Технологии дополненной реальности

2022/23 учебный год

Второй отборочный этап

Задачи второго этапа инженерного тура по профилю «Технологии дополненной реальности» направлены на формирование комплекса компетенций, позволяющих успешно выполнить задания и реализовать проектную задачу финала.

Тема заданий второго этапа — «Сумма технологий». Речь пойдет о сумме технологий необходимых для реализации комплексной инженерной задачи на базе webAR: прикрепление объектов к маркеру, создание мультимаркерной системы, способы обеспечения интерактивности — взаимодействие с виртуальными объектами, расположение объектов на внешних и внутренних гранях куба и на плоскости, динамическая загрузка информации в зависимости от местоположения пользователя, совсем немного дизайна и трехмерного моделирования.

Все задачи разделены на два блока: базовые и расширенные.

В базовом блоке расположены очень простые задачи, предполагающие, что каждый участник финала и член команды должен владеть набором умений, позволяющих создавать автономные эффекты с использованием webAR-технологий. Этот блок задач участники могут проходить в индивидуальном порядке.

В расширенной части представлены задания, которые носят проектный характер — это задачи на навыки разработки сложных комплексных решений, в данном блоке потребуется усилие нескольких сленов команды. С технической точки зрения, к простым автономным AR-эффектам требуется добавить сценарии взаимодействия пользователя с виртуальными объектами, средой, цифровыми моделями друг с другом. Цифровые модели оверлеев, представленные во второй части заданий должны иметь уникальный авторский дизайн.

Базовые компетенции второго этапа.

ПК-1 Способность использовать средства разработки программного обеспечения (языки программирования, базы данных) и языки разметки сетевого контента для реализации веб-проектов с внедрением технологий дополненной реальности.

Рекомендуемые технологии: HTML, JS, AR-JS, A-Frame, Three.js.

ПК-2 Способность использовать инструменты компьютерной графики и анимации необходимые для формирования авторских маркеров и создания трехмерных цифровых моделей объектов

Рекомендуемые технологии: Blender.

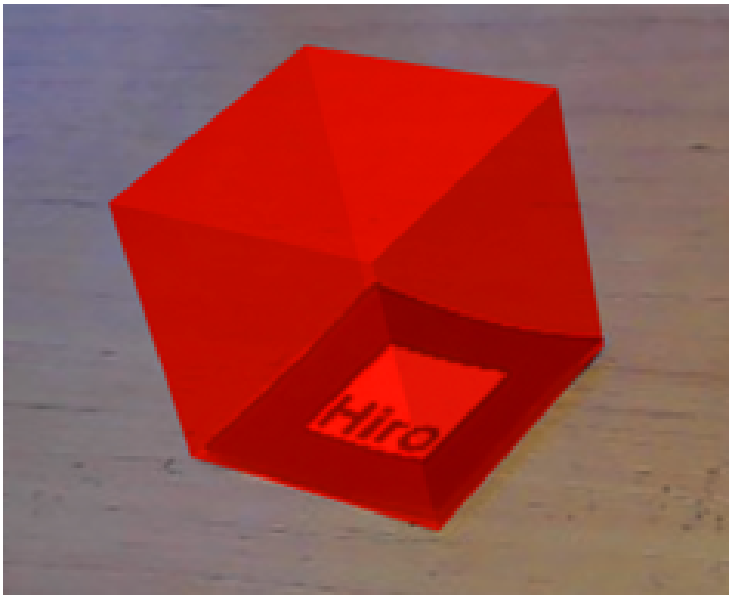
ПК-3 Способность проектировать и создавать сложные мультифункциональные веб-проекты с использованием технологий дополненной реальности.

Задания базового уровня

Задача IV.1.1. Триггер и оверлей (4 балла)

Темы: маркер, триггер, оверлей, HTML, A-Frame.

Условие



На картинке представлен AR-сценарий привязки трехмерного объекта (куба) к предопределенному маркеру (`Hiro`) и его отображение при наведении устройства камеры на маркер.

Напишите код к такому сценарию, используя библиотеку A-Frame.

В ответе отметьте минимальный набор необходимых для решения данной задачи тегов.

Варианты тегов:

- `a-plane`
- `a-gltf-model`
- `a-asset-item`
- `a-cursor`
- `a-asset`
- `a-image`
- `a-box`
- `a-scene`
- `a-video`
- `a-camera`
- `a-marker`

Решение

```
<html>
  <head>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script
 src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-ar-nft.js">
    </script>
  </head>
  <body>
```

```
    <!--Первым тегом необходимо обозначить пространство(сцену), на которой будет
    ↪  отображаться контент дополненной реальности -->
    <a-scene embedded arjs>
      <!-- Далее необходим триггер для отображения дополненной реальности -->
      <!--В качестве триггера AR, по условию задачи, используется стандартный маркер
      ↪  Hiro-->
      <a-marker preset="hiro">
        <!-- Теперь необходимо обозначить оверлей, которым должен быть красный куб -->
        <a-box position='0 0.5 0'
         material='opacity: 0.5; side: double;color:red;'>
        </a-box>
       </a-marker>
<!-- После описания маркера и оверлея обязательным компонентом для дополненной
↪  реальности является камера-->
      <a-entity camera></a-entity>
    </a-scene>
  </body>
</html>
```

Таким образом обязательными тегами для реализации данной задачи являются: `a-scene`, `a-box`, `a-marker`, `a-camera`.

Ответ: `a-scene`, `a-box`, `a-marker`, `a-camera`.

Задача IV.1.2. Текст и простые объекты (4 балла)

Темы: маркер, триггер, оверлей, HTML, A-Frame.

Условие

Опираясь на решение предыдущей задачи напишите теги, позволяющие заменить куб на сферу.

Из представленных ниже блоков, соберите программу. Укажите правильную последовательность блоков в программе (номера нужных блоков через пробел). Учтите, что в представленных блоках могут быть лишние или ошибочные варианты.

№	Блок кода

1.	`<html>`
	`<head>`
	`<script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>`
	`<script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/a`
	`↪ frame-ar.js"></script>`
	`</head> <body>`

2.	`<a-scene embedded arjs>`

3.	`<html>`
	`<head>`
	`<body>`

4.	`</a-scene>`

5.	`<a-marker>`
	`<a-sphere`
	`radius="0.5"`
	`position="0 -1 -1"`
	`color="red">`
	`</a-sphere>`
	`</a-marker>`

№	Блок кода

6.	`<a-marker preset='hiro'>`
	`<a-box`
	`width="1"`
	`height="1"`
	`position="0 -1 -1"`
	`color="red">`
	`</a-box>`
	`</a-marker>`

7.	`<a-entity>`
8.	`<a-entity>`
9.	`</body>` `</html>`

10.	`</body>`
	`<head>`
	`<script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>`
	`<script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/a`
	`↪ frame-ar.js"></script>`
	`</head>`
	`</html>`

11.	`<a-marker preset="hiro">`
	`<a-sphere`
	`radius="0.5"`
	`position="0 -1 -1"`
	`color="red">`
	`</a-sphere>`
	`</a-marker>`

12.	`<a-marker preset="hiro">`
	`<a-sphere`
	`radius="0.5"`
	`position="0 -1 -1"`
	`color="red">`
	`</a-sphere>`
	`</a-marker>`

Пример ответа: 2 1.

Что означает, что программа состоит из двух блоков, которые нужно соединить в указанной последовательность (т. е. первым идет второй блок, а затем блок под номером 1).

Решение

Для определения порядка блоков необходимо знать основных правил верстки `html` страниц:

- Тег `<html>` является контейнером, который заключает в себе все содержимое веб-страницы. Закрытие данного тега идет в самом конце веб-страницы.
- Тег `<head>` предназначен для указания технической информации и подгрузки дополнительных скриптов и должен идти перед тегом `<body>`
- В теге `<body>` размещается контент страницы.

Таким образом можно исключить блоки 3 9 10 как ошибочные и поставить на первое место блок № 1.

Для дальнейшего определения последовательности блоков необходимо знание AR-тегов и их порядок в коде:

- Сперва необходимо обозначить пространство, в которой будет размещен контент (тег `<a-scene>`)
- Далее описать триггер (тег `<a-marker>`) и оверлей (теги `a-entity/a-box/a-shape` и т. п.). По условию задачи нам необходимо отображение сферы на маркере Hiro, для этого необходим тег `<a-shape>`.
- Добавить на сцену камеру (тег `<a-camera>`).
- Закрыть тег `</a-scene>`.

Исходя из перечисленного выше последовательность блоков для реализации AR следующая: 2 11 12 4.

Последним блоком необходимо закрыть теги, открытые ранее, для этого подходит блок 9.

Получившаяся последовательность 1 2 11 12 4 9.

Полный код решения

```
<html>
    <head>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script src=
    "https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-ar-nft.js">
    </script>
    </head>
    <body>
        <a-scene embedded arjs>
            <a-marker preset="hiro">
                <a-sphere
                  position="0 1.25 -5"
                  radius="1.25"
                  color="red">
                </a-sphere>
            </a-marker>
            <a-entity camera></a-entity>
        </a-scene>
    </body>
</html>
```

Ответ: 1 2 11 12 4 9.

Задача IV.1.3. Объекты сложной формы (4 балла)

Темы: маркер, триггер, оверлей, HTML, A-Frame.

Условие

`Ar.js` может отображать сложные трехмерные объекты, в том числе и с авторским дизайном, определенные в формате `glTF`.

Для этого используются тег `<a-entity>` с атрибутом «`gltf-model`», в который необходимо передать ссылку на объект.

Отметьте минимальный необходимый набор атрибутов тега `<a-entity>`, чтобы отобразить модель в качестве оверлея для маркера `hiro`.

Варианты ответов:

- ```
  def generate_trajectory(R=5,N=10, H=64, W=64):
      directions = [(0,1), (0, -1), (1, 0), (-1,0)]
      trajectory = []
      images = []
      start_img = np.zeros((H,W))
      coords = (random.randint(R,H-R), random.randint(R,W-R))
      mask = draw_circle(start_img, R, coord=coords)
      start_img[mask] = 1
      images.append(start_img)
      for i in range(N-1):
          dir_id = random.randint(0,3)
          new_direction = directions[dir_id]
          new_circle = np.zeros((H,W))
          if not (R < coords[0] + new_direction[0] < H-R):
              new_direction = (-new_direction[0],new_direction[1])

          if not (R < coords[1] + new_direction[1] < W-R):
              new_direction = (new_direction[0],-new_direction[1])

          coords = (coords[0] + new_direction[0], coords[1] + new_direction[1])
          mask = draw_circle(new_circle, R, coord=coords)
          new_circle[mask] = 1
          images.append(new_circle)
          trajectory.append(new_direction)
      return images, trajectory
  ```
- ```
  <a-entity sition="0 -1 0" glf-model="#example" ></a-entity>
  ```
- ```
  <a-entity sition="0 -1 0" glf-model="#example" rotation="90 0 -90"></a-entity>
  ```
- ```
  <html>
      <head>
          <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
          <script src=
          "https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-ar-nft.js">
          </script>
      </head>
      <body>
          <a-assets>
              <a-asset-item id="example" src="/src/example.glf"></a-asset-item>
          </a-assets>
          <a-scene embedded arjs>
              <a-marker preset="hiro">
                  <a-entity position="0 -1 0" glf-model="#example"></a-entity>
              </a-marker>
              <a-entity camera></a-entity>
          </a-scene>
      </body>
  </html>
  ```

Решение

Тег a-entity имеет несколько свойств:

- `glf-model` — ссылка на объект;
- `sition` — смещение позиции объекта относительно начальной;
- `rotation` — поворот объекта относительно начальной позиции.

