");
33                  grid.transform.localScale = new Vector3(1,1,1);
34
35                  foreach (Transform child in grid.transform)
36                          {
37                                  GameObject.Destroy(child.gameObject);
38                            //child is your child transform
39                          }
40
41          }
42
43          public void generate(){
44
45                  clear();
46                  string width_str = xcount.text;
47                  string height_str = ycount.text;
48
49
50                  int width = int.Parse(width_str);
51                  int height = int.Parse(height_str);
52
53                  widthf = 5.0f;
54                  heightf = 5.0f;
55
56                  float genx = 0;
57                  float geny = 0;
58                  float genz = 0;
59
60                  float x_step = cell.transform.localScale.x;
61                  float y_step = cell.transform.localScale.z;
62
63                  for (int j = 0; j < height; j++)
64                  {
65                   genx = 0;
66                   for (int i = 0; i < width; i++)
67                   {
68                    Vector3 pos = new Vector3(genx, genz, geny);
69                    GameObject new_cell = Instantiate (cell, pos,
70                           cell.transform.rotation);
71                    new_cell.name = "cell" + i.ToString() + j.ToString();
72                    new_cell.SetActive(true);
73                    new_cell.transform.parent = GameObject.Find("Grid").transform;
74                    new_cell.transform.localPosition = pos;
75                    genx += x_step;
76                   }
77                   geny += y_step;
78
79                  }
80                  GridTransform.x_count = width;
81                  GridTransform.y_count = height;
82                  GridTransform.transform = true;
83                  SynFromJson.colorise = true;
84          }
85  }
```

Полученное сгенерированное поле помещается под единый родительский объект Grid (нужно создать заранее). Это делается для того, чтобы сетку было

удобнее масштабировать под маркеры.

3. Далее нам нужно, чтобы поле автоматически натягивалось между двумя маркерами. Для этого будет использован класс GridTransform. Он получает координаты внутренних углов маркеров. Вычисляет позицию куда нужно переместить поле, а также из расстояния между маркерами определяет, как нужно растянуть сетку.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class GridTransform : MonoBehaviour {

        public GameObject top_left;
        //public GameObject top_right;

        //public GameObject bottom_left;
        public GameObject bottom_right;

        public GameObject grid;
        public static bool transform;
        public static int x_count;
        public static int y_count;


        // Use this for initialization
        void Start () {
                transform = false;

        }

        // Update is called once per frame
        void Update () {

                if (transform == true)
                {
                        float top_left_x = top_left.transform.position.x;
                        float top_left_y = top_left.transform.position.y;
                        float top_left_z = top_left.transform.position.z;

                        /*float top_right_x = top_right.transform.position.x;
                        float top_right_y = top_right.transform.position.y;
                        float top_right_z = top_right.transform.position.z;

                        float bottom_left_x = bottom_left.transform.position.x;
                        float bottom_left_y = bottom_left.transform.position.y;
                        float bottom_left_z = bottom_left.transform.position.z;*/

                        float bottom_right_x = bottom_right.transform.position.x;
                        float bottom_right_y = bottom_right.transform.position.y;
                        float bottom_right_z = bottom_right.transform.position.z;

                        //Vector3 pos = top_left.transform.position;
                        //grid.transform.position = pos;

                        float scale_x = Mathf.Abs(top_left_x - bottom_right_x);
                        float scale_y = Mathf.Abs(top_left_y - bottom_right_y);
                        float scale_z = Mathf.Abs(top_left_z - bottom_right_z);

```

```
53
54                          float tx = scale_x/x_count;
55                          float tz = scale_z/y_count;
56
57
58                          grid.transform.localScale = new Vector3(tx, 1, tz);
59
60                          float pos_x = ((top_left_x + bottom_right_x)/2);
61                           pos_x -= scale_x/2 - (0.5f*tx);
62
63                          float pos_y = ((top_left_y + bottom_right_y) / 2);
64                          // pos_y -= scale_y;
65
66                          float pos_z = ((top_left_z + bottom_right_z)/2);
67                           pos_z -= scale_z/2 - (0.5f*tz);
68
69                          // Vector3 pos = bottom_left.transform.position;
70                           grid.transform.localPosition = new Vector3(pos_x,
71                            pos_y, pos_z);
72
73
74                  /*
75                  grid.transform.rotation = top_left.transform.rotation;
76
77
78
79                  grid.transform.localScale = new Vector3(scale_x, scale_y,
80                  scale_z);
81                  //Vector3 rot = top_left.transform.rotation;*/
82          }
83  }
84  }
```

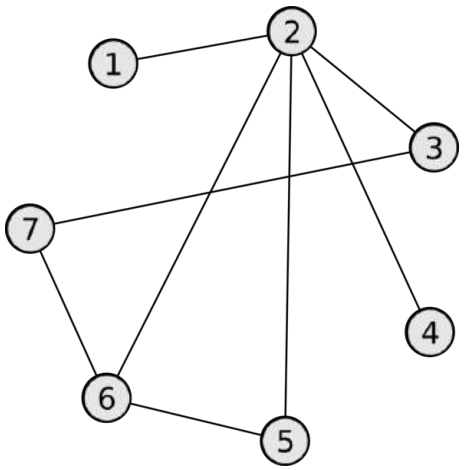
Все это делается в методе Update для того чтобы сетка автоматически обновлялась и подстраивалась при перемещении маркеров.

4. Для того, чтобы по нажатию на клетку появлялись объекты, можно в объект Cell добавить элемент Cavas'a button. Нужно реализовать класс Changer в котором прописать метод change. Его добавить на нижнюю часть собранной клетки. В onClick у кнопки нужно указать ссылку на метод change нужного объекта. В методе реализуется генерация нужного 3D объекта с помощью метода Instance

3.8. Построение маршрутов на местности

Задача 3.8.1. (2 балла)

Что из приведенного ниже является простым путем в графе, изображенном на рисунке:



1. 7-6-5-2-6
2. 2-3-7-6-2
3. 1-2-3-4-5
4. 3-7-6-2-4
5. 2-6-5-3-7
6. 1-2-3-7-6

Пояснения к ответу

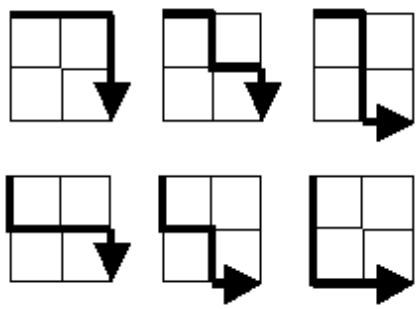
Простой путь - такой, который не проходит дважды через одну вершину, например 3-7-6-2-4 или 1-2-3-7-6.

Ответ: 4, 6.

Задача 3.8.2. (6 баллов)

На рисунке изображена сетка 2x2 клетки. Если двигаться от левого верхнего угла сетки в правый нижний, перемещаясь только вниз или вправо, то существует ровно 6 различных маршрутов (см. рисунок).

Определите число различных маршрутов для сетки размером 20×20 клеток.



Решение

Из условия задачи следует, что длина любого пути одинакова и равна $2N$, где N - число клеток. Обозначим конкретный путь буквами В (вниз) и П (право), число ходов вниз и вправо одинаково. Например так выглядят обозначения путей, приведённых на рисунке: ППВВ, ПВПВ, ПВВП, ВППВ, ВПВП, ВВПП. Рассчитаем число возможных перестановок (всего 4!), с учётом, что буквы В и П повторяются $\frac{4!}{2!\cdot 2!} = \frac{1\cdot 2\cdot 3\cdot 4}{1\cdot 2\cdot 1\cdot 2} = 2\cdot 3 = 6$. Для поля $20\cdot 20$, соответственно, число путей $= \frac{40!}{20!\cdot 20!} = 137846528820$.

Ответ: 137846528820.

Задача 3.8.3. (10 баллов)

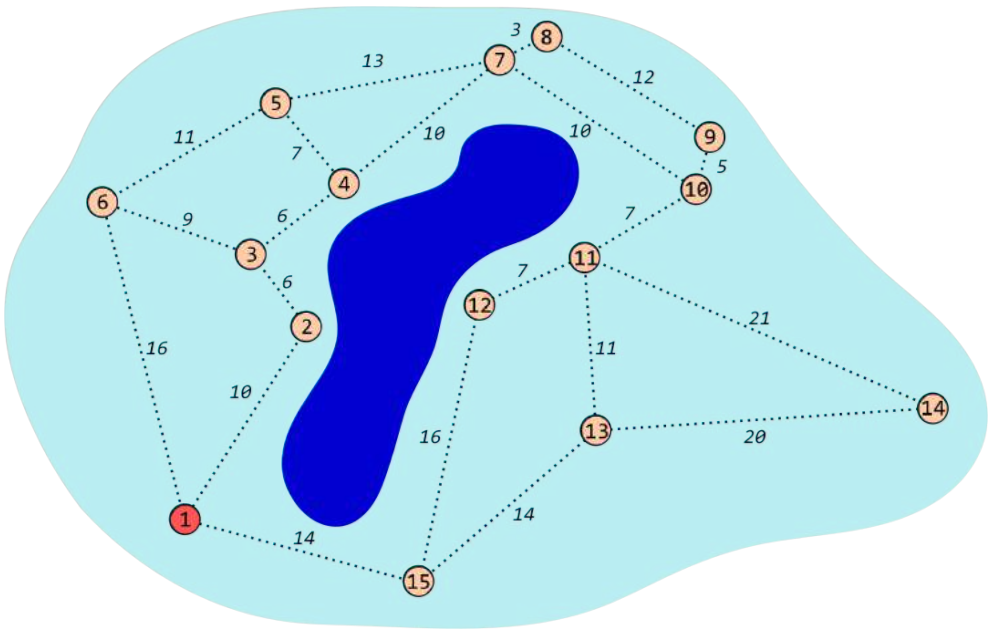
С самого своего появления Зона притягивала к себе авантюристов - сталкеров. В Зоне на некоторых локациях располагаются артефакты. Каждый сталкер (или команда) собирает только определённые артефакты, которые нужны заказчику.

Постепенно совместными усилиями сталкеры составили карту всех локаций в Зоне, пускай и неточную, но сильно упростившую ориентацию на местности. Карта одинаковая для всех задач и представлена в виде графа (`https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)#%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_%D1%80%D1%91%D0%B1%D0%B5%D1%80`), заданного числом вершин (от 1 до N) и набором рёбер (указывается также длина ребра - время пути между локациями в минутах).

Для удобства мы приводим схематическую иллюстрацию карты и её текстовое представление (в тексте ниже первая строка - число вершин и рёбер; далее следуют номера соединяемых вершин и вес каждого ребра)

Используйте эту карту для выполнения заданий в следующих шагах этого модуля.

Для поиска кратчайшего пути вам может потребоваться алгоритм Дейкстры или поиск по графу в ширину.



15 21
1 2 10
1 15 14
1 6 16
2 3 6
3 6 9
3 4 6
4 5 7
4 7 10
5 6 11
5 7 13
7 8 3
7 10 10
8 9 12
9 10 5
10 11 7
11 12 7
11 13 11
11 14 21
12 15 16
13 14 20
13 15 14

Для заданной карты необходимо проложить кратчайший маршрут от данной локации до всех остальных. Программа получает на вход одно число от 1 до 15. В качестве результата сообщите кратчайший путь до каждой локации (15 чисел в одну строку через пробел, путь до локации старта = 0).

Стандартный ввод
1

Стандартный вывод
0 10 16 22 27 16 32 35 47 42 37 30 28 48 14

Пояснения к ответу

Возможно несколько способов решения задачи, мы будем использовать рекурсивный алгоритм. Выберем начальную вершину (из входных данных), например 1-ю. Создадим линейный массив размером N (число вершин), в котором будем хранить длину кратчайшего пути от начальной вершины до всех остальных. В ячейку, соответствующую начальной вершине запишем число 0, остальные заполним числом -1, чтобы обозначить, что до остальных вершин длина пути неизвестна. Число -1 выбрано, так как длина не может быть отрицательной.

