

Комплект оценочных материалов по дисциплине «Программирование WebGL»

Задания закрытого типа

Задания закрытого типа на выбор правильного ответа

1. *Выберите правильный ответ.*

Базовым понятием векторной графики является:

- А) каркас
- Б) пиксель
- В) текстура
- Г) линия
- Д) вектор

Правильный ответ: Д

Компетенции (индикаторы): ПК-2

2. *Выберите правильный ответ.*

Что не входит в структуру векторных файлов?

- А) команды рисования
- Б) инструкции для масштабирования
- В) таблицы информации о цвете
- Г) шрифты
- Д) примитивы

Правильный ответ: А

Компетенции (индикаторы): ПК-2

3. *Выберите правильный ответ.*

Процесс определения пикселей, наилучшим образом аппроксимирующих заданный отрезок, называется:

- А) векторизация
- Б) растеризация
- В) связность
- Г) разложение в растр
- Д) развёртка

Правильный ответ: Г

Компетенции (индикаторы): ПК-2

4. *Выберите правильные ответы.*

Решение задач компьютерной графики охватывает:

- А) синтез и анализ изображений
- Б) факторный анализ
- В) моделирование и анимация изображений
- Г) обработка и визуализация данных
- Д) управление данными

Правильные ответы: А, В, Г

Компетенции (индикаторы): ПК-2

Задания закрытого типа на установление соответствия

1. Установите соответствие между этапами рендеринга изображения и их описанием. Каждому элементу левого столбца соответствует только один элемент правого столбца.

Этап	Описание
1) Рендеринг кадра	А) определение видимых частей объектов с учётом их положения и перспективы
2) Построение сцены	Б) использование алгоритмов для преобразования всех данных сцены в финальное изображение
3) Расчёт геометрии	В) определение угла обзора, положения и параметров
4) Установка источников освещения	Г) определение объектов, их свойств и пространственных взаимоотношений между ними
5) Камера	Д) настройка света, его яркости, положения и взаимодействия с объектами

Правильный ответ: 1-Б, 2-Г, 3-А, 4-Д, 5-В

Компетенции (индикаторы): ПК-2

2. Установите соответствие между матрицей и её видом в однородных координатах. Каждому элементу левого столбца соответствует только один элемент правого столбца.

1) Матрица масштабирования	А) $\begin{bmatrix} \cos \alpha & \sin \alpha & 0 \\ -\sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$
----------------------------	---

2) Матрица переноса

Б)
$$\begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3) Матрица вращения

В)
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ D_x & D_y & 1 \end{bmatrix}$$

Правильный ответ: 1-Б, 2-В, 3-А

Компетенции (индикаторы): ПК-2

3. Установите соответствие между типом проекции и её описанием. Каждому элементу левого столбца соответствует только один элемент правого столбца.

- | | |
|--|--|
| 1) центральная
(перспективная) | А) центр проекции удалён на бесконечность |
| 2) параллельная
(аксонометрическая) | Б) нормаль к проекционной плоскости образует три различных угла с осями |
| 3) ортографическая | В) проекторы не ортогональны к проекционной плоскости |
| 4) косоугольная | Г) направление проецирования является нормалью к проекционной плоскости |
| 5) триметрическая | Д) центр проекции находится на конечном расстоянии от проекционной плоскости |

Правильный ответ: 1-Д, 2-А, 3-Г, 4-В, 5-Б

Компетенции (индикаторы): ПК-2

Задания закрытого типа на установление правильной последовательности

1. Установите правильную последовательность преобразований для визуализации трёхмерного объекта на экране компьютера.

А) спроецировать полученные координаты на проекционную плоскость в системе координат наблюдателя

Б) преобразовать координаты объекта, заданные в собственной системе координат, в мировые координаты

В) преобразовать координаты объекта, заданные в мировой системе координат, в систему координат наблюдателя

Правильный ответ: Б, В, А

Компетенции (индикаторы): ПК-2

2. Установите правильную последовательность удаления невидимых граней в алгоритме Робертса.