Важным моментом является то, что `glf` объект должен быть подгружен заранее

в теге `a-assets`, где ему задается `id`, для дальнейшего использования в коде через `#`.

Обязательным атрибутом тема `<a-entity>` для отображения модели является только ссылка на объект.

Пример кода с подгрузкой объекта в формате `glTF`.

```
<img id="img1" src="....">
```

Ответ: `<a-entity glf-model="#example»`.

Задача IV.1.4. Встроенный звук (4 балла)

Темы: маркер, триггер, оверлей, HTML, A-Frame.

Условие

Дополненная реальность в вашем проекте может носить аудиовизуальный характер, т. е. появляющемуся в поле зрения камеры объекту можно добавить звуковое сопровождение.

Ниже представлен код для загрузки звукового сопровождения, возникающего при наведении устройства на маркер, однако он не работает.

Найдите ошибки и исправьте код. В ответе укажите номера строк через пробел, в которых были допущены ошибки, в порядке их обнаружения в коде.

- Если вы думаете, что ошибок нет, напишите 0.
- Если ошибка может быть в одном из двух мест, то напишите номер верхней строки, в которой она может быть.

Например, есть код с загрузкой картинки, где ей был дан `id="img1"`,

```
<img id="img1" src="....">
```

позже в коде встречается ее использование в качестве наполнения, но допущена ошибка и использовано другое `id`

```
<html>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script src=
    "https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-ar.js">
    </script>
    <script>
        AFRAME.registerComponent('soundhandler', {
            tick: function () {
                var entity = document.querySelector('[sound]');

                if (document.querySelector('a-marker').object3D.visible == true){
                    entity.components.sound.playSound();

                } else {
                    entity.components.sound.pauseSound();
                }
            }
        });
```

```
    </script>
    <body>
        <a-scene embedded arjs soundhandler>
            <a-assets>
                <img crossorigin="anonymous" id="img" src="https://i.imgur.com/xDzgIs6.jpeg">
                <audio  id="audio" src ="audio.mp3" crossorigin="anonymous"></audio>
            </a-assets>

            <a-marker preset='kanji'>
                <a-plane src="#img" height="1" width ="1" rotation="-90 0 0">
                    <a-sound src="#audio"></a-sound>
                </a-plane>
            </a-marker>
            <a-entity camera></a-entity>
        </a-scene>
    </body>
</html>
```

Ошибка может быть как в назначении `id`, мы могли обозначить `id="img"`, так и в строке, где она используется, можно указать `src="#img1"`. В таких случаях следует указать в качестве ошибки первую строку

Пример ответа: 1 2 7.

Что соответствует тому, что в коде были допущены ошибки на строках 1, 2 и 7.

Решение

При просмотре структуры и порядка открытия и закрытия тегов можно обнаружить, что на строке 27 есть два непарных тега: `<a-sound>` и `</a-entity>`. Прикрепление звука возможно и тем и другим тегом, но дальнейшие параметры указывают на использование тега `<a-sound>` и соответветственно вместо `</a-entity>` должно быть закрытие тега `</a-sound>`.

Кроме структуры необходимо посмотреть еще и на названия `id` и `class` компонентов. На строке номер 5 обязательно должно быть указано название сцены, которое также прописывается в данном коде на строке 19. И так как по условию задачи, если ошибка возможна в двух местах, то необходимо указать первую строку, возможная ошибка может быть на строке 5.

На строке 9 прописывается функция слежения за тем, найден ли маркер на текущий момент времени. Поэтому, там обязательно нужно прописывать «`querySelector('a-marker')`» в условии. И если маркер найден, то проигрывать аудио: за это отвечает строка 10 в конце которой должно быть прописано «`playSound();`». Если же маркер пропал из поля зрения камеры, то воспроизведение нужно останавливать, это контролирует строка 13, на конце которой должно быть прописано «`pauseSound();`».

Ответ: задача генеративная, верный ответ будет варьироваться. При этом строка 27 всегда содержит ошибку и должна присутствовать в любом правильном ответе.

Верный код.

```
<html>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script src=
```

```
    "https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-ar.js">
    </script>
    <script>
        AFRAME.registerComponent('soundhandler', {
            tick: function () {
                var entity = document.querySelector('[sound]');

                if (document.querySelector('a-marker').object3D.visible == true){
                    entity.components.sound.playSound();

                } else {
                    entity.components.sound.pauseSound();
                }
            }
        });
    </script>
    <body>
        <a-scene embedded arjs soundhandler>
            <a-assets>
                <img crossorigin="anonymous" id="img"
                ↪  src="https://i.imgur.com/xDzgIs6.jpeg">
                <audio  id="audio" src ="audio.mp3" crossorigin="anonymous"></audio>
            </a-assets>

            <a-marker preset='kanji'>
                <a-plane src="#img" height="1" width ="1" rotation="-90 0 0">
                    <a-sound src="#audio"></a-sound>
                </a-plane>
            </a-marker>
            <a-entity camera></a-entity>
        </a-scene>
    </body>
</html>
```

Задача IV.1.5. Встроенное видео (4 балла)

Темы: маркер, триггер, оверлей, HTML, A-Frame.

Условие

Написан код, который поверх маркера размещает объект с текстурой в форме видео. Видео должно полностью перекрывать маркер. Пример доступен по ссылке: https://disk.yandex.ru/i/ziWOb_IMct_xBQ.

Из представленных ниже блоков, соберите программу. Укажите правильную последовательность блоков в программе (номера нужных блоков без пробелов). Учтите, что в представленных блоках могут быть лишние или ошибочные варианты.

№	Блок кода

1.	`<html>`
	`<head>`
	`  <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>`
	`  <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/a`
	`  ↪ frame-ar-nft.js"></script>`
	`</head>`
	`<body>`

№	Блок кода

2.	marker.addEventListener('markerFound', function() {
	    var markerId = marker.id;
	    console.log('markerFound', markerId);
	    video.play();
	});
	
	marker.addEventListener('markerLost', function() {
	    var markerId = marker.id;
	    console.log('markerLost', markerId);
	    video.pause();
	});
	
	});
	</script>

3.	marker.addEventListener('markerLost', function() {
	    var markerId = marker.id;
	    console.log('markerLost', markerId);
	    video.play();
	});
	
	marker.addEventListener('markerFound', function() {
	    var markerId = marker.id;
	    console.log('markerFound', markerId);
	    video.pause();
	});

4.	<a-scene embedded ajs> <a-assets> <video id="video" src="sintel.mp4" loop="true"><video> </a-assets>

5.	<script>
	  document.addEventListener("DOMContentLoaded", () =>{
	      let marker = document.querySelector("#marker");
	      let video = document.querySelector("#video");

6.	</a-scene> </body> </html>
7.	<a-marker id="marker" preset="hiro"> <a-video src="#video" rotation="90 0 0"></a-video> </a-marker> <a-entity camera></entity>
8.	<a-marker id="marker" preset="hiro"> <a-video src="#video" rotation="-90 0 0"></a-video> </a-marker> <a-entity camera></entity>

Пример ответа: 2 1.

Что означает, что программа состоит из двух блоков, которые нужно соединить в указанной последовательность (т. е. первым идет второй блок, а затем блок под номером 1).

Решение

Для определения порядка блоков необходимо знать основных правил верстки `html` страниц:

- Тег `<html>` является контейнером, который заключает в себе все содержимое

веб-страницы. Закрытие данного тега идет в самом конце веб-страницы.

- Тег `<head>` предназначен для указания технической информации и подгрузки дополнительных скриптов и должен идти перед тегом `<body>`
- В теге `<body>` размещается контент страницы.

В данном примере только один блок содержит все необходимое: блок № 1. При этом другие блоки не относятся к данным правилам и исключать их пока нельзя.

В предложенных блоках кода присутствует тег `<script>`, который может быть расположен в любом месте веб-страницы, но принято его прописывать перед AR-сценой, поэтому блок № 5 будет вторым. После которого может стоять либо блок № 2, либо блок № 3. В блоке № 3 допущено несколько ошибок: перепутаны команды запуска и паузы, нет закрытия тега `<script>`. Следовательно, следующий верный блок № 2.

Для дальнейшего определения последовательности блоков необходимо знание AR тегов и их порядок в коде:

- сперва необходимо обозначить пространство, в которой будет размещен контент (тег `<a-scene>`);
- прописать необходимые ресурсы и id к ним в теге `<a-assets>`;
- далее описать триггер (тег `<a-marker>`) и оверлей (`<a-video>`.);
- добавить на сцену камеру (тег `<a-camera>`);
- закрыть тег `</a-scene>`.

Исходя из перечисленного выше последовательность блоков для реализации AR следующая: 4 8 или 4 7.

При кода тестировании становится понятно, что положение видео при повороте по оси x на 90 неверное, соответственно блок № 7 ошибочный.

Последним блоком необходимо закрыть теги, открытые ранее, для этого подходит блок 6.

Получившаяся последовательность 1 5 2 4 8 6.

Полный код решения

```
<html>
<head>
    <script src=
    "https://aframe.io/releases/1.3.0/aframe.min.js">
    </script>
    <script src=
    "https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-ar-nft.js">
    </script>
</head>
<body>
    <script>
        document.addEventListener("DOMContentLoaded", () =>{
            let marker = document.querySelector("#marker");
            let video = document.querySelector("#video");

            marker.addEventListener('markerFound', function() {
                var markerId = marker.id;
                console.log('markerFound', markerId);
                video.play();
            });
```

```
            marker.addEventListener('markerLost', function() {
                var markerId = marker.id;
                console.log('markerLost', markerId);
                video.pause();
            });

        });
    </script>
        <a-scene embedded ajs>
            <a-assets>
                <video id="video" src="sintel.mp4"  loop="true"><video>
            </a-assets>

            <a-marker id="marker" preset="hiro">
                <a-video src="#video" rotation="-90 0 0"></a-video>
            </a-marker>
            <a-entity camera></entity>
        </a-scene>

</body>
</html>
```

Ответ: 1 5 2 4 8 6.