По исходным данным (карта дана во введении) построим матрицу смежности (`https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0_%D1%81%D0%BC%D0%B5%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D0%B8`).

Используя рекурсию переходим от начальной вершины к тем, которые связаны с ней, отмечая для этих вершин длину пути = 1. Аналогичным образом делаем следующий шаг, но в массив записывается длина пути = 2. Важно на каждом шаге сравнивать длину пути до данной вершины, которую мы получили рекурсивным обходом в данный момент и длину, записанную в массиве. Стоит делать шаг только в том случае, если в массиве хранится длина -1 (то есть по этой вершине ещё не проходили) или бОльшее значение, чем получили рекурсивным обходом.

Когда все вызовы рекурсий завершатся, в массиве получим длины кратчайших путей от начальной вершины до всех остальных.

Задача 3.8.4. (12 баллов)

Для заданного графа (см. описание карты) каждому из трёх сталкеров известны номера локаций на карте, которые содержат нужные артефакты. Артефакты необходимо собирать в том порядке, каком они перечислены. Вход и выход из Зоны через локацию № 1.

Определите, какой из сталкеров сможет собрать артефакты за наименьшее время.

Формат входных данных

Три строки с номерами локаций с артефактами

Формат выходных данных

Номер строки (одного из сталкеров) и общее время прохода его по маршруту

Стандартный ввод
5 8 9 2 9 13 6 11 15

Стандартный вывод
3 94

Пояснения к ответу

Используя алгоритм Дейкстры (https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%94%D0%B5%D0%B9%D0%BA%D1%81%D1%82%D1%80%D1%8B) или описанный в решении задачи 3 рекурсивный обход графа, построим участки пути от точки старта (локации 1) через необходимые локации обратно до точки старта. Сложив длины каждого участка получим общую длину пути для каждого сталкера. Сравнив пути выбираем кратчайший и выводим на экран.

Задача 3.8.5. (15 баллов)

Для заданного графа (см. описание карты) каждому из трёх сталкеров известны номера локаций на карте, которые содержат нужные артефакты. Артефакты необходимо собирать в любом порядке (в отличие от предыдущего задания). Вход и выход из Зоны через локацию № 1.

Определите, какой из сталкеров сможет собрать артефакты за наименьшее время.

Формат входных данных

Три строки с номерами локаций с артефактами

Формат выходных данных

Номер строки (одного из сталкеров) и общее время прохода его по маршруту.

Стандартный ввод
4 6 11 10 12 14 5 7 8
Стандартный вывод
3 78

Пояснения к ответу

Отличие условий этой задачи от предыдущей в том, что локации с артефактами можно обходить в любом порядке, значит нужно определить длину пути для каждой из перестановок. Для трёх локаций существует 3! = 6 перестановок: (5 7 8) (5 8 7) (7 5 8) (7 8 5) (8 5 7) (8 7 5). Дальнейшие этапы решения аналогичны задаче 4:

Используя алгоритм Дейкстры (https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%94%D0%B5%D0%B9%D0%BA%D1%81%D1%82%D1%80%D1%8B) или описанный в решении задачи 3 рекурсивный обход графа, построим участки пути от точки старта (локации 1) через необходимые локации обратно до точки старта. Сложив длины каждого участка получим общую длину пути для каждого сталкера. Сравнив пути выбираем кратчайший и выводим на экран.

Задача 3.8.6. (30 баллов)

Теперь сталкеры работают в Зоне одновременно, но им нельзя вместе находится на одной локации, чтобы не раскрыть друг друга. Ваша задача как организатора - собрать все артефакты и провести каждого сталкера так, чтобы он не встретился с остальными. Критерием будет общее время нахождения сталкеров в Зоне. Сталкер может проходить по локациям, где есть "чужие"артефакты. Время старта каждого сталкера можно выбирать произвольно.

Формат входных данных

В 1-й строке - число минут, имеющееся в вашем распоряжении, далее - три строки с номерами локаций с артефактами

Формат выходных данных

3 строки, где первое число в строке: момент старта (рекомендуется 0 для одного из сталкеров), третье и последующие числа в строке: путь обхода (номера локаций, начинается и заканчивается в локации 1). Если по истечении выделенного времени кто-то из сталкеров остался в Зоне, решение не принимается.

Стандартный ввод
99 2 8 10 3 9 11 5 7 13

Стандартный вывод
2 1 2 3 4 7 8 7 10 7 4 3 2 1 0 1 2 3 4 7 8 9 10 11 12 15 1 1 1 6 5 7 10 11 13 15 1

Пояснения к ответу

Перед решением этой задачи необходимо решить предыдущие (№№ 3, 4, 5).

Аналогично решению задачи 5 необходимо построить маршрут через требуемые локации с артефактами. Важно во время обхода графа записывать не только длину пути, но и пройденные вершины (в список или стек). Используя перестановку порядка обхода локаций, находим оптимальный по длине маршрут для каждого сталкера.

Особенность этой задачи в том, что сталкеры не могут входить в Зону одновременно.

Определим для каждого из сталкеров паузу перед стартом (в минутах), например: 0, 1 и 2. Как и в случае с локациями, здесь 3! = 6 перестановок: (0 1 2), (0 2 1), (1 0 2), (1 2 0), (2 0 1), (2 1 0). Кроме того, нужно проверить, не окажутся ли два сталкера на одной локации одновременно. Для этого построим для пути каждого сталкера список из пар значений (время; локация). Проверим, нет ли совпадающих пар (столкновений на локациях). Если случилось столкновение, можно попробовать другие перестановки пауз или удлинить паузу того сталкера, длительность пути которого меньше.

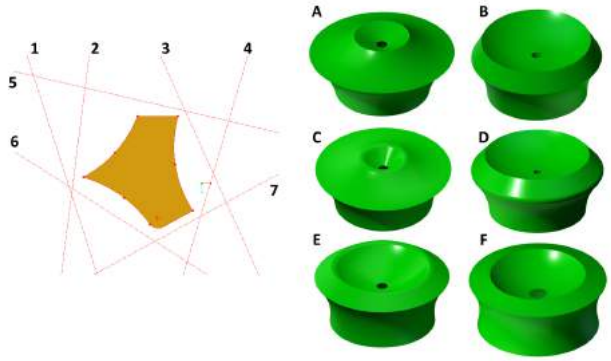
Заметим, что оценивается общее время нахождения сталкеров на локациях, поэтому варьировать паузы на старте лучше у тех сталкеров, путь которых короче.

Основы инженерного 3D-моделирования

4.1. Работа в САПР: общее понимание и базовое моделирование

Задача 4.1.1. (2 балла)

Каждое из показанных справа тел было получено из эскиза слева, вращением плоской фигуры вокруг одной из осей:



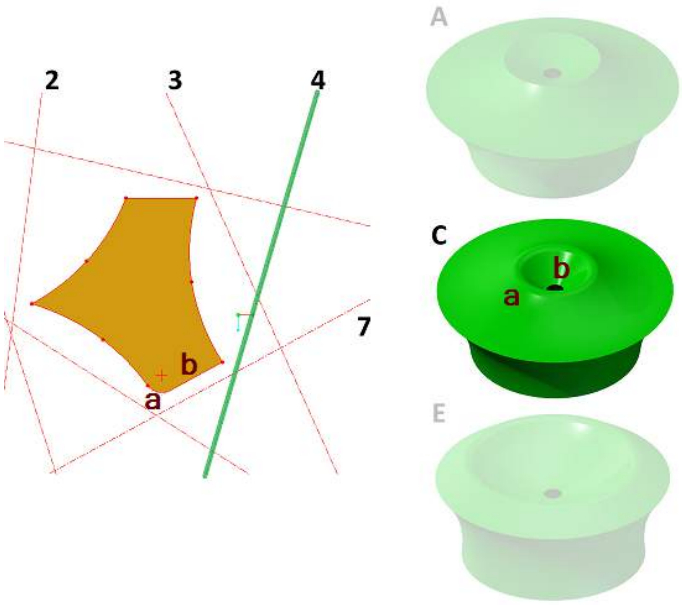
1. Тело A	а. Ось 5
2. Тело B	б. Ось 3
3. Тело C	в. Ось 6
4. Тело D	г. Ось 7
5. Тело E	д. Ось 2
6. Тело F	е. Ось 4
7. Не используется	ж. Ось 1

Решение

По эскизу определяем характерные элементы, соответствие с которыми нужно искать в телах вращения. Для каждого из показанных тел определяем ось, если надо - проверяем несколько осей, исключая неподходящие. Например, объект C имеет следующие характерные признаки:

1. сглаженный край верхнего "кратера", которому должен соответствовать скругленный угол в нижней части контура

2. короткий прямой склон "кратера", угол наклона и малый диаметр отверстия говорят о том, что "кратер" образован отрезком b, а ось должна проходить рядом с правым концом этого отрезка, под углом около 45 градусов к нему.



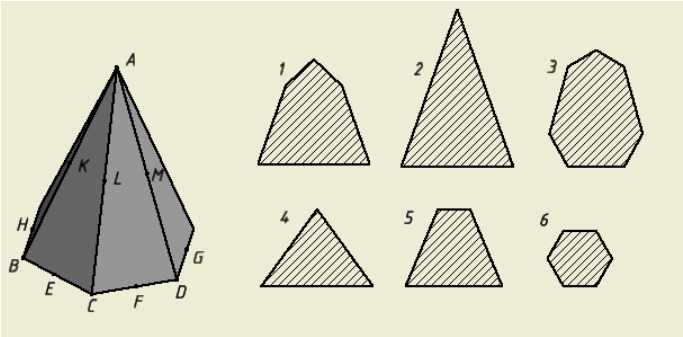
Эти признаки однозначно определяют **ось 4** как нужную нам ось. Аналогично определяются оси для каждого из остальных тел.

Ответ:

1. Тело A	б. Ось 3
2. Тело B	г. Ось 7
3. Тело C	е. Ось 4
4. Тело D	а. Ось 5
5. Тело E	д. Ось 2
6. Тело F	ж. Ось 1
7. Не используется	в. Ось 6

Задача 4.1.2. Геометрия сечений (2 балла)

Имеется пирамида, некоторые вершины и середины ребер которой обозначены буквами. Построено несколько плоскостей сечения, каждая из которых проходит, по крайней мере, через 3 точки и обозначается по этим точкам (например, "плоскость BLD"). Эти плоскости создали следующие сечения. Для каждого сечения укажите соответствующую секущую плоскость (еще 3 сечения этой же пирамиды Вы увидите на следующем шаге):



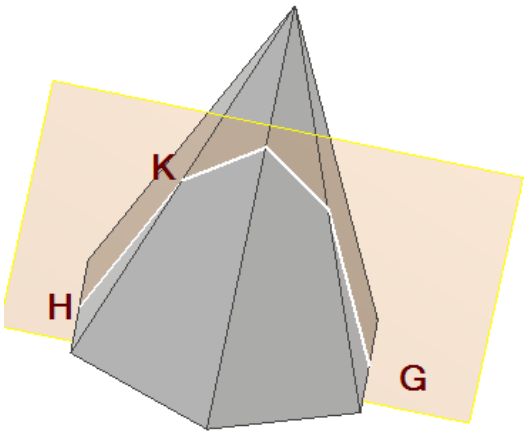
1. Плоскость FGK
2. Плоскость GHK
3. Плоскость ABD

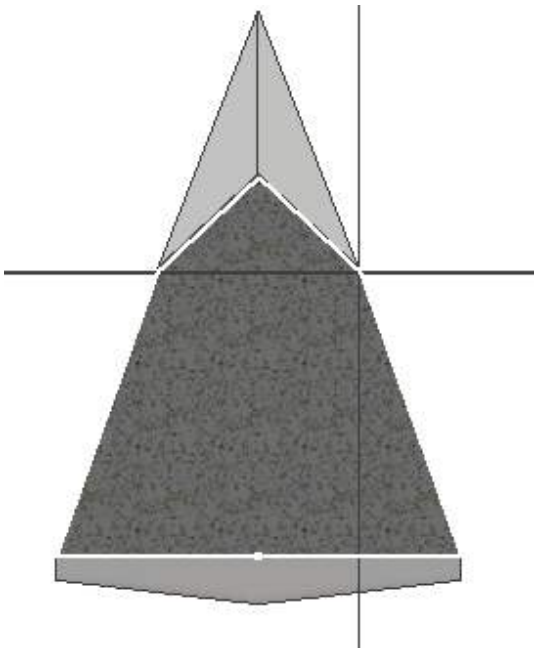
а. Сечение 1
б. Сечение 3
в. Сечение 2

Решение

Обратите внимание, что каждое из сечений данного тела может быть получено при разрезании тела по нескольким разным плоскостям, а сами плоскости могут быть обозначены разными комбинациями букв. Поэтому начинаем не с рассматривания сечений, а с построения плоскостей, предлагаемых как варианты ответов.