Задача IV.1.6. Мультимаркер (4 балла)

Темы: маркер, триггер, оверлей, HTML, A-Frame.

Условие

У нас может быть несколько маркеров в одной и той же AR-среде — мультимаркер, и каждый объект будет показан на соответствующем маркере.

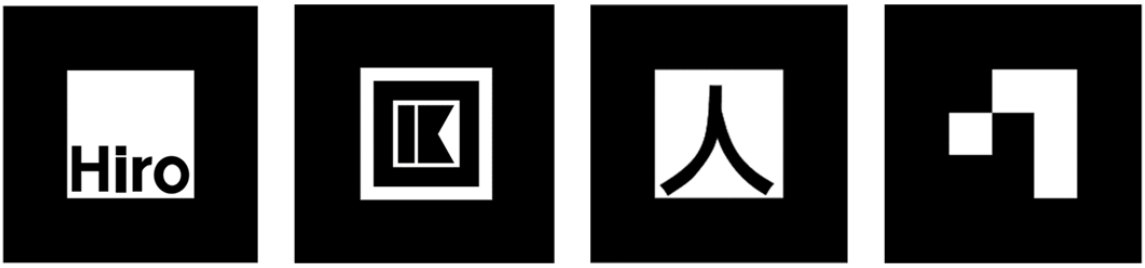
Ниже представлены используемые маркеры и блоки кода, которые были вставлены в следующий шаблон:

```
<html>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-a
    ↪  r.js"></script>
    <body style="margin : 0px; overflow: hidden;">
        <a-scene embedded arjs='sourceType: webcam; detectionMode: mono_and_matrix;
        ↪  matrixCodeType: 3x3;'>

            <!-- Сюда вставить блоки кода-->

            <a-entity camera></a-entity>
        </a-scene>
    </body>
</html>
```

Соотнесите каждый маркер с блоком кода, в котором написан сценарий, инициирующий его использование.



Блок кода 1

```
<a-marker preset="custom" type='pattern' url='pattern-custom.patt'>
    <a-box depth="1" height="1" width="1" position='0 0.5 0' material='opacity:0.5;
    ↪  side:double; color:green;'></a-box>
</a-marker>
```

Блок кода 2

```
<a-marker preset='hiro'>
   <a-torus-knot radius='0.26' radius-tubular='0.05'
   animation="property: rotation; to:360 0 0; dur: 5000; easing: linear; loop: true">
   </a-torus-knot>
</a-marker>
```

Блок кода 3

```
<a-marker preset='kanji'>
   <a-box position='0 0.5 0' material='opacity: 0.5; side: double;color:green;'>
      <a-torus-knot radius='0.26' radius-tubular='0.05'
      animation="property: rotation; to:360 0 0; dur: 5000; easing: linear;loop:
      ↪  true">
      </a-torus-knot>
   </a-box>
</a-marker>
```

Блок кода 4

```
<a-marker preset='kanji'>
   <a-box position='0 0.5 0' material='opacity: 0.5; side: double;color:green;'>
      <a-torus-knot radius='0.26' radius-tubular='0.05'
      animation="property: rotation; to:360 0 0; dur: 5000; easing: linear;loop:
      ↪  true">
      </a-torus-knot>
   </a-box>
</a-marker>
```

Решение

Для решения данной задачи необходимо обратить внимание на тег `<a-marker>` и его параметр `type` или `preset`, в котором указывается тип маркера или название.

В библиотеке `AR.js` представлены три типа маркеров:

- стандартные (например, `Hiro`);
- `Barcode`;

- `Pattern` (собственные маркеры).

В задаче даны два стандартных маркера, а именно: `hiro`, `kanji`, которые задаются через параметр preset и указание имени маркера. Таким образом маркеру № 1 соответствует блок кода № 2, а маркеру № 3 блок кода № 4.

Тип маркера `Barcode` представляет из себя составленные фигуры, похожие на фигуры из тетриса, с рамкой в виде квадрата, всего вариантов `Barcode 32`. Тип маркера указывается через свойство `type='barcode'` а вариант через свойство `value`. В примере использован 5 вариант такого типа маркеров. Маркеру № 4 соответствует блок кода № 3.

Второй маркер является пользовательским, то есть он был сгенерированным из изображения пользователя с помощью специального инструмента в формате `.patt`. В таком типе маркеров указываются свойства: `preset="custom" type='pattern'`. Сгенерированный маркер необходимо разместить в папке проекта и в свойстве `url` указать путь к нему.

Пример кода

```
<html>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script
src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-ar.js"></scri
↪  pt>
    <body style="margin : 0px; overflow: hidden;">
        <a-scene embedded arjs='sourceType: webcam; detectionMode: mono_and_matrix;
        ↪  matrixCodeType: 3x3;'>
{блок кода 1
            <a-marker preset="custom" type='pattern' url='pattern-custom.patt'>
                <a-box depth="1" height="1" width="1" position='0 0.5 0'
                ↪  material='opacity:0.5; side:double; color:green;'></a-box>
            </a-marker>
}
{блок кода 2
            <a-marker preset='hiro'>
                <a-torus-knot radius='0.26' radius-tubular='0.05'
                animation="property: rotation; to:360 0 0; dur: 5000; easing: linear;
                ↪  loop: true">
                </a-torus-knot>
            </a-marker>
}
{блок кода 3
            <a-marker type='barcode' value='5'>
                <a-sphere
                scale="1 1 1"
                color="red"
                >
                </a-sphere>
            </a-marker>
 }
{блок кода 4
            <a-marker preset='kanji'>
                <a-box position='0 0.5 0' material='opacity: 0.5; side:
                ↪  double;color:green;'>
                    <a-torus-knot radius='0.26' radius-tubular='0.05'
                    animation="property: rotation; to:360 0 0; dur: 5000; easing:
                    ↪  linear;loop: true">
                    </a-torus-knot>
                </a-box>
```

```
            </a-marker>
}
            <a-entity camera></a-entity>
        </a-scene>
    </body>
</html>
```

Ответ:

1. 
2. 

3. 
4. 

Задача IV.1.7. Вращающийся глобус (4 балла)

Темы: маркер, триггер, оверлей, HTML, A-Frame.

Условие

Реализован фрагмент кода, который позволит при помощи веб-ресурса отображать поверх маркера вращающийся глобус. В коде есть пропуски, которые вам необходимо заполнить соответствующим тегом.

Пропуски в шаблоне обозначены «_?_».

Пример кода с подгрузкой объекта в формате `glTF`.

```
<html>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-a
    ↪  r.js"></script>
    <body>
        <a-scene embedded arjs>
            <_?_>
                <img id="globe" _?_="https://i.imgur.com/Ip19PQt.jpeg">
            <_?_>
            <a-marker preset='hiro'>
                <a-sphere  src="_?_" radius="5"  rotation="0 0 0"
         _?_="_?_: rotation; to: 0 360 0; loop: _?_; dur: 10000">
                </a-sphere>
            </a-marker>
            <a-camera camera></a-camera>
        </a-scene>
    </body>
</html>
```

Решение

```
<html>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-a
    ↪  r.js"></script>
```

```
    <body>
        <a-scene embedded arjs>
            <!-- Сперва необходим тег a-assets для загрузки текстуры глобуса-->
            <a-assets>
                 <!-- адрес текстуры указывается через атрибут src-->
                 <img id="globe" src="https://i.imgur.com/Ip19PQt.jpeg">
            <!-- Подгрузка дополнительных материалов закончена, закрываем тег
            ↪  a-assets-->
            </a-assets>
            <a-marker preset='hiro'>
                 <!-- В данном случае необходимо использовать ссылку на загруженный
                 ↪  ранее asset с текстурой. Для этого ставим # и после него
                 ↪  прописываем id, которое соответствует указанному ранее для
                 ↪  текстуры-->
                 <a-sphere  src="#globe" radius="5"  rotation="0 0 0"
                 animation="property: rotation; to: 0 360 0; loop: true; dur: 10000">
                 <!-- По условию задачи глобус должен вращаться, поэтому необходимо
                 ↪  свойство animation и его property для указания направления и
                 ↪  скорости. Для зацикливания анимации необходимо указать в
                 ↪  свойствах loop: true -->
                </a-sphere>
            </a-marker>
            <a-entity camera></a-entity>
        </a-scene>
    </body>
</html>
```

Ответ:

1. `<a-assets>`;
2. `src`;
3. `</a-assets>`;
4. `#globe`;
5. `animation`;
6. `property`;
7. `true`.

Задача IV.1.8. Движущаяся плоскость (4 балла)

Темы: маркер, триггер, оверлей, HTML, A-Frame.

Условие

Реализован фрагмент кода, который позволит с помощью веб-ресурса отображать над маркером плоскость, которая движется вверх и вниз.

Из представленных ниже блоков, соберите программу. Укажите правильную последовательность блоков в программе (номера нужных блоков через пробел). Учтите, что в представленных блоках могут быть лишние или ошибочные варианты.

№	Блок кода

1.	`    console.log("up");`
	`        plane.emit("up");`
	`})`
	`marker.addEventListener('markerLost', function() {`

2.	`document.addEventListener("DOMContentLoaded", () =>{`
	`    let marker = document.querySelector("#marker");`
	`    let plane = document.querySelector("#plane");`
	
	`    marker.addEventListener('markerFound', function() {`
	`        var markerId = marker.id;`

3.	`        var markerId = marker.id;`
	`        console.log('markerLost', markerId);`
	`    });`
	
	`});`
	`</script>`

4.	`     console.log("down");`
	`     plane.emit("down");`
	`})`
	`plane.addEventListener('animationcomplete__2',function(){`

5.	`<a-plane id ="plane" src="#img" position="0 0 0" rotation="-90 0 0"`
	`animation__1="property: position; from:0 0 0; to: 0 5 0; dur: 500; easing:`
	`↪  easeInOutQuad; startEvents: down; stopEvents: down"`
	`animation__2="property: position; from:0 5 0; to: 0 0 0; dur: 500; easing:`
	`↪  easeInOutElastic; startEvents: up; stopEvents: up"></a-plane>`

6.	`<body style="margin : 0px; overflow: hidden;">`
	`    <a-scene embedded arjs>`
	`        <a-assets>`
	`            <img id ="img" src="pattern-custom.png">`
	`        </a-assets>`
	`        <a-marker id= "marker" preset='hiro'>`

7.	`            </a-marker>`
	`            <a-entity camera></a-entity>`
	`        </a-scene>`
	`    </body>`
	`</html>`

8.	`    console.log('markerFound', markerId);`
	`        plane.emit("up");`
	`});`
	`plane.addEventListener('animationcomplete__1',function(){`

9.	`<html>`
	`    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>`
	`    <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/a`
	`    ↪  frame-ar.js"></script>`
	`    <script>`

Пример ответа: 2 1.

Что означает, что программа состоит из двух блоков, которые нужно соединить в указанной последовательность (т. е. первым идет второй блок, а затем блок под номером 1).