Например, первая из предложенных плоскостей, GHK, пройдет, как показано на рисунке. Сразу становится очевидно, что сечением в этом случае становится фигура 1:





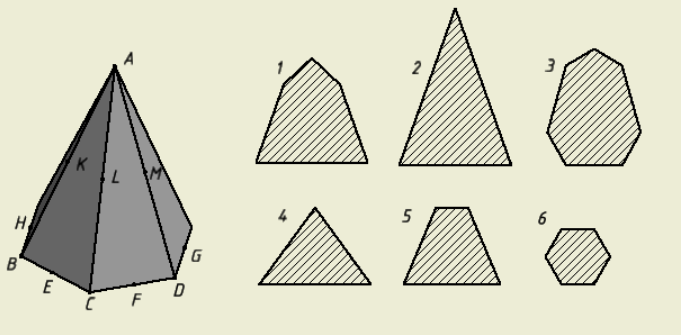
Ответ:

1. Плоскость FGK
2. Плоскость GHK
3. Плоскость ABD

б. Сечение 3
а. Сечение 1
в. Сечение 2

Задача 4.1.3. Геометрия сечений (1 балл)

Продолжаем предыдущую задачу. Сопоставьте еще 3 плоскости и 3 сечения пирамиды:



1. Плоскость EGM
2. Плоскость BDL
3. Плоскость KLM

а. Сечение 6
б. Сечение 4
в. Сечение 5

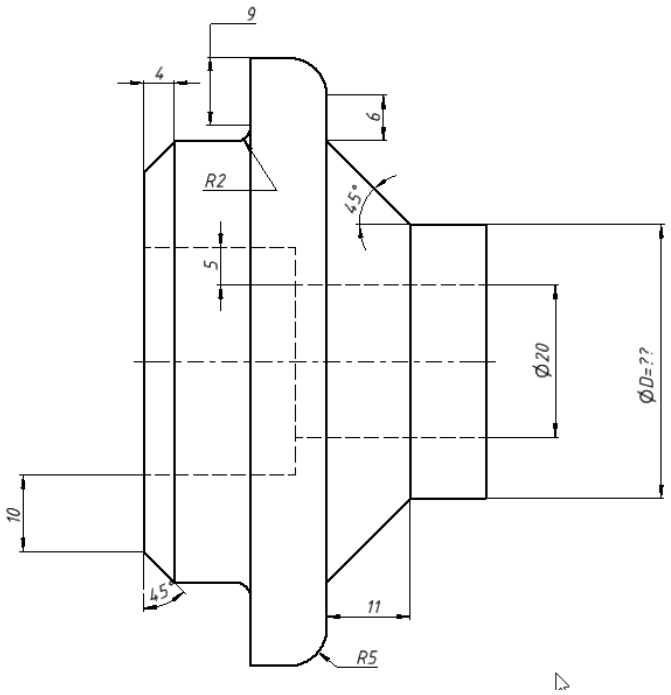
Ответ:

1. Плоскость EGM
2. Плоскость BDL
3. Плоскость KLM

в. Сечение 5
б. Сечение 4
а. Сечение 6

Задача 4.1.4. (1 балл)

По размерам, имеющимся на чертеже (все размеры даны в мм), определите диаметр d:

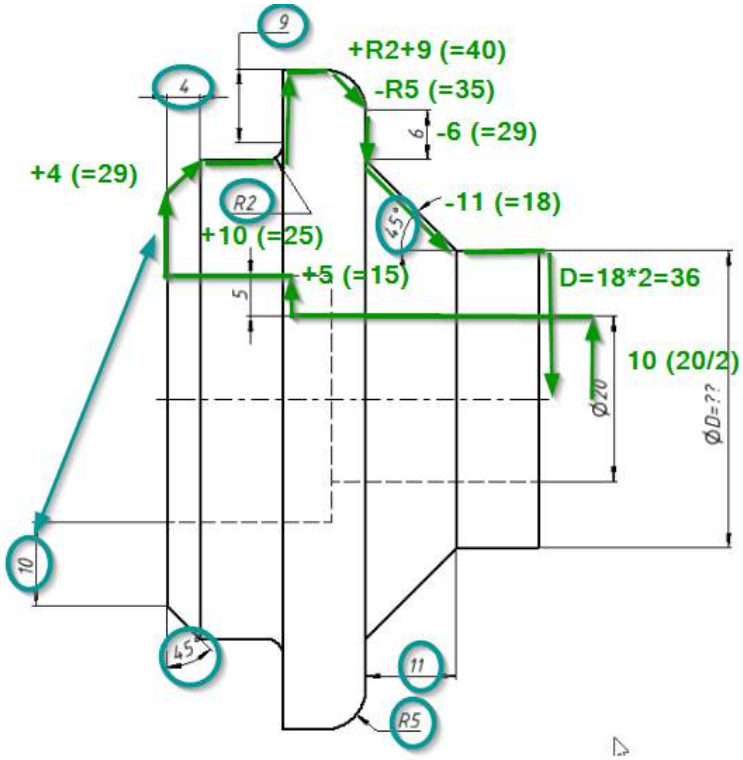


Вычислите d, введите только число (целое).

Решение

В задачах этого типа вас посылают в путешествие вокруг детали, начиная с какого-то известного размера, карабкаясь по цепочке размеров, добавляя или вычитая очередной размер. Препятствия, которые вам могут встретиться в пути:

- Размер поставлен не с той стороны, где вы его ожидаете увидеть,
- Вы "залезаете"или "съезжаете"по склону с наклоном 45 или 30 градусов, и вместо высоты склона, вам дана его ширина. Вычислять хитрые синусы/косинусы не понадобится, но кое-что про свойства треугольников надо знать,
- Вместо вертикальной стенки, вам могут встретиться скругления с заданным радиусом,
- Вы можете забыть в нужный момент перейти от радиуса к диаметру или обратно.

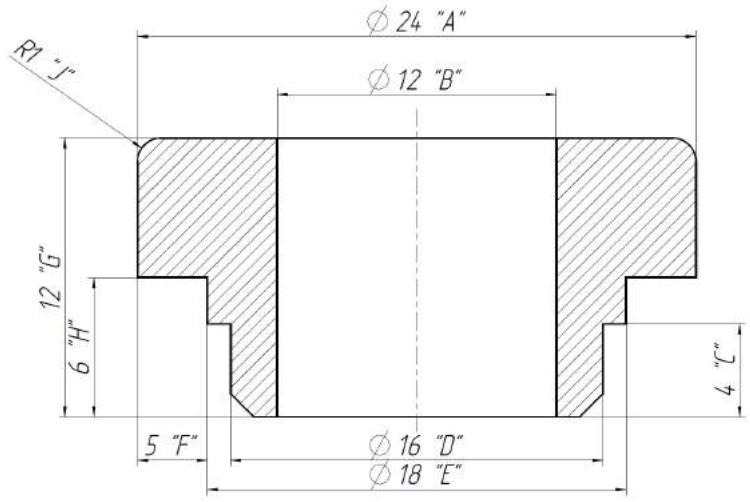


Как видно из рисунка, в данной задаче ответом будет число 36.

Ответ: 36.

Задача 4.1.5. (1 балл)

На этом чертеже неправильно указан один из размеров:



Найдите неверный размер и введите для него правильное значение, указав букву размера и верное значение в формате «буква размера>=<размер>"(например A=24). Все буквы прописные.

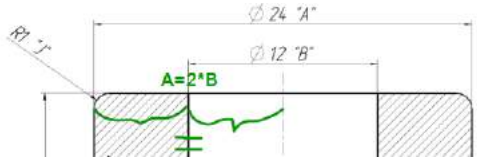
Решение

Назовем избыточным размер, который можно вычислить из других размеров, имеющихся на чертеже. Если такая цепочка размеров присутствует, то избыточным можно считать, на выбор, любой из входящих в нее размеров. При этом совсем необязательно, чтобы избыточный размер оказался неправильным. На самом деле, при генерации чертежей в САПР по 3D-модели наставить лишних размеров очень легко и просто, но нужно специально постараться (вручную изменяя размеры), чтобы один из этих размеров стал неправильным.

В данном примере есть только одна избыточная цепочка - это размеры A, E и F. Эти размеры должны удовлетворять соотношению $2 \cdot F = (A - E)$, однако на чертеже это соотношение не выполняется: $2 \cdot F = 10$, а $A - E = 24 - 18 = 6$. Значит, действительно, один из размеров в цепочке задан неверно. Очень хочется сразу решить, что неправильным (и лишним) является размер F и его настоящее значение должно быть $F = (A - E)/2 = 3$ (мм).

Однако это пока только гипотеза, с тем же успехом ошибка может быть в размере A или в размере E. Попробовать выяснить, который из этих трех размеров ошибочен можно, только сопоставляя их с другими размерами чертежа. Поэтому проверим каждый из размеров A, E, F, учитывая, что по условию задачи неправильный размер есть только один:

- Предположим, что ошибочен размер E. В этом случае его значение должно быть: $E = A - 2 \cdot F = 14$ мм, что не согласуется с чертежом: размер $D = 16$ должен быть меньше E.
- Предположим, что ошибочен размер A. Но на чертеже мы видим, что с показанными на чертеже значениями $A = 2 \cdot B$, и это подтверждается визуально равенством отмеченных отрезков. Если принять что $A = E + 2 \cdot F = 28$, то пропорции чертежа существенно изменились бы.



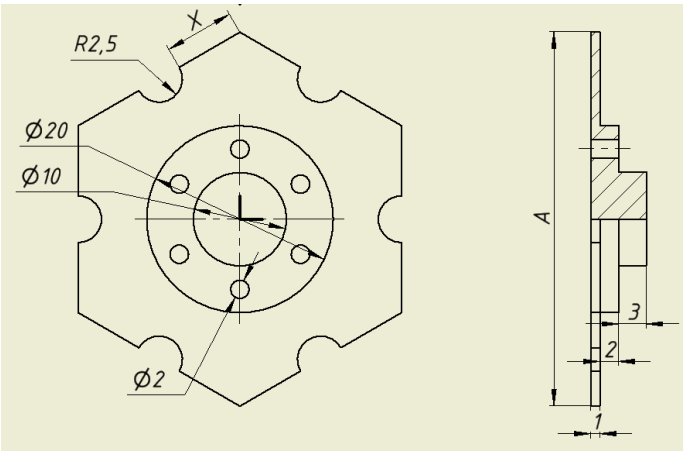
Таким образом, "неправильным"размером действительно является F и ответ, ожидаемый Stepik'ом: $F = 3$.

Ответ: F=3.

4.2. Базовое моделирование

Задача 4.2.1. Пластина (4 балла)

На чертеже изображена некоторая пластина (все размеры проставлены в мм). Определите размер, помеченный на чертеже символом X, при условии, что значение параметра A=40 мм.



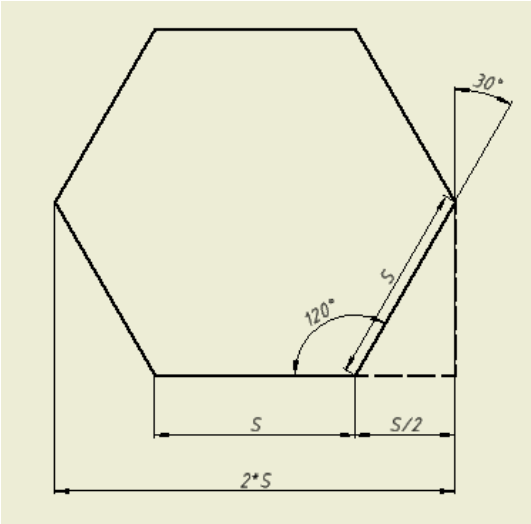
Определите размер X в мм (только число, точный ответ).

Примечание: здесь возможно как довольно простое аналитическое решение, так и решение построением в САПР. Учитывая, что в следующем же шаге та же самая деталь все равно строится полностью (с нахождением объема), быстрее просто начать строить ее и, по ходу, получить требуемый размер. Тем не менее, давайте начнем с аналитического решения.

Решение

Аналитическое

Как показано на рисунке ниже, правильный 6-угольник имеет интересное свойство: расстояние между противолежащими вершинами равно удвоенной длине стороны (т.к. $sin(30°) = 0.5$).

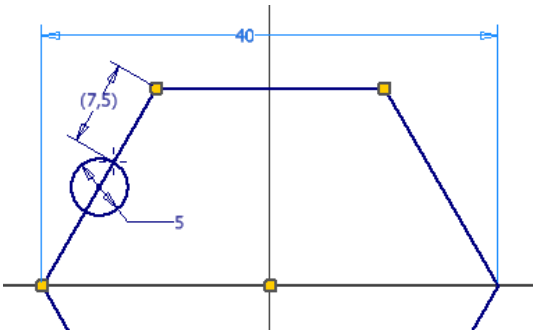


Следовательно, в нашей задаче:

$$X = (A/2 - 2 \cdot R)/2 = (20 - 5)/2 = 7.5 \text{ мм}$$

Построением

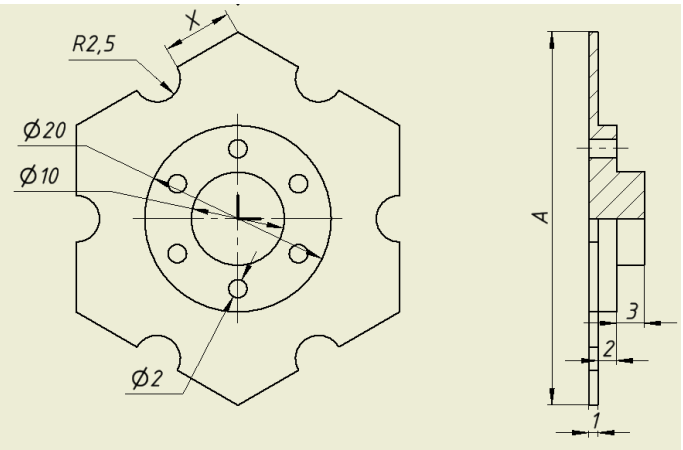
Тривиально: просто строим в эскизе в САПР 6-угольник заданного размера, строим окружность от средней точки ребра, ставим справочный размер. Получается, разумеется, те же 7.5 мм.



Ответ: 7.5

Задача 4.2.2. Пластина-2 (4 балла)

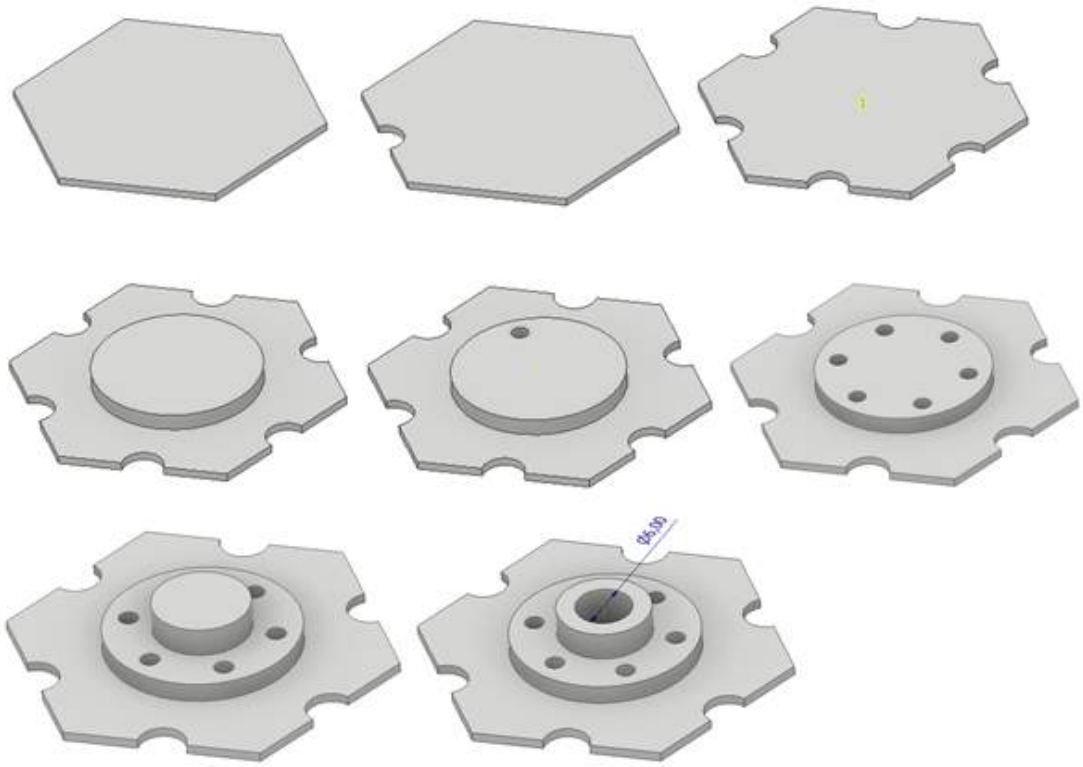
Постройте 3D-модель той же самой пластины (если Вы еще этого не сделали). Дополните модель детали недостающими элементами, чтобы пластину можно было надеть на круглый вал радиусом 3 мм.



Определите массу получившейся детали в граммах (с точностью не ниже 0.1 г) при условии, что она выполнена из высокопрочной низколегированной стали с плотностью 7.85 г/см^3.

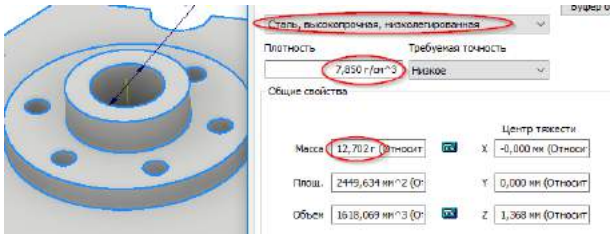
Решение

Просто строим модель по чертежу. Можно сильно упростить себе жизнь, если не пытаться делать сложные эскизы, а строить элементы модели последовательно, вот так:



Здесь применяются только выдавливания и круговые массивы. Очень важно полностью определять каждый эскиз, задавая размеры и привязки для всех его элементов. Обратите внимание на скрытую в задании "засаду": указан *радиус* отверстия, в то время как нам нужен его *диаметр*! Всегда внимательно читайте задание!

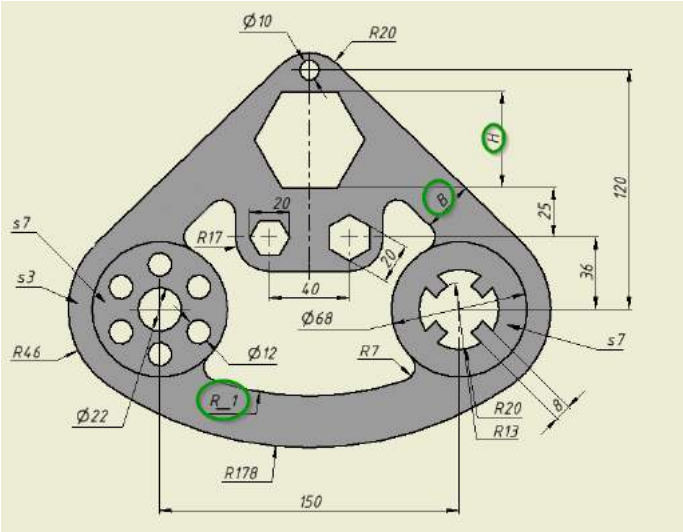
Когда модель готова, средствами вашего САПРа получаем ее массу, для заданного материала. Например, в Autodesk Inventor это будет выглядеть так:



Ответ: 12.7 ± 0.1

Задача 4.2.3. Ключ (6 баллов)

Смоделируйте ключ по приведенному чертежу. Значение параметра H взять равным 48 (мм), параметра R_1 — равным 152 (мм), значение параметра B определяется равенством B=178 – R_1 (мм). Радиусы всех сопряжений по углам внутреннего выреза одинаковы (R7).

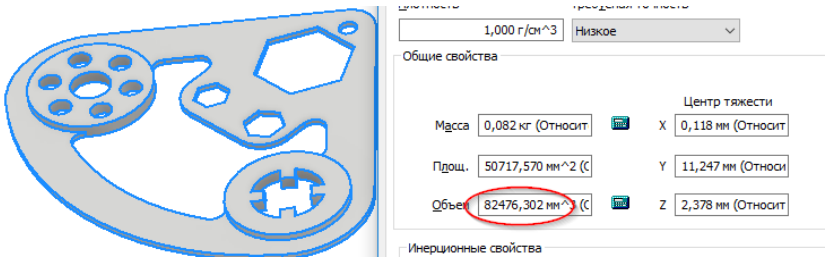


Определите объем детали в мм^3 (число, с точностью до целых).

Пояснения к ответу

Деталь почти плоская, всего лишь с двумя элементами разной толщины. Поскольку вид сбоку не показан, может быть не сразу понятно, откуда брать информацию о толщинах. Она задана выносками "s3"и "s7 т.е. основная пластина детали имеет толщину 3 мм, а два круглых утолщения - 7 мм.

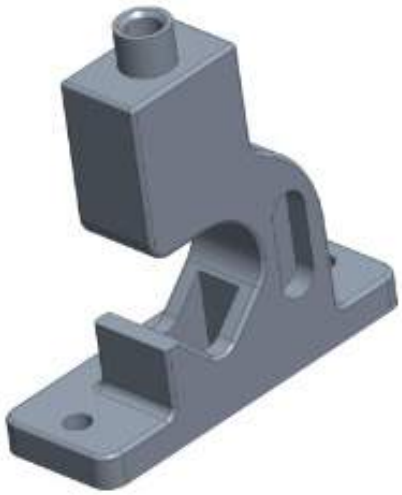
Основная сложность при моделировании этого ключа - правильно построить первый эскиз, применив все показанные на чертеже размеры и догадавшись обо всех использованных привязках и симметриях. Критерий правильности построения - полная определенность эскиза. Например, в Autodesk Inventor с обычной цветовой схемой, все линии эскиз должны стать темно-синими. По готовности модели, получаем объем. Например, в Autodesk Inventor:



Ответ: 82476 ± 20

Задача 4.2.4. Корпус тисков (6 баллов)

Скачайте чертеж корпуса тисков (`https://drive.google.com/file/d/1krX248Yo9j-rr7Zo1zUXn9mG79l2Mcp1/view`) и точно смоделируйте деталь. (На чертеже не указаны размеры: радиус паза = 9 мм, сторона фаски на верхнем отверстии = 1 мм).



Найдите объем детали, в мм^3 (вводить только число, округляя до целого, погрешность не более 5 мм^3)

Пояснения к ответу

Как и в предыдущей задаче, здесь нужно точно смоделировать деталь по чертежу и получить ее объем, однако эта деталь имеет гораздо более сложную форму.