Решение

Для определения порядка блоков необходимо знать основных правил верстки `html` страниц:

- Тег `<html>` является контейнером, который заключает в себе все содержимое веб-страницы. Закрытие данного тега идет в самом конце веб-страницы.
- Тег `<head>` предназначен для указания технической информации и подгрузки дополнительных скриптов и должен идти перед тегом `<body>`.
- В теге `<body>` размещается контент страницы.

В данном примере только один блок содержит все необходимое: блок № 9. При этом другие блоки не относятся к данным правилам и исключать их пока нельзя.

В предложенных блоках кода присутствует тег `<script>`, который может быть расположен в любом месте веб-страницы, но принято его прописывать перед AR-сценой. Всего блоков в которых представлен скрипт, а не структура страницы пять: 1 2 3 4 8.

Первым нужно поставить блок № 2, в котором создаются переменные для объектов, которыми в дальнейшем манипулируют. В нем содержится начало функции, которая срабатывает при обнаружении маркера, ее окончание находится в блоке № 8, что можно отследить по логам в консоль. В конце блока № 8 прописано `'animationcomplete__1'`. Оно встречается только в блоке № 5, и в свойствах которого прописано `Start Events: up`, следовательно следующим блоком будет блок № 4, в котором прописанно движение вверх. За ним по тому же принципу, что и выше блок № 1 и блок № 4. Лишних и ошибочных блоков в скриптах нет.

Для дальнейшего определения последовательности блоков необходимо знание AR тегов и их порядок в коде:

- сперва необходимо обозначить пространство, в которой будет размещен контент (тег `<a-scene>`);
- прописать необходимые ресурсы и `id` к ним в теге `<a-assets>`;
- далее описать триггер (тег `<a-marker>`) и оверлей (`<a-plane>`);
- добавить на сцену камеру (тег `<a-camera>`);
- закрыть тег `</a-scene>`.

Исходя из перечисленного выше последовательность блоков для реализации AR следующая: 6 5 7.

Последним блоком необходимо закрыть теги, открытые ранее, для этого подходит блок 7.

Получившаяся последовательность 9 2 8 4 1 3 6 5 7.

Полный код решения

```
<html>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-a
    ↪  r.js"></script>
    <script>
       document.addEventListener("DOMContentLoaded", () =>{
            let marker = document.querySelector("#marker");
            let plane = document.querySelector("#plane");

            marker.addEventListener('markerFound', function() {
                var markerId = marker.id;
                console.log('markerFound', markerId);
                plane.emit("up");
            });
            plane.addEventListener('animationcomplete__1',function(){
```

```
                console.log("down");
                plane.emit("down");
            })
            plane.addEventListener('animationcomplete__2',function(){
               console.log("up");
                plane.emit("up");
            })
                    marker.addEventListener('markerLost', function() {
                var markerId = marker.id;
                console.log('markerLost', markerId);
            });

        });
    </script>
    <body style="margin : 0px; overflow: hidden;">
        <a-scene embedded arjs>
            <a-assets>
                <img id ="img" src="pattern-custom.png">
            </a-assets>
            <a-marker id= "marker" preset='hiro'>

            <a-plane id ="plane" src="#img" position="0 0 0" rotation="-90 0 0"
            animation__1="property: position; from:0 0 0; to: 0 5 0; dur: 500;
            ↪  easing: easeInOutQuad; startEvents: down; stopEvents: down"
            animation__2="property: position; from:0 5 0; to: 0 0 0; dur: 500;
            ↪  easing: easeInOutElastic; startEvents: up; stopEvents: up"></a-plane>

            </a-marker>
            <a-entity camera></a-entity>
        </a-scene>
    </body>
</html>
```

Ответ: 9 2 8 4 1 3 6 5 7.

Задача IV.1.9. Теги и их значение (4 балла)

Темы: маркер, триггер, оверлей, HTML, A-Frame.

Теперь, после решения основных базовых задач, вы уже ознакомились с основными тегами `A-frame`.

Сопоставьте название тега и его функцию.

№	Тег	Функция
1	`a-plane`	Создает плоскую поверхность
2	`a-gltf-model`	Отображает 3D модели с разрешением `gltf/glb`
3	`a-asset-item`	Вызывает загрузчик файла `three.js`, с помощью которого можно загрузить любой вид файла
4	`a-cursor`	Сетка, которая позволяет взаимодействовать со сценой на устройствах, не имеющих ручного контроллера
5	`a-asset`	Используется для корректной предзагрузки все файлов разного типа и их дальнейшего корректного отображения.
6	`a-entity`	Универсальный объект, который может обеспечивать внешний вид, поведение и функционал элементов
7	`a-image`	Отображает изображение на плоской поверхности

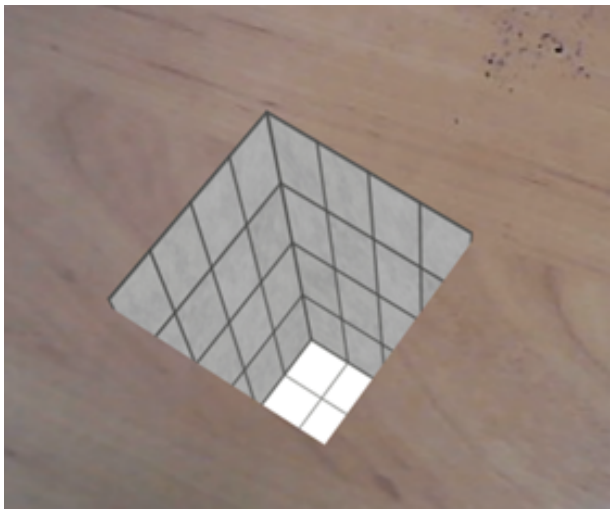
№	Тег	Функция
8	`a-box`	Создает примитивную фигуру куба, с помощью которого можно реализовывать такие объекты, как стены и ящики
9	`a-scene`	Обрабатывает `three.js` и `WebVR/WebXR`, с помощью которых настраивает визуализацию, камеру, освещение, VR-эффект и интерфейс
10	`a-video`	Отображает видео как текстуру на плоской поверхности
11	`a-marker`	Создает объект, который создает относительную сетку координат вокруг распознанного фрагмента изображения
12	`a-torus-knot`	Создает примитивный объект, который по форме напоминает крендель
13	`a-sphere`	Создает примитивный объект сферической формы, с помощью которого можно реализовать такие объекты, как мячи и планеты

Ответ: представлен в таблице выше.

Задача IV.1.10. Дыра в полу (коробка) (4 балла)

Темы: маркер, триггер, оверлей, HTML, A-Frame.

Условие



Из представленных ниже блоков, соберите код, который бы воспроизводил сценарий, по которому в месте размещения маркера появляется дыра кубической формы.

Укажите правильную последовательность блоков в программе (номера нужных блоков через пробел). Учтите, что в представленных блоках могут быть лишние или ошибочные варианты.

№	Блок кода

1.	`<html>`
	`<head>`
	`<script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>`
	`<script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/a`
	`↪ frame-ar-nft.js"></script>`
	`</head>`
	`<body>`

2.	`<a-entity camera></entity>`

3.	`<a-assets>`
	`<img id="tiles" src="tiles.jpg" ></img>`
	`</a-assets>`

4.	`<a-assets>`
	`<img id="video" src="vide.mp4" ></img>`
	`</a-assets>`

5.	`<a-box src="#tiles" material="side:back" position="0 -1.5 0" scale="1.2 3 1.2">`

6.	`<script>`
	`document.addEventListener("DOMContentLoaded", () =>{`
	`let marker = document.querySelector("#marker");`
	`let plane = document.querySelector("#plane");`
	
	`marker.addEventListener('markerFound', function() {`
	`var markerId = marker.id;`
	`console.log('markerFound', markerId);`
	`video.play();`
	`});`
	
	`marker.addEventListener('markerLost', function() {`
	`var markerId = marker.id;`
	`console.log('markerLost', markerId);`
	`video.pause();`
	`});`
	
	`});`
	`</script>`

7.	`<a-box color="black"`
	`material="side:front; shader:portal; blending:subtractive"`
	`segments-depth="1"`
	`segments-height="1"`
	`segments-width="0">`
	`</a-box>`
	
	`<a-box color="black"`
	`material="side:front; shader:portal; blending:subtractive"`
	`rotation="0 90 0"`
	`segments-depth="1"`
	`segments-height="1"`
	`segments-width="0">`
	`</a-box>`

8.	`</a-marker>`
9.	`<a-scene embedded ajs>`

10.	`</a-scene>`
	`</body>`
	`</html>`

11.	`<a-marker id="marker" preset="hiro">`

Пример ответа: 2 1.

Что означает, что программа состоит из двух блоков, которые нужно соединить

в указанной последовательность (т. е. первым идет второй блок, а затем блок под номером 1).

Решение

Для определения порядка блоков необходимо знать основных правил верстки `html` страниц:

- Тег `<html>` является контейнером, который заключает в себе все содержимое веб-страницы. Закрытие данного тега идет в самом конце веб-страницы.
- Тег `<head>` предназначен для указания технической информации и подгрузки дополнительных скриптов и должен идти перед тегом `<body>`
- В теге `<body>` размещается контент страницы.

В данном примере только один блок содержит все необходимое: блок № 1.

Представленный в блоке № 6 скрипт относится к воспроизведению видео, которое не требуется для данной задачи, а значит он лишний.

Для дальнейшего определения последовательности блоков необходимо знание AR тегов и их порядок в коде:

- сперва необходимо обозначить пространство, в которой будет размещен контент (тег `<a-scene>`);
- прописать необходимые ресурсы и `id` к ним в теге `<a-assets>`;
- далее описать триггер (тег `<a-marker>`) и оверлей (`<a-box>`);
- добавить на сцену камеру (тег `<a-camera>`);
- закрыть тег `</a-scene>`.

Можно исключить блок № 4, так как в нем идет загрузка видео, которое не нужно в данной задаче.

Исходя из перечисленного выше последовательность блоков для реализации AR следующая: 9 3 11 5 7 8 2.

Последним блоком необходимо закрыть теги, открытые ранее, для этого подходит блок 10.

Получившаяся последовательность 1 9 3 11 5 7 8 2 10.

Полный код решения

```
<html>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-a
    ↪  r.js"></script>
    <script src="https://stemkoski.github.io/A-Frame-Examples/js/aframe-spritesheet-a
    ↪  nimation.js"></script>
    <body style="margin : 0px; overflow: hidden;">
        <a-scene embedded arjs>
            <a-assets>
                <img id="animate" src="https://stemkoski.github.io/A-Frame-Examples/i
                ↪  mages/spritesheet-test.png">
            </a-assets>
            <a-marker preset='hiro'>
                <!--Допишите код-->
            </a-marker>
            <a-entity camera></a-entity>
```

```
        </a-scene>
    </body>
</html>
```

Ответ: 1 9 3 11 5 7 8 2 10.