Приводить здесь ход построения большого смысла не имеет, т.к. на следующий год в задании будут фигурировать совсем другие детали. Для тренировки, моделируйте любые детали похожей сложности из учебника по черчению, либо поищите подобные чертежи в интернете. *Следите, чтобы каждый из эскизов был полностью определен!*

Ответ: 391724 ± 20

4.3. Работа в САПР: Сборки

Задача 4.3.1. (2 балла)

Соберите головоломку, используя приложенные файлы (в формате STEP):

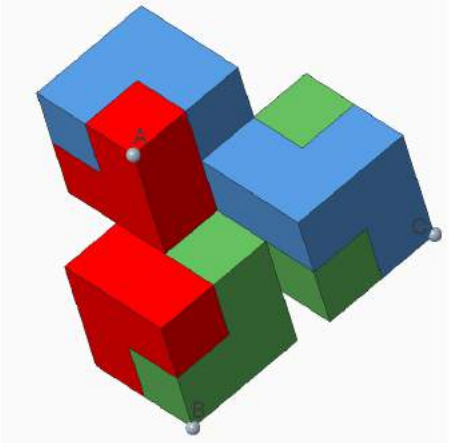
Деталь A

`https://drive.google.com/file/d/1JcXJwKiEi42S5JiICzoJdA31vLrjrEB-/view?usp=sharing`

Деталь B

`https://drive.google.com/file/d/1jTdQmNkdBCKJRBW_yd5K-0kuQvOURAFe/view?usp=sharing` Деталь C

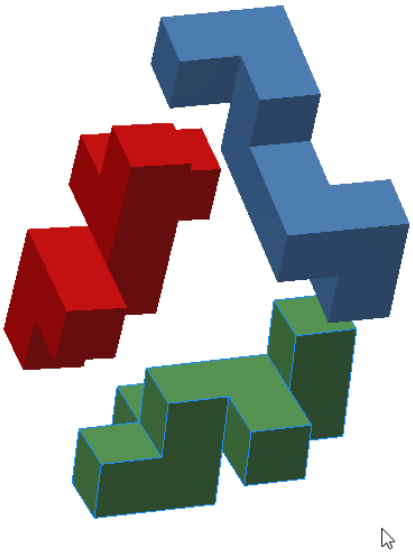
`https://drive.google.com/file/d/1prU86HaGt6WiuD1ZvGuyYfNLL1i98_tx/view?usp=sharing`



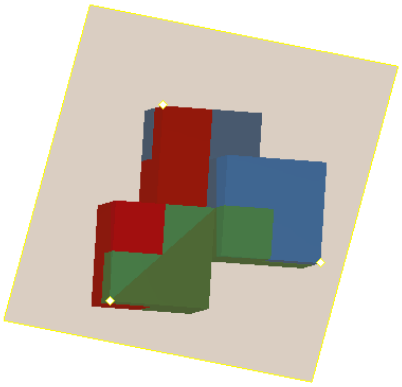
Какова площадь (в мм^2) треугольника, построенного по точкам ABC, указанным на рисунке?

Решение

Создаем сборочную модель, загружаем в нее 3 детали в формате STEP, скачанные по ссылкам. Крутим каждую деталь (в Autodesk Inventor - команда "свободный поворот"), пока она не займет положение, узнаваемое по верхнему рисунку.

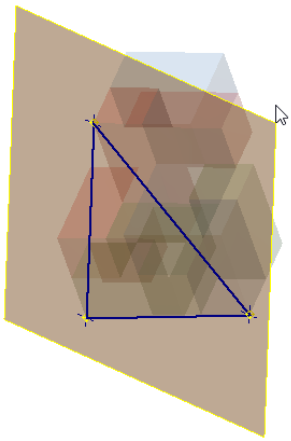


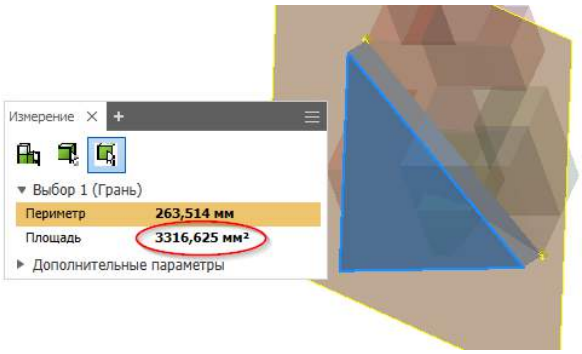
Соединяем детали сборочными зависимостями по граням, ребрам или точкам. Обычно требуется 3 зависимости для соединения каждых двух деталей.



Когда головоломка собрана, строим плоскость по 3-м точкам, показанным в задании. Учитывая, что работа идет в сборке, дальнейшее будет различаться в разных САПР.

Так, в Autodesk Inventor придется создать новую деталь в контексте сборки. После этого В плоскости строим эскиз. а в эскизе - треугольник, построенный по проекциям этих 3х точек. Слегка выдавливаем контур, чтобы измерить его площадь.

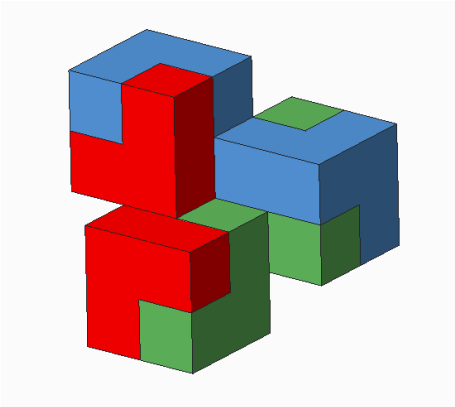




Ответ: 3316.625 ± 1

Задача 4.3.2. (2 балла)

В головоломке, собранной в предыдущей задаче, какая площадь поверхности фигуры C (красная деталь), соприкасается с другими деталями?



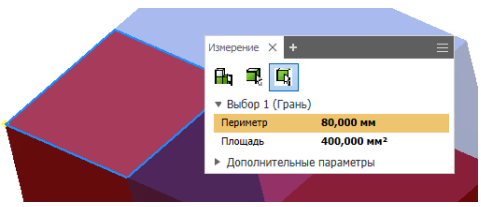
Введите площадь, в мм^2 (только число).

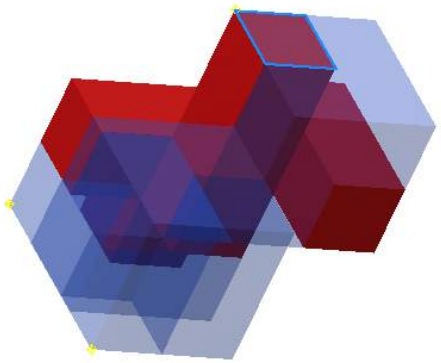
Решение

Для начала, посмотрим площадь единичного квадрата на торце элемента головоломки. Как видно, она составляет 400 мм^2.

Для облегчения подсчетов, делаем остальные 2 детали полупрозрачными. Считаем грани.

Получается 12 единичных квадратов, или 4800 мм^2.





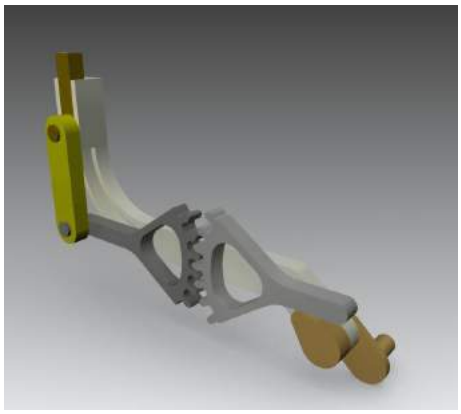
Ответ: 4800.

Задача 4.3.3. Кулачковый механизм (5 баллов)

Скачайте архив с деталями с формате STEP.

(https://drive.google.com/file/d/1NaBgymX6Suge4ll5Ia8cB15TWLGF4S6A/view?usp=sharing)

Соберите из этих деталей кулачковый механизм, как показано на рисунке:



Введите амплитуду движения поршня при вращении рукоятки (т.е. расстояние между крайними положениями поршня), в мм, только число, с точностью не хуже 0.1.

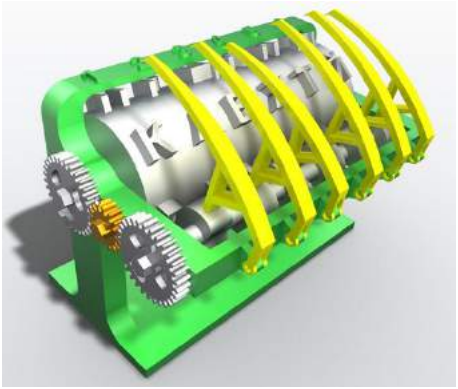
Пояснения к ответу

В задачах этого типа вам предлагается собрать механизм и вычислить диапазон движений, либо найти секретный код, определяемый по движению механизма. Требуется умение применять в сборках динамические зависимости - зубчатые передачи, кулачковые механизмы. Возможны задачи, в которых нужно изменить или добавить детали, чтобы механизм работал. В этом случае спрашивается какой-либо ключевой параметр измененной/добавленной детали. Полный процесс сборки кулачкового механизма (Вариант А) в САПР Autodesk Inventor показан в видеоуроке по адресу: https://bit.ly/2Jih7gs.

Ответ: 10.14 ± 0.1

Задача 4.3.4. Кодовая машина (8 баллов)

Скачайте комплект деталей (https://drive.google.com/file/d/162uWxhMfkUMoLGutxY6GeZiJncI3fekF/view?usp=sharing) и соберите из них показанное ниже устройство. Обратите внимание, что на зубчатых колёсах есть метки для правильной ориентации кулачкового вала относительно барабана с буквами. Если всё собрано правильно, то при вращении барабана рычаги опускаются напротив букв, составляющих слово, которое нужно ввести в ответ задания. Первый кулачок определяет первую букву слова, второй - вторую букву и так далее, всего шесть букв в слове.



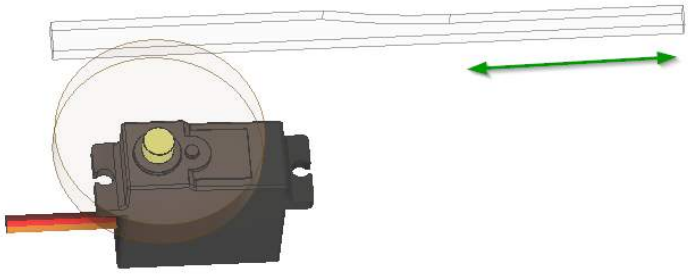
Введите кодовое слово. Ответ вписываем заглавными русскими буквами. Вместо пробела на валу используется символ "_ если в коде попался этот символ, так и вводим "_ ".

Ответ: К_ЦЕЛИ.

4.4. Механика

Задача 4.4.1. (5 баллов)

Вам надо спроектировать узел линейного перемещения для автомата, который раскладывает грузы по 8 ячейкам. Ячейки выстроены в одну линию, расстояние между соседними ячейками 40 мм. В качестве привода применен сервомотор с шестерней и зубчатой рейкой, как показано на рисунке:



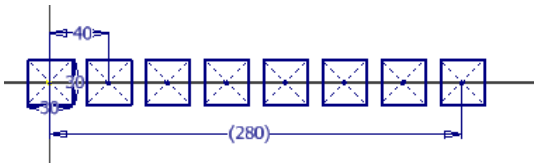
Сервопривод имеет вращательный момент (stall torque) 0.2 Н · м и диапазон поворота вала 270 градусов. Какое максимальное усилие можно получить на зубчатой рейке, при условии, что узел линейного перемещения должен обеспечивать позиционирование схвата над центром любой из ячеек?

Введите максимальное усилие, в Ньютонах (вводить только число), с точностью не ниже 0.1 Н.

Решение

Чем больше диаметр зубчатого колеса, тем больше ход зубчатой рейки, но меньше усилие. Нам нужно максимальное усилие, а значит, минимальный диаметр зубчатого колеса, при котором ход рейки оказывается достаточным для перемещения от

центра первой до центра последней ячейки. Для N ячеек число "перегонов"между ячейками составит $N-1$, а необходимый ход рейки $S = 7 \cdot 40 = 280$ мм:



Такой ход рейки должен составлять 3/4 окружности, поскольку угол поворота вала серво ограничен 270°. Следовательно, радиус зубчатого колеса будет равен:

$$R = \frac{4}{3} \cdot S/(2\pi) = \frac{2S}{3\pi} \approx 59.42 \text{ мм} \approx 0.0594 \text{ м}$$

Вращательный момент будет составлять:

$$T = \frac{Ts}{R} = \frac{0.2}{0.0594} = 3.37 \text{ Н}$$

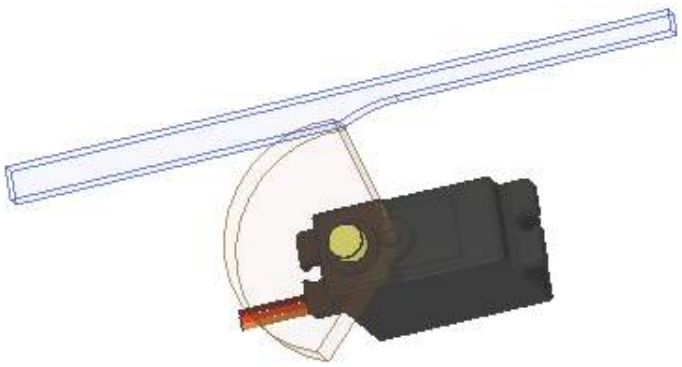
Обратите внимание, что в реальности зубчатое колесо не может иметь произвольный диаметр. При конструировании задается модуль зуба M, а диаметр делительной окружности зубчатого колеса получается умножением модуля зуба на число зубьев. Так, например, если было решено использовать зубчатую передачу с $M = 1.0$ мм, допустимые зубчатые колеса будут иметь целочисленный диаметр (а радиусы с шагом 0.5) и нам придется округлить результат до 60 мм.

В этом случае $T = 3.33$ Н. И тот и другой ответ находится в диапазоне допустимых погрешностей для данной задачи в Stepik.

Ответ: 3.33 ± 0.1

Задача 4.4.2. (4 балла)

Продолжаем конструировать тот же узел линейного перемещения. После покупки сервопривода неожиданно оказалось, что купленная модель обеспечивает поворот вала только на 120 градусов. Поэтому, и для сокращения размеров, было решено вместо полного зубчатого колеса использовать зубчатый сектор:



При использовании модуля зуба 1.5 мм, сколько зубьев будет иметь этот зубчатый сектор?

Ответ: 60 ± 1

Задача 4.4.3. (5 баллов)

Ось X 3D-принтера приводится в действие шаговым двигателем NEMA17 с вот такими параметрами:

Крутящий Момент: 4000g.cm	Угол Шага (градусы): 1.8 degree
Артикул: 17HS4401	Ток / Фаза: 1.7A
Фаза: 2	Тип: Гибридные

На валу двигателя смонтирован шкив с 20 зубьями под зубчатый ремень типа GT2 (с шагом зуба 2 мм), перемещающий каретку. В драйвере мотора используется микрошаг 1/16.

Сколько шагов мотора необходимо выполнить, чтобы переместить каретку на A мм?

Решение

В данной задаче не требуется работать с силами и моментами вращения, но нужны базовые знания о принципах работы шагового двигателя и ременной передачи.

Из таблички с данными о моторе нам понадобится только одна величина: угол шага 1.8°. Это означает, что 1 оборот вал мотора совершит за $360/1.8 = 200$ шагов, или, наоборот, за 1 шаг вал поворачивается на 1/200 оборота. Кроме того, драйвер шагового двигателя умеет работать с микрошагами, искусственными промежуточными положениями вала внутри одного шага. Например, микрошаг 1/16 означает, что для поворота вала двигателя на один шаг нужно подать 16 управляющих импульсов.

Нам не потребуется диаметр шкива и число π, чтобы вычислять длину окружности. Достаточно знать, что шкив рассчитан на ремень с шагом зуба 2 мм и имеет 20 зубцов. Диаметр шкива подобран так, чтобы именно столько зубцов укладывалось по окружности. Это означает, что за один оборот шкива ремень продвинется на $S = 20 \cdot 2 = 40$ мм.

Таким образом, чтобы переместить каретку на X мм, на драйвер мотора необходимо подать N шаговых импульсов:

$$N = \frac{X}{40} \cdot 200 \cdot 16 = 80 \cdot X \text{ (микрошагов)}$$

В задаче в Stepik вместо переменной X каждый раз появляется случайное число, и вам нужно ввести числовой ответ.

Ответ: $80 \cdot a$

Задачи второго этапа. Виртуальная реальность

5.1. Задачи

Задача 5.1.1. Gorn (100 баллов)

Вильгельм играет в игру Gorn в виртуальной реальности, сражаясь с N гладиаторами на арене. Он имеет при себе лук и бесконечное число стрел и никогда не промахивается по врагам.

Когда Вильгельм в первый раз попадает по вражескому гладиатору, он наносит ему D единиц урона, гладиатор при этом теряет D единиц здоровья.

При каждом следующем попадании по этому гладиатору урон увеличивается на K единиц из-за ослабевающей брони гладиатора. При попадании по новому гладиатору урон также начинается с D и увеличивается с каждым попаданием. Гладиатор погибает, когда количество его здоровья достигает нуля или становится отрицательным.

Сколько стрел должен потратить Вильгельм, чтобы убить всех гладиаторов?

Формат входных данных

Первая строка входного файла содержит целые числа N, D и K. В следующей строке содержится N целых чисел h_i – количество здоровья у вражеских гладиаторов.

$1 \leqslant N \leqslant 10^4$

$1 \leqslant D \leqslant 10^5$

$1 \leqslant K \leqslant 10^3$

Формат выходных данных

Программа должна вывести одно целое число – количество стрел, которые необходимо потратить.

Система оценки

Баллы за 1 подзадачу начисляются только в случае, если все тесты для этой подзадачи успешно пройдены. Баллы за подзадачу 2 начисляются за каждый пройденный тест, если тесты необходимых подзадач пройдены.

Подзадача	Баллы	Дополнительные ограничения, h_i	Необходимые подзадачи
1	40	$1 \leqslant h_i \leqslant 10^2$	
2	60	$1 \leqslant h_i \leqslant 10^5$	1

Пример №1

Стандартный ввод
4 1 1 1 2 3 4
Стандартный вывод
8

Решение

Обозначим за x количество стрел, которые попали в очередного гладиатора со здоровьем h. Общий урон этих стрел равен

$$D + (D + K) + (D + 2 \cdot K) + \cdots + D + x \cdot K = 1/2 \cdot (D + D + x \cdot K) \cdot x = 1/2 \cdot K \cdot x^2 + D \cdot x.$$

Тогда по условию задачи нужно найти наименьшее целое x, при котором

$$Kx^2 + (2D - K)x \geqslant 2 \cdot h.$$

Решая неравенство, получим:

$$x = \sqrt{(2 \cdot D - K)^2 + 8 \cdot K \cdot h}$$
$$x_1 = \lfloor \frac{K - 2 \cdot D - x}{2 \cdot K} \rfloor$$
$$x_2 = \lceil \frac{K - 2 \cdot D + x}{2 \cdot K} \rceil$$

Выбираем положительный корень. Суммируем величины, полученные для каждого гладиатора.

Пример программы-решения

Ниже представлено решение на языке Python3

```
from math import ceil, sqrt

def arrows(h):
    global D, K
    sd = sqrt((2*D - K)**2 + 8*K*h)
    x1, x2 = int((K - 2*D - sd)/2/K), int(ceil((K - 2*D + sd)/2/K))
    return x1 if x1 > 0 else x2

N, D, K, *H = map(int, open('input.txt', 'r').read().split())
open('output.txt', 'w').write(str(sum(map(arrows, H))))
```

Задача 5.1.2. Мортира (100 баллов)

Вы участвуете в разработке игры в виртуальной реальности жанра Tower Defense. В играх подобного жанра, целью является защита основного замка от наступления вражеских юнитов.

Одно из защитных орудий в игре – мортира. Это пушка которая может регулировать угол наклона в пределах от 45 до 90 градусов от земли. Сама себе пушка урон нанести не может.

Как разработчику, вам поручили задание разработать искусственный интеллект для этого орудия. На начальных уровнях искусственный интеллект должен быть достаточно прост, вплоть до того, что он ничего не должен знать о параметрах орудия. Орудие имеет ограниченное количество боеприпасов - 20 зарядов. До того как боезапас иссякнет, пушка должна поразить хотя бы одну цель. Все, что может узнать ваш ИИ - это расстояние, на котором находится цель, и расстояние, на которое улетел снаряд после выстрела. Искусственный интеллект может управлять лишь углом наклона орудия.

Ваши коллеги уже создали для вас простую тестовую сцену, и даже написали за вас API. Вам лишь необходимо реализовать интерфейс. Ссылка на репозиторий с проектом: `https://github.com/Lukaviy/AI-for-defense.git`.

Также, ваши коллеги еще не до конца уверены в выборе игрового движка. Возможно в скором будущем они перейдут на другой. Поэтому они убедительно просят вас воздержаться от использования функций, зависимых от Unity. (Вместо Mathf использовать System.Math, и т.д.)

Необходимо реализовать класс CannonAI со следующим интерфейсом:

```
public class CannonAI : ICannonAI
{
    // Расстояние на котором находится цель
    void SetTarget(double distance);
    // Угол наклона в градусах в который нужно установить пушку перед выстрелом
    double GetShootAngle();
    // Информация о дальности полета снаряда
    void FeedbackHitDistance(double distance);
}
```

Формат входных данных

Гарантируется, что пушка всегда может достать до противника.

$45 \leqslant Angle \leqslant 90$

$10 \leqslant Distance \leqslant 10^6$

Формат выходных данных

Файл с решением должен содержать только реализацию класса CannonAI. В качестве среды программирования необходимо выбирать C#.

Решение

При изменении угла в диапазоне от 45 до 90 градусов расстояние от точки запуска до точки падения изменяется монотонно. Таким образом, данная задача решается стандартным алгоритмом бинарного поиска.

Основной особенностью и целью задачи является знакомство участников с техникой реализации и отправки на проверку модулей Unity-проектов.

Пример программы-решения

Ниже представлено решение на языке C#

```
public class CannonAI : ICannonAI
{
    private double _left;
    private double _right;
    private double _target;

    public void SetTarget(double distance)
    {
        _target = distance;
        _left = 45;
        _right = 90;
    }

    public double GetShootAngle()
    {
        return (_right + _left) / 2;
    }

    public void FeedbackHitDistance(double distance)
    {
        if (distance > _target)
        {
            _left = GetShootAngle();
        }
        else
        {
            _right = GetShootAngle();
        }
    }
}
```

Задача 5.1.3. RollerBall-1 (100 баллов)

Юный программист Вася решил поиграть в RollerBall. Цель игры — собрать как можно больше монет, которые находятся в лабиринте. Для того, чтобы взять монетку, её границы нужно пересечь мячом, которым Вася может управлять. У мяча есть энергия, которая расходуется на каждом шаге.

Вася хочет собрать очень много монет и для этого решил написать программу, которая бы управляла мячом и сама собирала все монеты в лабиринте. К сожалению, Вася очень плохо знает C# и просит Вас ему помочь. Проект `https:`

//github.com/BabichMikhail/NTI_TaskC_Maze, который написал Вася, уже содержит игру и умеет вводить и выводить файлы нужных форматов. Вам осталось реализовать класс AutoBallControl для управления мячом.