Задача IV.1.11. Спрайтовая анимация (4 балла)

Темы: маркер, триггер, оверлей, HTML, A-Frame.

Условие

Дополните приведенный ниже шаблон кода собственным фрагментом, который позволит с помощью веб-ресурса имитировать в месте расположения маркера появление объекта с заданной анимацией.

Анимация выполнена со следующими условиями:

- построена на основе текстурного атласа, прикрепленного ниже;
- длительность кадра составляет 0,08;
- стартовое значение `rotation` объекта не стоит менять;
- остальные параметры следует подобрать исходя из представленного видео фрагмента.

Видео результата доступно по ссылке: https://disk.yandex.ru/i/BAVgjpy4y9sAEw.

Рис. IV.1.1. Текстурный атлас

```
<html>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-a
    ↪  r.js"></script>
    <script src="https://stemkoski.github.io/A-Frame-Examples/js/aframe-spritesheet-a
    ↪  nimation.js"></script>
    <body style="margin : 0px; overflow: hidden;">
        <a-scene embedded arjs>
            <a-assets>
                <img id="animate" src="https://stemkoski.github.io/A-Frame-Examples/i
                ↪  mages/spritesheet-test.png">
            </a-assets>
            <a-marker preset='hiro'>
                <a-box position="0 0 0"
                material="shader: flat; src: #animate; transparent: true; "
```

```
                spritesheet-animation="rows: 2; columns: 8; frameDuration: 0.08;
                ↪  loop: true;"
                animation="property: rotation; to: 0 -360 0; loop: true; easing:
                ↪  linear; dur: 4000">
                </a-box>
            </a-marker>
            <a-entity camera></a-entity>
        </a-scene>
    </body>
</html>
```

Решение

Из условия видно, что оверлей расположен на кубе, поэтому будем использовать тег `<a-box>`. Необходимо подгрузить текстурный атлас в качестве `asset` для дальнейшего его использования, и указать по `id` в свойстве `src` тега `<a-box>`.

Для работы с текстурным атласом и его анимацией требуется свойство `spritesheet-animation` внутри которого указывается:

- как именно порезать атлас, у нас будут использоваться колонки и столбцы, которых 8 и 2 соответственно (`rows: 2; columns: 8;`);
- длительность кадра 0,08 (`frameDuration: 0.08;`);
- зацикленность (`loop: true;`).

Кроме этого, нужно само свойство `animation`, в котором указываем:

- направление вращения (`property: rotation; to: 0 -360 0;`);
- зацикленность (`loop: true;`);
- изменить функцию плавности на линейную, иначе кубик будет дергаться (`easing: linear;`);
- скорость вращения (`dur: 4000`);

эти данные можно увидеть исходя из просмотра видео.

Полный код решения

```
<a-box position="0 0 0"
    material="shader: flat; src: #animate; transparent: true; "
    spritesheet-animation="rows: 2; columns: 8; frameDuration: 0.08; loop: true;"
    animation="property: rotation; to: 0 -360 0; loop: true; easing: linear; dur:
    ↪  4000">
</a-box>
```

Ответ: данная задача проверяется регулярным выражением и обязательными элементами являются выделенные части:

```
<html>
   <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
       <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aram
       ↪  e-ar.js"></script>
       <body style="margin : 0px; overflow: hidden;">
           <a-scene embedded arjs>
               <a-assets>
                   <img id="basketball" src="https://i.imgur.com/ioLOFLj.png">
                   <img id="wood"       src="https://i.imgur.com/onN1zfo.jpeg">
                   <img id="house"      src="https://i.imgur.com/YKLgAEN.jpeg">
```

```
            </a-assets>

            <a-marker preset='hiro'>
                <a-plane
                src="_?1?_"
                height="3"
                width="3"
                position="_?2?_"
                ></a-plane>

                <a-sphere
                src="_?3?_"
                scale="0.5 0.5 0.5"
                position="_?4?_"
                ></a-sphere>

                <a-box
                src="_?5?_"
                position="_?6?_"
                scale="0.5 0.5 0.5"
                ></a-box>

                <a-box
                color="_?7?_"
                position="0 0 0"
                scale="0.5 0.5 0.5"
                rotation="0 0 0"
                animation="property: _?8?_; to:360 360 360; loop: true; dur:
                ↪  10000">
                ></a-box>
            </a-marker>
            <a-entity camera></a-entity>
        </a-scene>
    </body>
</html>
```

Задача IV.1.12. Множество объектов (4 балла)

Темы: маркер, триггер, оверлей, HTML, A-Frame.

Условие

В шаблоне кода ниже, есть восемь пропусков, которые обозначены как «`_?n?_`», где n — номер пропуска.

Заполните пропуски так, чтобы код соответствовал условиям:

- при наведении на маркер `hiro` отобразилось 4 объекта: два куба, шар и картинка;
- ровно над маркером будет висеть синий куб;
- ближайший к человеку объект — это шар;
- на картинке изображен домик;
- самый левый от смотрящего куб обладает текстурой дерева;
- шар похож на баскетбольный мяч;
- картинка расположена дальше остальных;
- объект над маркером должен вращаться во всех плоскостях.

Шаблон кода

```
<html>
   <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
       <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-ar.js"></script>
       <body style="margin : 0px; overflow: hidden;">
           <a-scene embedded arjs>
               <a-assets>
                   <img id="basketball" src="https://i.imgur.com/ioLOFLj.png">
                   <img id="wood"       src="https://i.imgur.com/onN1zfo.jpeg">
                   <img id="house"      src="https://i.imgur.com/YKLgAEN.jpeg">
               </a-assets>

               <a-marker preset='hiro'>
                   <a-plane
                   src="_?1?_"
                   height="3"
                   width="3"
                   position="_?2?_"
                   ></a-plane>

                   <a-sphere
                   src="_?3?_"
                   scale="0.5 0.5 0.5"
                   position="_?4?_"
                   ></a-sphere>

                   <a-box
                   src="_?5?_"
                   position="_?6?_"
                   scale="0.5 0.5 0.5"
                   ></a-box>

                   <a-box
                   color="_?7?_"
                   position="0 0 0"
                   scale="0.5 0.5 0.5"
                   rotation="0 0 0"
                   animation="property: _?8?_; to:360 360 360; loop: true; dur: 10000">
                   ></a-box>
               </a-marker>
               <a-entity camera></a-entity>
           </a-scene>
       </body>
   </html>
```

Решение

По условию задачи «на картинке изображен домик», за вывод картинок отвечает в данном случае тег `<a-plane>` так как в единственном экземпляре, то в пропуске № 1 необходимо указать в качестве `id` ресурса `#house`. При этом «картинка расположена дальше остальных», поэтому подходят координаты $0\ 0\ -3$, которые указываем в пропуске № 2.

Далее по коду есть тег `<a-sphere>` который можно соотнести с условием «шар похож на баскетбольный мяч» и поставить в прописк № 3 `id` ресурса `#basketball`. Про его позицию сказано следующее: «самый ближайший к человеку объект — это

шар». Из предлагаемых вариантов подходит 1 0 1, который сдвинет шар ближе к человеку.

В пропуске № 6 вставляем оставшиеся координаты: −1 0 −1, что соответствует самому левому кубу, по условию который должен иметь структуру дерева, поэтому указываем в пропуске № 5 `src="#wood"`.

Последний тех `<a-box>` судя по координатам висит прямо над маркером, соответственно условию должен иметь голубой цвет, поэтому в пропуске № 7 выбираем `blue`. А в пропуске № 8 необходимо указать rotation для вращения куда по всем плоскостям.

```
<html>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-a
    ↪  r.js"></script>
    <body style="margin : 0px; overflow: hidden;">
        <a-scene embedded arjs>
            <a-assets>
                <img id="basketball" src="https://i.imgur.com/ioLOFLj.png">
                <img id="wood"       src="https://i.imgur.com/onN1zfo.jpeg">
                <img id="house"      src="https://i.imgur.com/YKLgAEN.jpeg">
            </a-assets>

            <a-marker preset='hiro'>
                <a-plane
                src="#house"
                height="3"
                width="3"
                position="0 0 -3"                >
                ></a-plane>

                <a-sphere
                src="#basketball"
                scale="0.5 0.5 0.5"
                position="1 0 1"
                ></a-sphere>

                <a-box
                src="#wood"
                position="-1 0 -1"
                scale="0.5 0.5 0.5"
                ></a-box>

                <a-box
                color="blue"
                position="0 0 0"
                scale="0.5 0.5 0.5"
                rotation="0 0 0"
                animation="property: rotation; to:360 360 360; loop: true; dur:
                ↪  10000">
                ></a-box>
            </a-marker>
            <a-entity camera></a-entity>
        </a-scene>
    </body>
</html>
```

Рис. IV.1.2. Пример результата

Ответ:

- пропуск 1: `#house`;
- пропуск 2: `0 0 -3`;
- пропуск 3: `#basketball`;
- пропуск 4: `1 0 1`;
- пропуск 5: `#wood`;
- пропуск 6: `-1 0 -1`;
- пропуск 7: `blue`;
- пропуск 8: `rotation`.

Задания расширенного уровня

Задача IV.2.1. 3D-анимация (15 баллов)

Темы: цифровой аватар, трехмерная модель, Blender, анимация.

Условие

В последнее время в нашем мире происходят активные изменения, все чаще люди учатся и работают из дома. Не обошла эта тенденция и самый волшебный праздник — новый год. Заботливые Дед Мороз и Снегурочка хотят убедиться, что все дети получат поздравления с наступающим новым годом, так что они решили разослать электронные анимации с поздравлениями и просят вас помочь.

Вам предстоит воссоздать сцену со снеговиком, приветствующим нас. В приложенных файлах вы найдете файл `.blend`, в котором и необходимо выполнить работу. Настройки камеры и освещения оставьте прежними. Ваш анимированный объект должен оставаться в центре сцены, самая нижняя точка тела объекта должна находиться в точке $(0, 0, 0)$ или приближена к этому.

Для облегчения работы вам предоставлены определенные параметры, а также

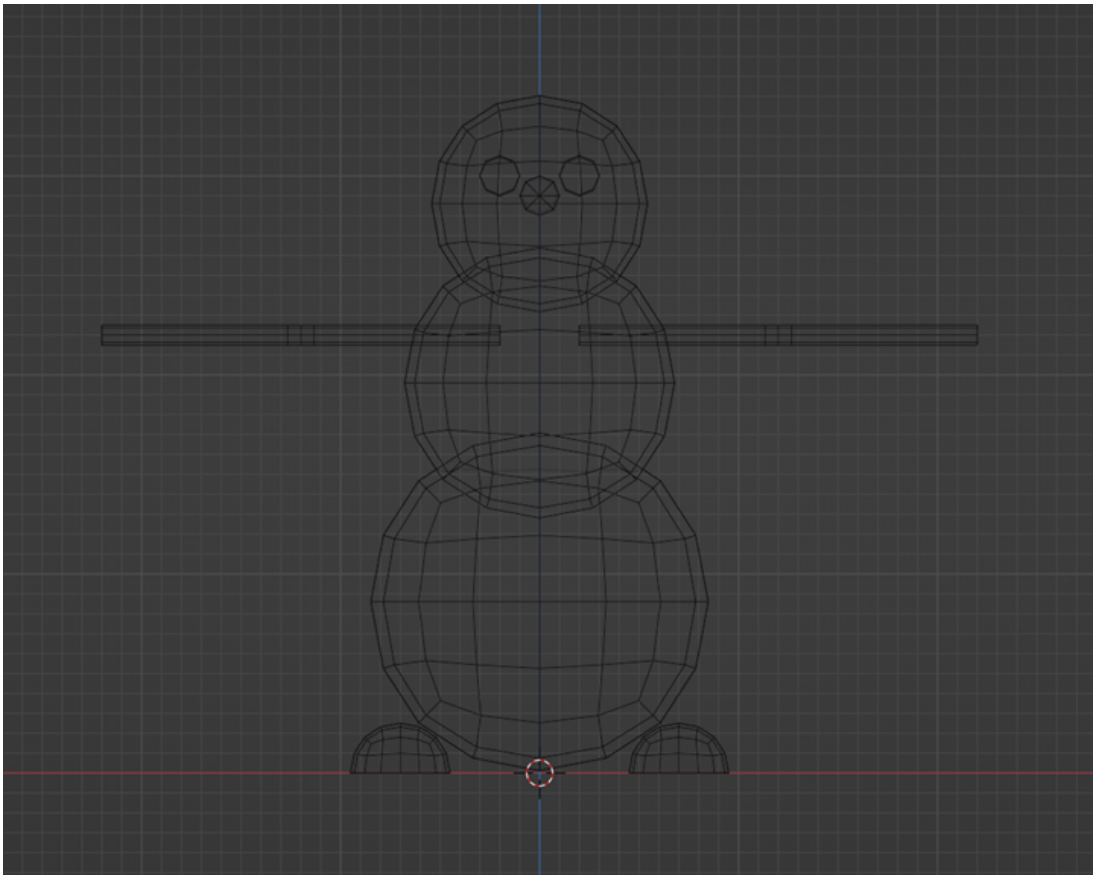
имеется сопроводительное видео, в котором продемонстрирована сама анимация и настройки объектов.

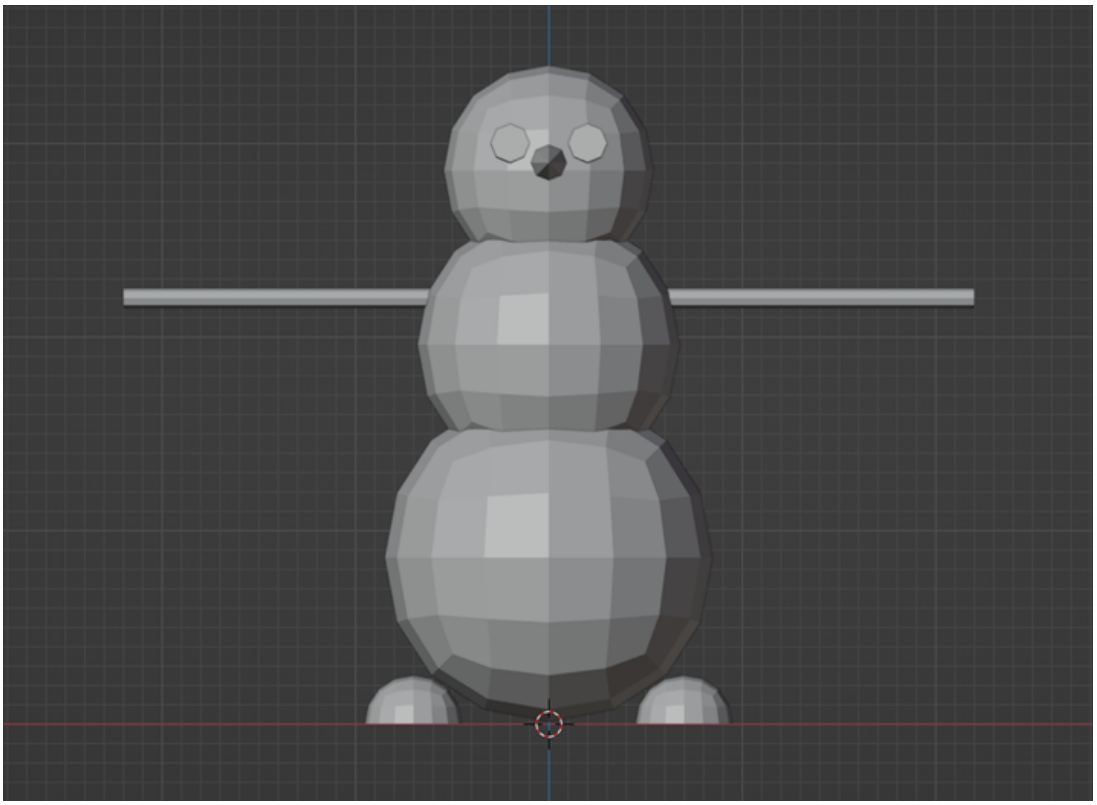
Все материалы, нужные для выполнения задания, уже содержатся в файле, однако при необходимости вы можете создать свои материалы со следующими цветами:

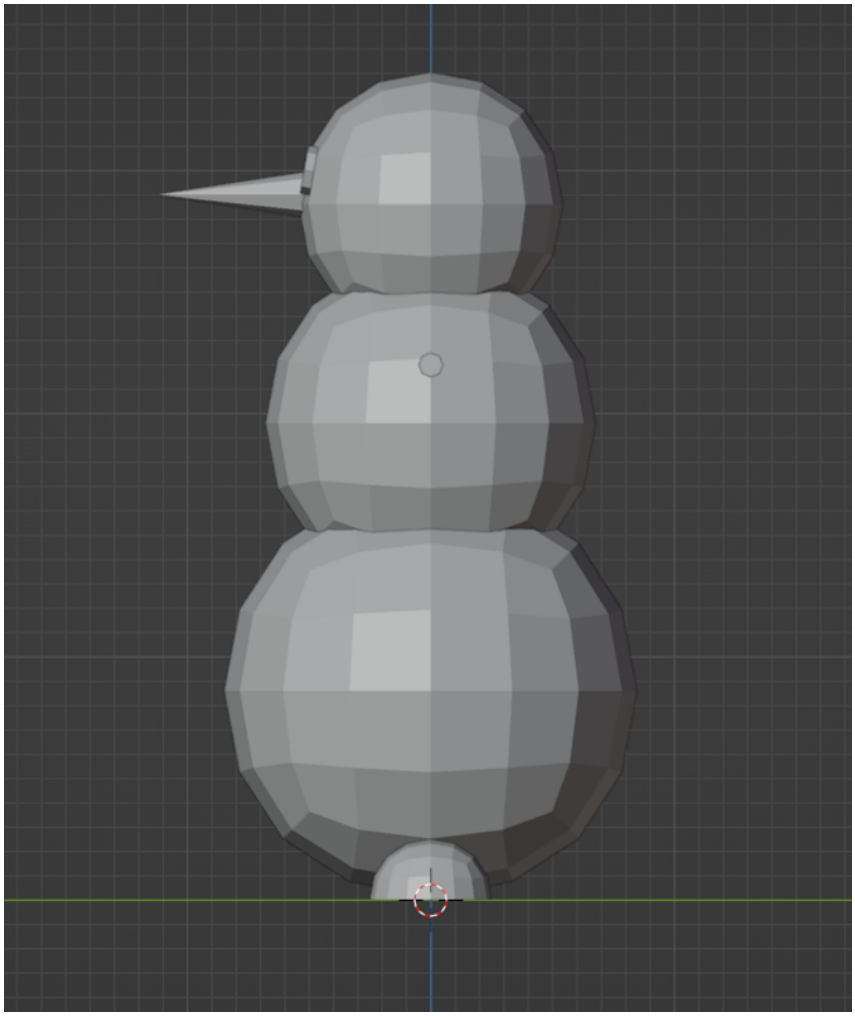
- основа снеговика – `#E7E7E7`;
- глаза – `#000000`;
- руки – `#743411`;
- морковь – `#E75330`.

Все остальные настройки материалов оставьте прежними.

Основа снеговика состоит из 3-х скругленных кубов (далее сфера), нижняя сфера имеет размерность 1,7 по каждой из осей, верхняя имеет размерность 1,09 по каждой из осей. Глаза и руки созданы из цилиндров с определенным количеством ребер. Морковь является конусом с определенным количеством ребер. Ноги — полусферы из скругленных кубов.