MazeCell.cs

```
public enum Direction
{
   Start,
   Right,
   Front,
   Left,
   Back,
};

public class MazeCell
{
   public bool IsVisited = false;
   public bool WallRight = false;
   public bool WallFront = false;
   public bool WallLeft = false;
   public bool WallBack = false;
   public bool IsGoal = false;
}
```

MazeSpawn.cs

```
using System.Collections.Generic;
using System.IO;
using System.Linq;
using UnityEngine;

public class MazeSpawner : MonoBehaviour
{
  public GameObject Floor;
  public GameObject Wall;
  public GameObject Pillar;
  public GameObject GoalPrefab;

  public bool ReadMazeDataFromFile;

  public int SeedValue;
  public int Rows;
  public int Cols;
  public int Coins;
  public int BallEnergy;

  private void ReadInputFile()
  {
  if (ReadMazeDataFromFile) {
  var args = File.ReadAllText("input.txt").Split(' ');
  MazeDescription.SeedValue = int.Parse(args[0]);
  MazeDescription.Rows = int.Parse(args[1]);
  MazeDescription.Cols = int.Parse(args[2]);
  MazeDescription.Coins = int.Parse(args[3]);
  MazeDescription.BallEnergy = int.Parse(args[4]);
  }
  else {
  MazeDescription.SeedValue = SeedValue;
```

```
33    MazeDescription.Rows = Rows;
34    MazeDescription.Cols = Cols;
35    MazeDescription.Coins = Coins;
36    MazeDescription.BallEnergy = BallEnergy;
37    }
38    }
39
40    private void Awake()
41    {
42    ReadInputFile();
43    MazeDescription.PillarPrefab = Pillar;
44    MazeDescription.WallPrefab = Wall;
45    MazeDescription.CoinPrefab = GoalPrefab;
46
47    if (MazeDescription.IsConsoleRun()) {
48    var mainCamera = GameObject.FindGameObjectWithTag("MainCamera");
49    if (mainCamera != null)
50    mainCamera.SetActive(true);
51    }
52
53    Random.InitState(MazeDescription.SeedValue);
54
55    var mazeDescriptionCells = new List<MazeDescriptionCell>();
56    var cellsWithCoinIndexes = new List<int>();
57    var mMazeGenerator = new TreeMazeGenerator(MazeDescription.Rows,
58    MazeDescription.Cols);
59    mMazeGenerator.GenerateMaze();
60    for (var row = 0; row < MazeDescription.Rows; ++row) {
61    for (var column = 0; column < MazeDescription.Cols; ++column) {
62    var x = column * MazeDescription.CellWidth;
63    var z = row * MazeDescription.CellHeight;
64    var cell = mMazeGenerator.GetMazeCell(row, column);
65
66    var floor = Instantiate(Floor, new Vector3(x, 0, z), Quaternion.Euler(0, 0, 0));
67    floor.transform.parent = transform;
68
69    if (cell.WallRight) {
70    var wall = Instantiate(Wall, new Vector3(x + MazeDescription.CellWidth / 2, 0, z) +
71    Wall.transform.position, Quaternion.Euler(0, 90, 0));
72    wall.transform.parent = transform;
73    }
74    if (cell.WallFront) {
75    var wall = Instantiate(Wall, new Vector3(x, 0, z + MazeDescription.CellHeight / 2) +
76    Wall.transform.position, Quaternion.Euler(0, 0, 0));
77    wall.transform.parent = transform;
78    }
79    if (cell.WallLeft) {
80    var wall = Instantiate(Wall, new Vector3(x - MazeDescription.CellWidth / 2, 0, z) +
81    Wall.transform.position, Quaternion.Euler(0, 270, 0));
82    wall.transform.parent = transform;
83    }
84    if (cell.WallBack) {
85    var wall = Instantiate(Wall, new Vector3(x, 0, z - MazeDescription.CellHeight / 2) +
86    Wall.transform.position, Quaternion.Euler(0, 180, 0));
87    wall.transform.parent = transform;
88    }
89
90    var mazeDescriptionCell = new MazeDescriptionCell {
91    Row = row,
92    Column = column,
```

```
    CanMoveBackward = !cell.WallBack,
    CanMoveLeft = !cell.WallLeft,
    CanMoveRight = !cell.WallRight,
    CanMoveForward = !cell.WallFront,
    HasCoin = false,
    };
    if (cell.IsGoal)
    cellsWithCoinIndexes.Add(mazeDescriptionCells.Count);
    mazeDescriptionCells.Add(mazeDescriptionCell);
    }
    }

    Debug.Assert(cellsWithCoinIndexes.Count >= MazeDescription.Coins);
    cellsWithCoinIndexes = cellsWithCoinIndexes.OrderBy(x => Random.value).ToList();
    for (var i = 0; i < MazeDescription.Coins; ++i) {
    var cell = mazeDescriptionCells[cellsWithCoinIndexes[i]];
    cell.HasCoin = true;
    var x = cell.Column * MazeDescription.CellWidth;
    var z = cell.Row * MazeDescription.CellHeight;
    var coin = Instantiate(GoalPrefab, new Vector3(x, 1, z), Quaternion.Euler(0, 0, 0));
    coin.transform.parent = transform;
    }

    if (Pillar != null) {
    for (var row = 0; row <= MazeDescription.Rows; ++row) {
    for (var column = 0; column <= MazeDescription.Cols; ++column) {
    var x = column * MazeDescription.CellWidth;
    var z = row * MazeDescription.CellHeight;
    var pillar = Instantiate(Pillar, new Vector3(x - MazeDescription.CellWidth / 2, 0,
    z - MazeDescription.CellHeight / 2), Quaternion.identity);
    pillar.transform.parent = transform;
    }
    }
    }

    MazeDescription.Cells = mazeDescriptionCells;
    }
}
```

RollerBall.cs

```
using System;
using System.Collections.Generic;
using System.IO;
using UnityEngine;

public class RollerBall : MonoBehaviour
{
    public GameObject ViewCamera;
    public bool UseManualBallController;

    private const int IntervalMilliseconds = 25;
    private const float Speed = 4f;

    private Rigidbody mRigidBody;
    private int lastCallTime = -IntervalMilliseconds;
    private int iterations;
    private BallControl ballController;
    private int visitedCoinCount;
    private float successTime;
```

```
20
21      private readonly List<Vector2> ballPositions = new List<Vector2>();
22      private bool finished;
23      private bool success;
24      private bool outputWrote;
25
26      private void Start()
27      {
28          mRigidBody = GetComponent<Rigidbody>();
29          Debug.Assert(mRigidBody != null);
30          lastCallTime = (int)(Time.time * 1000) + IntervalMilliseconds;
31          ballController = UseManualBallController ? (BallControl) new ManualBallControl() :
32              new AutoBallControl();
33          try {
34              ballController.SetMaze();
35          }
36          catch (Exception) {
37              Application.Quit(1);
38          }
39          if (MazeDescription.IsConsoleRun())
40              Time.timeScale = 100.0f;
41      }
42
43      private void Finish()
44      {
45          finished = true;
46          SaveBallPosition();
47          Time.timeScale = 1.0f;
48          if (MazeDescription.IsConsoleRun()) {
49              WriteOutputFile();
50              Application.Quit();
51          }
52      }
53
54      private void WriteOutputFile()
55      {
56          if (outputWrote)
57              return;
58          outputWrote = true;
59          var lines = new List<string>{
60              success ? "Success" : (MazeDescription.Coins - visitedCoinCount).ToString(),
61              ballPositions.Count.ToString(),
62          };
63          foreach (var ballPosition in ballPositions)
64              lines.Add(ballPosition.x + " " + ballPosition.y);
65          File.WriteAllLines("output.txt", lines.ToArray());
66      }
67
68      private void SaveBallPosition()
69      {
70          ballPositions.Add(new Vector2(transform.position.x, transform.position.z));
71      }
72
73      private void FixedUpdate()
74      {
75          if (visitedCoinCount == MazeDescription.Coins) {
76              if (!finished)
77                  successTime = Time.unscaledTime;
78              Debug.Log("Success. Time:  " + successTime);
79              success = true;
```

```
80          Finish();
81          return;
82       }
83
84       if (iterations == MazeDescription.BallEnergy) {
85          Debug.Log("Ball has no energy!");
86          success = false;
87          Finish();
88          return;
89       }
90
91                if (true || !MazeDescription.IsConsoleRun()) {
92          var stepCount = 0;
93          while (Time.time * 1000 + IntervalMilliseconds >= lastCallTime) {
94             ++stepCount;
95             lastCallTime += IntervalMilliseconds;
96          }
97          if (stepCount == 0)
98             return;
99       }
100
101      SaveBallPosition();
102      if (mRigidBody != null) {
103         int move = 0;
104         try {
105            move = ballController.GetMove(transform.position.x, transform.position.z);
106         }
107         catch (Exception) {
108            Application.Quit(1);
109         }
110         var velocity = Vector3.zero;
111         if ((move & BallControl.MoveTypeRight) != 0)
112            velocity += Vector3.right;
113         if ((move & BallControl.MoveTypeBottom) != 0)
114            velocity += Vector3.back;
115         if ((move & BallControl.MoveTypeLeft) != 0)
116            velocity -= Vector3.right;
117         if ((move & BallControl.MoveTypeTop) != 0)
118            velocity -= Vector3.back;
119
120         mRigidBody.velocity = velocity;
121         if (velocity != Vector3.zero)
122            mRigidBody.velocity = velocity.normalized * Speed;
123         Debug.Log(iterations);
124         ++iterations;
125      }
126
127      if (ViewCamera != null) {
128         var direction = (Vector3.up * 5 + Vector3.back) * 4;
129         RaycastHit hit;
130         Debug.DrawLine(transform.position,transform.position + direction,Color.red);
131         ViewCamera.transform.position =
132            Physics.Linecast(transform.position, transform.position + direction, out hit)
133               ? hit.point
134               : transform.position + direction;
135         ViewCamera.transform.LookAt(transform.position);
136      }
137   }
138
139   private void OnTriggerEnter(Collider other)
```

```
140     {
141         if (other.gameObject.tag.Equals("Coin")) {
142             Destroy(other.gameObject);
143             ++visitedCoinCount;
144         }
145     }
146 }
```