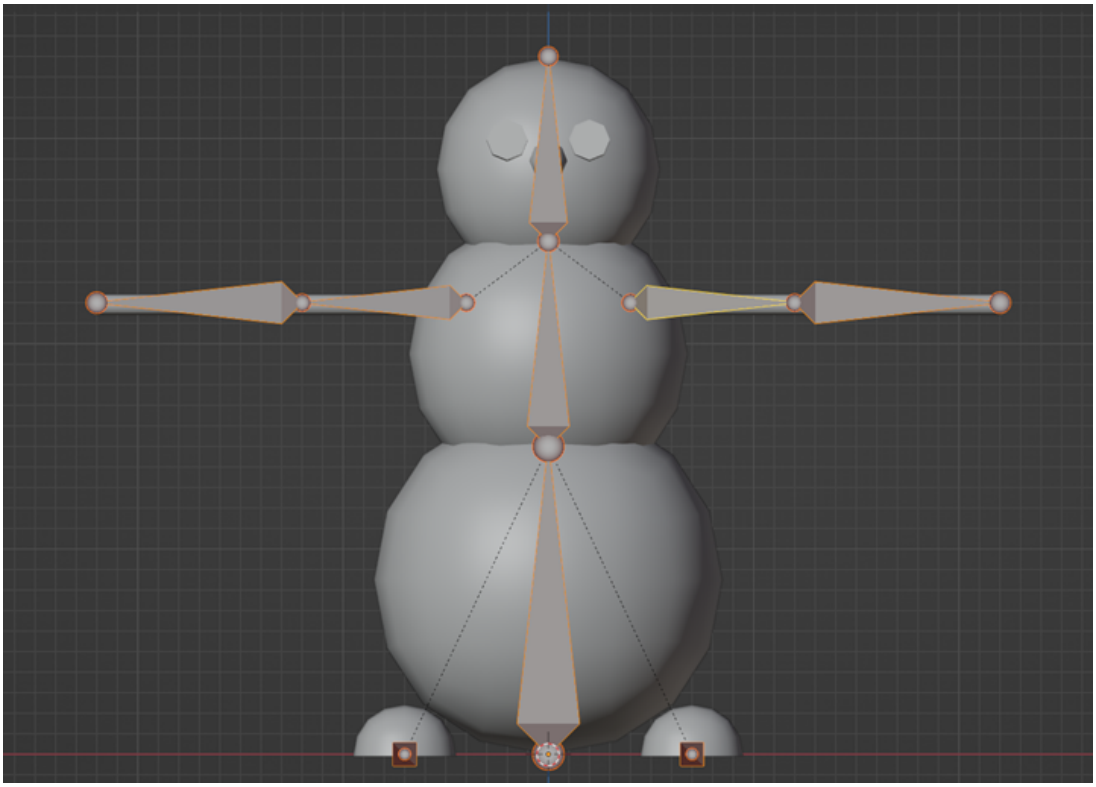




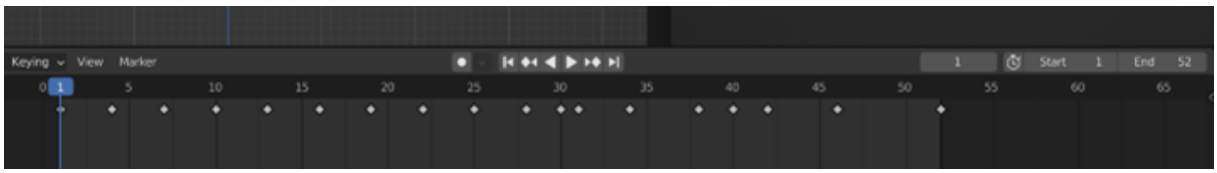
Ко всем частям применен мягкий шейдинг (`smooth shading`) и автоматическое

сглаживание с углом 47 градусов.

Для анимации объекта добавлен и присоединен риг со следующим расположением костей:



Всего в анимации 52 кадра, из них 1, 4, 7, 10, 13, 16, 19, 22, 25, 28, 30, 31, 34, 38, 40, 42, 46, 52 являются ключевыми. Анимация начинается с позиции шага (`contact`).



Так же, дано видео, в котором можно подробнее рассмотреть анимацию со всех сторон: https://disk.yandex.ru/i/-4qUsNQ77oarfg.

Для отклика на задание вам необходимо создать рендер вашей анимированной модели, настройки рендера уже содержатся в приложенном файле: https://disk.yandex.ru/d/LP3hMOXbef-H7Q.

Создайте рендер анимации, состоящий из 52 изображений и результат приложите в отклик в виде zip-архива (в архиве должны быть только изображения, никаких папок).

Анимацию необходимо повторить как можно ближе к оригиналу для получения зачета по заданию.

Способ оценки работы

Для проверки результата использовался следующий код на языке Python 3.

```python
from skimage.metrics import structural_similarity as compare_ssim
import cv2

if __name__ == "__main__":
    imageA = cv2.imread("путь к оригинальному изображению")
    imageB = cv2.imread("путь к изображению из отклика")

    grayA = cv2.cvtColor(imageA, cv2.COLOR_BGR2GRAY)
    grayB = cv2.cvtColor(imageB, cv2.COLOR_BGR2GRAY)

    score = compare_ssim(grayA, grayB, full=True)[0] * 100
    if score > 95:
        print(True)
    else:
        print(False)
```

Где `score` — регулируемая обсуждаемая величина. Для следующего пака изображений проверка выше проходит только для файла `002.png`, исходное изображение — `001.png`, остальные два изображения проверку по схожести не проходят. При величине `score` > 94 проверку проходят все 3 изображения.

Ответ: эталонное решение задачи, по которому происходит проверка доступно по ссылке: https://disk.yandex.ru/d/Tjh46yWv7yJ1dQ и `blend`-файл из которого сделан рендеринг доступен по ссылке: https://disk.yandex.ru/d/LP3hMOXbef-H7Q.

Задача IV.2.2. Волшебный куб (15 баллов)

Темы: маркер, триггер, оверлей, HTML, AR.js, A-Frame.

Условие

Разработайте систему, которая позволит реализовать «эффект волшебного куба», когда объекты дополненной реальности накладывается на куб, покрытый шестью маркерами.

Ниже представлен шаблон кода, допишите фрагмент кода так, чтобы:

- маркерами были буквы ABCDFG: https://disk.yandex.ru/d/AFfywBQlt1fjWA;
- на маркере A должна быть подгружена картинка с названием `image1.jpg` и `id = "img1"`;
- маркер B — аудиофайл с названием `audio.mp3` и `id = "audio1"`;
- маркер C — текст «Привет, мир!»;
- маркер D — картинка с названием `image2.jpg` и `id = "img2"`;
- маркер F — видео с названием `move1.mp4` и `id = "move1"`;
- маркер G — 3D фигура с названием `model3d.gltf` и `id = "model1"`.

```html
<html>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-a
    ↪  r.js"></script>
    <body style="margin : 0px; overflow: hidden;">
        <a-scene arjs embedded>
```

```
            <!--ваш фрагмент кода-->
            <a-entity camera></a-entity>
        </a-scene>
    </body>
</html>
```

В ответ напишите фрагмент кода.

Примечание: отступы и переносы строк могут быть в любом месте.

Способ оценки работы

Для проверки результата использовался следующий код на языке Python 3.

```
import re
def check(reply):
    r1="a-asset[^&]+img[^&]+(id[^&]+img1|src[^&]+image1\.jpg)[^&]+(?!\1)(id[^&]+img1|
    ↪  src[^&]+image1\.jpg)[^&]+/a-asset[^&]+a-marker[^&]+preset[^&]+letterA\.patt[^
    ↪  &]+[a-plane|a-entity]+[^&]+src[^&]+img1[^&]+/[a-plane|a-entity]+[^&]+/a-marke
    ↪  r"
    r2 = "a-asset[^&]+audio[^&]+(id[^&]+audio1|src[^&]+audio\.mp3)[^&]+(?!\1)(id[^&]+
    ↪  audio1|src[^&]+audio\.mp3)[^&]+/a-asset[^&]+a-marker[^&]+preset[^&]+letterB\.
    ↪  patt[^&]+a-sound[^&]+src[^&]+audio1[^&]+/a-marker"
    r3 = "a-marker[^&]+preset[^&]+letterC\.patt[^&]+text[^&]+value[^&]+Привет,
    ↪  мир![^&]+/a-marker"
    r4 = "a-asset[^&]+img[^&]+(id[^&]+img2|src[^&]+image2\.jpg)[^&]+(?!\1)(id[^&]+img
    ↪  2|src[^&]+image2\.jpg)[^&]+/a-asset[^&]+a-marker[^&]+preset[^&]+letterD\.patt
    ↪  [^&]+[a-plane|a-entity]+[^&]+src[^&]+img2[^&]+/[a-plane|a-entity]+[^&]+/a-mar
    ↪  ker"
    r5 = "a-asset[^&]+video[^&]+(id[^&]+move1|src[^&]+move1\.mp4)[^&]+(?!\1)(id[^&]+m
    ↪  ove1|src[^&]+move1\.mp4)[^&]+/a-asset[^&]+a-marker[^&]+preset[^&]+letterF\.pa
    ↪  tt[^&]+a-video[^&]+src[^&]+move1[^&]+/a-video[^&]+/a-marker"
    r6 = "a-asset[^&]+a-asset-item[^&]+(id[^&]+model1|src[^&]+model3d\.gltf)[^&]+(?!\
    ↪  1)(id[^&]+model1|src[^&]+model3d\.gltf)[^&]+/a-asset-item[^&]+/a-asset[^&]+a-
    ↪  marker[^&]+preset[^&]+letterG\.patt[^&]+(a-entity[^&]+gltf-model[^&]+|a-gltf-
    ↪  model[^&]+src[^&]+)+model1[^&]+/a-marker"

    if (re.search(r1, reply)!=None and re.search(r2, reply)!=None and re.search(r3,
    ↪  reply)!=None and re.search(r4, reply)!=None and re.search(r5, reply)!=None
    ↪  and re.search(r6, reply)!=None):
        return True
    return False
```

Ответ:

```
<html>
    <script src="https://aframe.io/releases/1.3.0/aframe.min.js"></script>
    <script src="https://raw.githack.com/AR-js-org/AR.js/master/aframe/build/aframe-a
    ↪  r.js"></script>

    <body style="margin : 0px; overflow: hidden;">

        <a-scene arjs embedded>

            <a-asset>
                <img  src="image1.jpg" id="img1">
                <img  src="image2.jpg" id="img2">
```

```
            <audio  id="audio1" src ="audio.mp3" crossorigin="anonymous"></audio>
            <video id="move1" src="move1.mp4"  loop="true"><video>

            <a-asset-item id="model1" src="https://cdn.aframe.io/test-models/mode
            ↪  ls/glTF-2.0/virtualcity/model3d.gltf"></a-asset-item
            ↪  >

        </a-asset>

        <a-marker preset="./src/markers/patterns/letterA.patt">
            <a-plane src="img1"></a-plane>
        </a-marker>

        <a-marker preset="./src/markers/patterns/letterB.patt">
            <a-plane src="#img" height="1" width ="1" rotation="-90 0 0">
                <a-sound src="#audio1"></a-entity>
            </a-plane>
        </a-marker>

        <a-marker preset="./src/markers/patterns/letterC.patt">
            <a-entity text="value: Привет, мир!;"></a-entity>
        </a-marker>

        <a-marker preset="./src/markers/patterns/letterD.patt">
            <a-plane src="img2"></a-plane>
        </a-marker>

        <a-marker preset="./src/markers/patterns/letterF.patt">
            <a-video src="#move1" rotation="-90 0 0"></a-video>
        </a-marker>

        <a-marker preset="./src/markers/patterns/letterG.patt">
           <a-entity gltf-model="#model1" modify-materials></a-entity>

        </a-marker>
        <a-entity camera></a-entity>
    </a-scene>
</body>

</html>
```

Задача IV.2.3. Сумма технологий (22 баллов)

Темы: webAR, AR опыт, HTML, CSS, A-Frame.

Условие

Создайте сайт, который будет объединять все задачи Базового и Расширенного уровней. А также будет содержать видеопрезентацию команды, оформленную в виде AR-проекта.

Сайт должен состоять из нескольких обязательных компонентов.

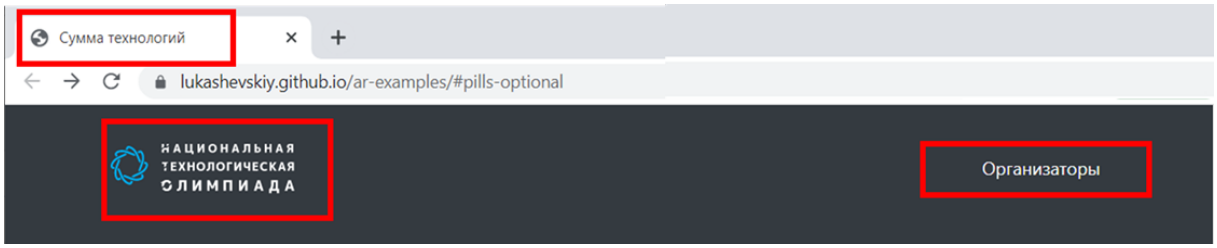
Шапка сайта Задание: Вставьте логотип команды и ее название. Подпишите название сайта «Сумма технологий»
О команде Раскрывающийся блок или ссылка на другую страницу Задание: используйте маркер в виде куба из прошлого задания для видеопрезентации команды. Структура куба следующая: верхняя грань — название команды; нижняя — видео о городе; боковые грани — видео о членах команды.
Задачи базового уровня Раскрывающиеся блоки или ссылки на другие страницы Задание: разместите каждую задачу в виде блока, внутри которого укажите: название задачи; видеофрагмент демонстрацией эффекта; картинка вашего маркера для данного задания; ссылка на код решения; ссылка на рабочую демонстрации (открывается страничка, где можно протестировать эффекта, т. е. показать маркер и на него наложится оверлей)
Задача расширенного уровня Раскрывающийся блок или ссылка на другую страницу Задание: маркер в виде куба. на гранях которого появляются тематические «новогодние» анимированные 3D-фигуры, выполненные по образцу снеговика (задание 2 этапа)

При этом, для всех задач нужно заменить предустановленные маркеры на отрисованные вами командные маркеры.

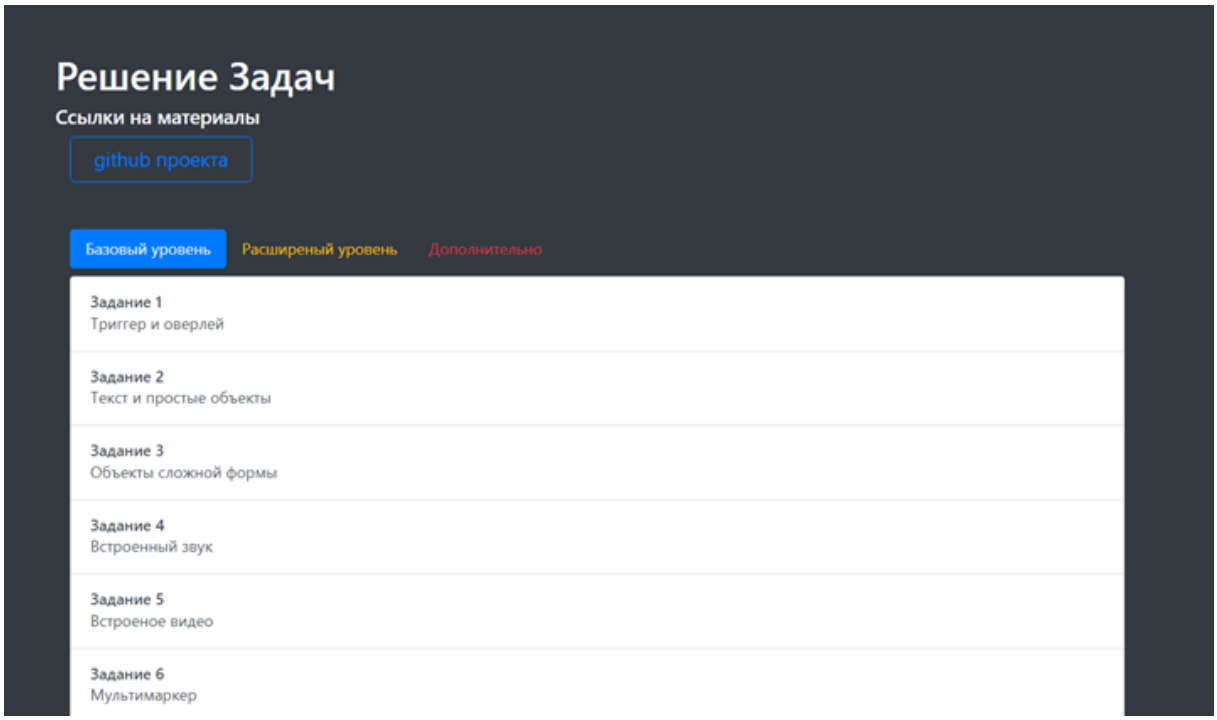
В отклик прикрепите ссылку на созданный сайт.

Критерии оценивания

- В представленном решении имеется блок «Шапка сайта», в который вставлено название и логотип команды, и название сайта.



- В решении имеется блок «О команде», где демонстрируется кубический маркер с видео и текстом о команде. Все стороны куба имеют уникальный маркер, который идентифицируется как уникальный.
- В решении имеются все задачи базового уровня с наличием следующих элементов у задач: демо-видео, код, страничка/блок с работающим решением, прикреплены маркеры, необходимые для визуализации оверлея.



- В решении имеются задача из Расширенного уровня: кубический маркер, на каждой стороне которого представлена уникальная анимированная 3D фигура.

Решение

Ссылка на решение: https://lukashevskiy.github.io/ar-examples/.

Ссылка на гит проекта: https://github.com/Lukashevskiy/ar-examples.

По ссылке представлен код главной страницы, где имеются все необходимые разделы с навигационной структурой сайта: https://github.com/Lukashevskiy/ar-examples/blob/master/index.html.

Коды отдельных задач представлены в данном документе выше.

Пример кода страницы задачи, в котором представлен элемент загружаемого контейнера, через который можно было также реализовать просмотр решения со страницы сайта.

```
<!doctype html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1,
        ↪  shrink-to-fit=no">

        <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dis
        ↪  t/css/bootstrap.min.css"
        ↪  integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9Jvo
        ↪  RxT2MZw1T"
        ↪  crossorigin="anonymous">
        <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popp
        ↪  er.min.js"
        ↪  integrity="sha384-oBqDVmMz9ATKxIep9tiCxS/Z9fNfEXiDAYTujMAeBAsjFuCZSmKbSSU
        ↪  nQlmh/jp3"
        ↪  crossorigin="anonymous"></script>
        <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/boot
        ↪  strap.min.js"
        ↪  integrity="sha384-mQ93GR66B00ZXjt0YO5KlohRA5SY2XofN4zfuZxLkoj1gXtW8ANNCe9
        ↪  d5Y3eG5eD"
        ↪  crossorigin="anonymous"></script>

        <title>Задание</title>
    </head>
    <script>
        let show;
        let codeElement;
        document.addEventListener("DOMContentLoaded", () =>{

            show = false;
            codeElement = document.querySelector("#codeText");
            var xhr= new XMLHttpRequest();
            xhr.open('GET', '../simple_object/index.html', true);
            xhr.onreadystatechange= function() {
            if (this.readyState!==4) return;
            if (this.status!==200) return;
                codeElement.innerText= this.responseText;
            };
            xhr.send();

        });
        function showCode(){
            if(show){
                show = false;
```

```
            codeElement.style.display = "block";
        }else{
            show = true;
            codeElement.style.display = "none";
        }
    }

</script>
<body class="bg-dark">
    <header class="p-3 bg-dark">
        <div class="container">
            <div class="d-flex flex-wrap align-items-center
            ↪  justify-content-md-between">
                <a href="https://ntcontest.ru/" class="d-flex align-items-center
                ↪  mb-2 mb-lg-0 text-white text-decoration-none">
                    <img class="bi me-2" width="200px" height="80px" role="img"
                    ↪  src="https://ntcontest.ru/local/templates/main/img/header
                    ↪  -logo-nto.svg"></svg>
                </a>

                <ul class="nav col-12 col-lg-auto me-lg-auto mb-2
                ↪  justify-content-center mb-md-0">
                    <li><a href="../index.html" class="nav-link px-2
                    ↪  text-light">Организаторы</a></li>
                </ul>

            </div>
        </div>

    </header>

    <div class="container">
        <h1 class="text-light">
            Задание 1
        </h1>
        <h5 class="text-light">
            Простой объект и оверлей
        </h5>

        <video class="container" class="object-fit-fill" controls>
            <source src="../videos/basic_cude_demonstration.mp4" type="video/mp4">
        </video>

        <h5 class="text-light">
            Ссылки на материалы
          </h5>

          <div class="gap-2 d-sm-flex mb-5">
            <a href="https://github.com/Lukashevskiy/ar-examples/tree/master/simp
            ↪  le_object/">
              <button type="button" class="btn btn-outline-primary btn-lg
              ↪  px-4">Материал задания</button>
            </a>
            <a href="https://github.com/Lukashevskiy/ar-examples/tree/master/simp
            ↪  le_object/src/marker/">
              <button type="button" class="btn btn-outline-secondary btn-lg
              ↪  px-4">Маркеры</button>
            </a>
              <button type="button" class="btn btn-outline-secondary btn-lg px-4"
              ↪  onclick="showCode()">Код страницы</button>
```

```
            </div>
        </div>
        <div class="container py-5">
            <code id="codeText" style="display:none" class="container bg-light
            ↪  py-4">
            </code>
        </div>
    </body>
</html>
```

Пример кода с блоком о команде и решением, где все маркеры отличимы друг от друга: https://github.com/Lukashevskiy/ar-examples/blob/master/global_cube/index.html.

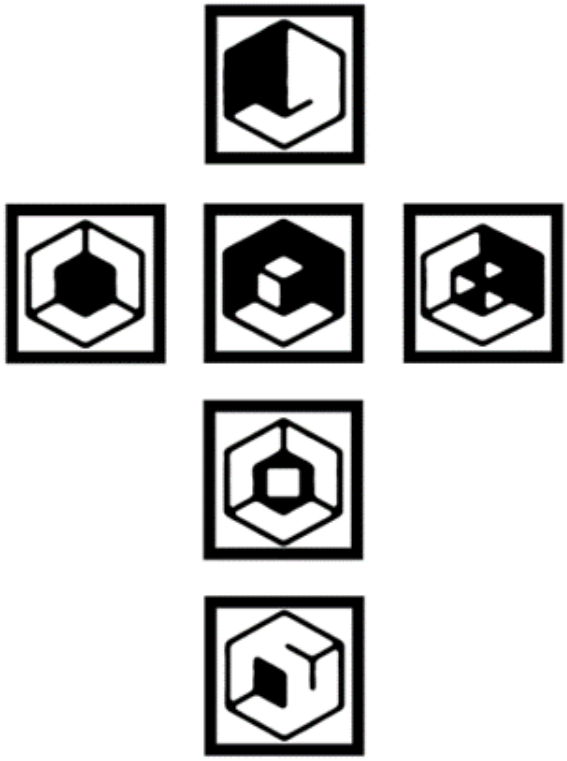


Рис. IV.2.1. Используемые маркеры для задачи «о команде»