TreeMazeGenerator.cs

```
1  using System;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using Random = UnityEngine.Random;
5
6  public class TreeMazeGenerator
7  {
8    private struct CellToVisit
9    {
10     public readonly int Row;
11     public readonly int Column;
12     public readonly Direction MoveMade;
13
14     public CellToVisit(int row, int column, Direction move)
15     {
16       Row = row;
17       Column = column;
18       MoveMade = move;
19     }
20   }
21
22   private int RowCount { get; set; }
23   private int ColumnCount { get; set; }
24
25   private readonly MazeCell[, ] mMaze;
26   private readonly List<CellToVisit> mCellsToVisit = new List<CellToVisit> ();
27
28   public TreeMazeGenerator(int rows, int columns)
29   {
30     RowCount = Mathf.Abs(rows);
31     if (RowCount == 0)
32       RowCount = 1;
33     ColumnCount = Mathf.Abs(columns);
34     if (ColumnCount == 0)
35       ColumnCount = 1;
36
37     mMaze = new MazeCell[rows, columns];
38     for (var row = 0; row < rows; ++row)
39     for (var column = 0; column < columns; ++column)
40       mMaze[row, column] = new MazeCell();
41   }
42
43   public MazeCell GetMazeCell(int row, int column)
44   {
45     if (row >= 0 && column >= 0 && row < RowCount && column < ColumnCount)
46       return mMaze[row, column];
47     throw new ArgumentOutOfRangeException();
48   }
49
50   public void GenerateMaze ()
```

```
51    {
52      var movesAvailable = new Direction[4];
53      mCellsToVisit.Add(new CellToVisit(Random.Range(0, RowCount),
54      Random.Range(0, ColumnCount), Direction.Start));
55      while (mCellsToVisit.Count > 0) {
56        var movesAvailableCount = 0;
57        var ctv = mCellsToVisit[mCellsToVisit.Count - 1];
58
59        //check move right
60        if (ctv.Column + 1 < ColumnCount
61          && !GetMazeCell(ctv.Row, ctv.Column + 1).IsVisited
62          && !IsCellInList(ctv.Row, ctv.Column + 1)) {
63          movesAvailable[movesAvailableCount] = Direction.Right;
64          ++movesAvailableCount;
65        }
66        else if (!GetMazeCell(ctv.Row, ctv.Column).IsVisited &&
67          ctv.MoveMade != Direction.Left) {
68          GetMazeCell(ctv.Row, ctv.Column).WallRight = true;
69          if (ctv.Column + 1 < ColumnCount)
70            GetMazeCell(ctv.Row,ctv.Column+1).WallLeft = true;
71        }
72
73        //check move forward
74        if (ctv.Row + 1 < RowCount
75          && !GetMazeCell(ctv.Row + 1, ctv.Column).IsVisited
76          && !IsCellInList(ctv.Row + 1, ctv.Column)){
77          movesAvailable[movesAvailableCount] = Direction.Front;
78          ++movesAvailableCount;
79        }
80        else if (!GetMazeCell(ctv.Row, ctv.Column).IsVisited
81          && ctv.MoveMade != Direction.Back) {
82          GetMazeCell(ctv.Row, ctv.Column).WallFront = true;
83          if (ctv.Row + 1 < RowCount)
84            GetMazeCell(ctv.Row+1,ctv.Column).WallBack = true;
85        }
86
87        //check move left
88        if(ctv.Column > 0 && ctv.Column - 1 >= 0
89          && !GetMazeCell(ctv.Row, ctv.Column - 1).IsVisited
90          && !IsCellInList(ctv.Row, ctv.Column - 1)){
91          movesAvailable[movesAvailableCount] = Direction.Left;
92          ++movesAvailableCount;
93        }
94        else if (!GetMazeCell(ctv.Row, ctv.Column).IsVisited
95          && ctv.MoveMade != Direction.Right){
96          GetMazeCell(ctv.Row, ctv.Column).WallLeft = true;
97          if (ctv.Column > 0 && ctv.Column - 1 >= 0)
98            GetMazeCell(ctv.Row, ctv.Column - 1).WallRight = true;
99        }
100
101       //check move backward
102       if (ctv.Row > 0 && ctv.Row - 1 >= 0 &&
103         !GetMazeCell(ctv.Row - 1, ctv.Column).IsVisited &&
104         !IsCellInList(ctv.Row - 1, ctv.Column)) {
105         movesAvailable[movesAvailableCount] = Direction.Back;
106         ++movesAvailableCount;
107       }
108       else if (!GetMazeCell(ctv.Row, ctv.Column).IsVisited
109         && ctv.MoveMade != Direction.Front) {
110         GetMazeCell(ctv.Row, ctv.Column).WallBack = true;
```

```
111           if (ctv.Row > 0 && ctv.Row - 1 >= 0)
112             GetMazeCell(ctv.Row - 1, ctv.Column).WallFront = true;
113         }
114
115         if (!GetMazeCell(ctv.Row, ctv.Column).IsVisited && movesAvailableCount == 0)
116           GetMazeCell(ctv.Row, ctv.Column).IsGoal = true;
117         GetMazeCell(ctv.Row, ctv.Column).IsVisited = true;
118
119         if (movesAvailableCount > 0) {
120           switch (movesAvailable[Random.Range(0, movesAvailableCount)]) {
121           case Direction.Start:
122             break;
123           case Direction.Right:
124             mCellsToVisit.Add(new CellToVisit(ctv.Row, ctv.Column + 1, Direction.Right));
125             break;
126           case Direction.Front:
127             mCellsToVisit.Add(new CellToVisit(ctv.Row + 1, ctv.Column, Direction.Front));
128             break;
129           case Direction.Left:
130             mCellsToVisit.Add(new CellToVisit(ctv.Row, ctv.Column - 1, Direction.Left));
131             break;
132           case Direction.Back:
133             mCellsToVisit.Add(new CellToVisit(ctv.Row - 1, ctv.Column, Direction.Back));
134             break;
135           default:
136             throw new ArgumentOutOfRangeException();
137           }
138         }
139         else
140           mCellsToVisit.Remove(ctv);
141       }
142     }
143
144     private bool IsCellInList(int row, int column)
145     {
146       return mCellsToVisit.FindIndex(other => other.Row == row
147       && other.Column == column) >= 0;
148     }
149   }
```

Первый тест совпадает с примером, содержащимся в файле input.txt в репозитории проекта.

Формат входных данных

Входной файл содержит 5 целых чисел S, R, C, N, E —- номер (seed) лабиринта, количество строк и столбцов лабиринта, количество монет, количество энергии у мяча.

$0 \leqslant S \leqslant 10^7$

$1 \leqslant R \leqslant 600$

$1 \leqslant C \leqslant 600$

$2 \leqslant R \cdot C \leqslant 600$

$0 \leqslant N \leqslant 9$

$0 \leqslant E \leqslant 3 \cdot 10^4$

Формат выходных данных

Файл с решением должен содержать реализацию класса AutoBallControl.

Решение

Данная задача требует решения на двух уровнях – алгоритмическом и техническом. Алгоритмически здесь требуется обойти лабиринт при помощи любого стандартного метода, например поиском в ширину.

Технически необходимо реализовать модуль в проекте на Unity. Поскольку проверяющая система использует физический движок Unity, участники также должны тестировать свои решения непоседственно в этой среде.

Пример программы-решения

Ниже представлено решение на языке C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using UnityEngine;

public class AutoBallControl : BallControl
{
   private int colCount;
   private int rowCount;
   private int currentPath = 0, currentVertex = 0;
   private float width = 0;
   private float height = 0;

   private readonly List<List<int>> g = new List<List<int>>();
   private readonly List<List<List<int>>> paths = new List<List<List<int>>>();
   private readonly List<List<int>> sequences = new List<List<int>>();

   private const float Step = 0.751f;
   private List<int> visitOrder;

   private int GetVertex(int column, int row)
   {
      return column * rowCount + row;
   }

   private List<int> GetPath(int u, int v)
   {
      var dists = new List<int>();
      dists.AddRange(Enumerable.Repeat(0, colCount * rowCount));
      var prev = new List<int>();
      prev.AddRange(Enumerable.Repeat(0, colCount * rowCount));
      for (var i = 0; i < dists.Count; ++i)
         dists[i] = 10000000;
      dists[u] = 0;
      prev[u] = -1;

      var q = new Queue<int>();
      q.Enqueue(u);
```

```
39          while (q.Count > 0) {
40              var w = q.Dequeue();
41              for (var i = 0; i < g[w].Count; ++i) {
42                  if (dists[g[w][i]] > dists[w] + 1) {
43                      dists[g[w][i]] = dists[w] + 1;
44                      prev[g[w][i]] = w;
45                      q.Enqueue(g[w][i]);
46                  }
47              }
48          }
49
50          var path = new List<int>();
51          var t = v;
52          while (t != -1) {
53              path.Add(t);
54              t = prev[t];
55          }
56
57          path.Reverse();
58          return path;
59      }
60
61      public override void SetMaze()
62      {
63          width = MazeDescription.CellWidth;
64          height = MazeDescription.CellHeight;
65
66          var maxCol = 0;
67          var maxRow = 0;
68          foreach (var cell in MazeDescription.Cells) {
69              maxCol = Math.Max(cell.Column, maxCol);
70              maxRow = Math.Max(cell.Row, maxRow);
71          }
72
73          rowCount = maxRow + 1;
74          colCount = maxCol + 1;
75
76          var c = new List<int>();
77          for (var i = 0; i < colCount * rowCount; ++i)
78              g.Add(new List<int>());
79          foreach (var cell in MazeDescription.Cells) {
80              var v = GetVertex(cell.Column, cell.Row);
81              if (cell.CanMoveRight) {
82                  var u = GetVertex(cell.Column + 1, cell.Row);
83                  g[v].Add(u);
84                  g[u].Add(v);
85              }
86
87              if (cell.CanMoveForward) {
88                  var u = GetVertex(cell.Column, cell.Row + 1);
89                  g[v].Add(u);
90                  g[u].Add(v);
91              }
92
93              if (cell.HasCoin || cell.Row == 0 && cell.Column == 0)
94                  c.Add(v);
95          }
96
97          for (var i = 0; i < c.Count; ++i) {
98              paths.Add(new List<List<int>>());
```

```
99          for (var j = 0; j < c.Count; ++j)
100             paths[i].Add(new List<int>());
101      }
102
103      for (var i = 0; i < c.Count; ++i) {
104         for (var j = 0; j < c.Count; ++j) {
105            if (i == j) continue; // TODO
106            var path = GetPath(c[i], c[j]);
107            paths[i][j] = path;
108         }
109      }
110
111      visitOrder = GetBestPathOrder();
112   }
113
114   private KeyValuePair<int, int> GetRowColumn(int vertex)
115   {
116      return new KeyValuePair<int, int>(vertex % rowCount, vertex / rowCount);
117   }
118
119   private List<int> GetBestPathOrder()
120   {
121      var visited = new List<bool>();
122      visited.AddRange(Enumerable.Repeat(false, paths.Count));
123      visited[0] = true;
124      var initialSequence = new Stack<int>();
125      initialSequence.Push(0);
126      GenerateAllPathOrderVariants(visited, initialSequence);
127
128      List<int> bestPathOrder = null;
129      var bestCost = 0;
130      foreach (var sequence in sequences) {
131         var cost = 0;
132         for (var j = 0; j < sequence.Count - 1; ++j)
133            cost += paths[sequence[j]][sequence[j + 1]].Count;
134
135         if (bestPathOrder == null || cost < bestCost) {
136            bestCost = cost;
137            bestPathOrder = sequence;
138         }
139      }
140      Debug.Assert(bestPathOrder != null);
141
142      return bestPathOrder;
143   }
144
145   private static List<int> PathStackToList(IEnumerable<int> pathStack)
146   {
147      var clonedStack = new Stack<int>(new Stack<int>(pathStack));
148      var path = new List<int>();
149      while (clonedStack.Count > 0)
150         path.Add(clonedStack.Pop());
151      path.Reverse();
152      return path;
153   }
154
155   private void GenerateAllPathOrderVariants(IList<bool> visited, Stack<int> sequence)
156   {
157      if (sequence.Count == visited.Count) {
158         sequences.Add(PathStackToList(sequence));
```

```
159             return;
160         }
161
162         for (var i = 0; i < paths.Count; ++i) {
163             if (!visited[i]) {
164                 visited[i] = true;
165                 sequence.Push(i);
166                 GenerateAllPathOrderVariants(visited, sequence);
167                 sequence.Pop();
168                 visited[i] = false;
169             }
170         }
171     }
172
173     private void DebugPrintPath()
174     {
175         for (var i = 0; i < paths.Count - 1; ++i) {
176             for (var j = 0; j < paths[i][i + 1].Count; ++j) {
177                 var rowCol = GetRowColumn(paths[i][i + 1][j]);
178             }
179         }
180     }
181
182     public override int GetMove(float x, float y)
183     {
184         if (currentPath == paths.Count - 1)
185             return 0;
186         var path = paths[visitOrder[currentPath]][visitOrder[currentPath + 1]];
187         var rowCol = GetRowColumn(path[currentVertex]);
188         var targetY = rowCol.Key * height;
189         var targetX = rowCol.Value * width;
190
191         var dx = x - targetX;
192         var dy = y - targetY;
193         if (dx * dx + dy * dy < Step * Step) {
194             if (path[currentVertex] == path[path.Count - 1]) {
195                 ++currentPath;
196                 currentVertex = 0;
197             }
198             else
199                 ++currentVertex;
200         }
201
202         if (Math.Abs(dx) > Math.Abs(dy))
203             return dx > 0 ? MoveTypeLeft : MoveTypeRight;
204         return dy > 0 ? MoveTypeBottom : MoveTypeTop;
205     }
206 }
```