А) рендерятся только видимые грани, создавая правильное представление объекта с точки зрения наблюдателя

Б) для каждой грани объекта рассчитывается нормаль (вектор, перпендикулярный плоскости грани), помогающая определить ориентацию грани относительно наблюдателя

В) невидимые грани исключаются из дальнейшей обработки

Г) задаётся геометрическая модель объекта, включая координаты вершин и рёбра

Д) определяется угол между нормалью грани и направлением на наблюдателя (или камеры): если нормаль направлена от наблюдателя (скалярное произведение меньше или равно нулю), грань считается невидимой

Е) видимые грани упорядочиваются для корректного отображения с учётом перекрытий (например, сортировка по глубине)

Правильный ответ: Г, Б, Д, В, Е, А

Компетенции (индикаторы): ПК-2

3. Упорядочить этапы пересчёта координат объекта из трёхмерного пространства в диметрическую проекцию с использованием матрицы преобразований.

А) преобразование координат объекта: трёхмерные координаты (x, y, z) объекта представлены в виде столбца вектора, а для получения проекционных координат этот вектор умножается на матрицу диметрической проекции

Б) для корректного отображения объекта производится упорядочивание элементов сцены на основе глубины z до начала прорисовки

В) новые координаты (x', y') определяются, основываясь на преобразованных значениях. z обычно игнорируется в итоговой проекции

Г) определение матрицы диметрической проекции, учитывающей угол наклона осей относительно наблюдателя

Д) на основе пересчитанных координат и масштаба визуализируется двумерная проекция объекта на экране

Правильный ответ: Г, А, В, Б, Д

Компетенции (индикаторы): ПК-2

Задания открытого типа

Задания открытого типа на дополнение

1. *Напишите пропущенное слово.*

_____ – преобразование точек пространства размерности n в точки пространства размерности меньшей, чем n , или на подпространство исходного пространства.

Правильный ответ: проекция

Компетенции (индикаторы): ПК-2

2. *Напишите пропущенное слово.*

Система координат с осями, расположенными взаимно перпендикулярно, называется _____.

Правильный ответ: ортогональной

Компетенции (индикаторы): ПК-2

3. *Напишите пропущенное словосочетание.*

_____ – это система координат, связанная с положением и ориентацией наблюдателя (камеры) в пространстве.

Правильный ответ: система координат наблюдателя

Компетенции (индикаторы): ПК-2

4. *Напишите пропущенное словосочетание.*

_____ – точка пересечения центральных проекций любой совокупности параллельных прямых, которые не параллельны проекционной плоскости.

Правильный ответ: точка схода

Компетенции (индикаторы): ПК-2

5. *Напишите пропущенное слово.*

_____ – массивы данных, к которым есть произвольный доступ в программе шейдера.

Правильный ответ: текстуры

Компетенции (индикаторы): ПК-2

6. *Напишите пропущенное слово.*

Основная задача _____ шейдера – вычислить цвет для каждого пикселя примитива, который в данный момент отрисовывается.

Правильный ответ: фрагментного

Компетенции (индикаторы): ПК-2

7. Напишите пропущенное слово.

Процесс перевода двумерного изображения, описанного векторным форматом, в пиксели или точки, называется _____.

Правильный ответ: растеризация

Компетенции (индикаторы): ПК-2

8. Напишите пропущенное слово.

При обработке изображений _____ определяет, какие пиксели или области, соединённые по сторонам и диагоналям, считаются соседними.

Правильный ответ: связность

Компетенции (индикаторы): ПК-2

Задания открытого типа с кратким свободным ответом

1. Закончите утверждение.

Суть алгоритма z-буфера заключается в _____.

Правильный ответ: проверке координаты z для каждого пикселя / сравнении значения глубины (координаты z) для каждого пикселя в буфере / определении видимых поверхностей в трёхмерной сцене

Компетенции (индикаторы): ПК-2

2. Закончите утверждение.

Отрезок, концы которого согласно алгоритму Коэна-Сазерленда заданы кодами 1001 и 0011, _____.

Правильный ответ: отсекается / полностью отбрасывается / отрезок гарантированно не пересекает видимую область и полностью отбрасывается

Компетенции (индикаторы): ПК-2

3. Закончите утверждение.

Метод построения кривых Безье основан на _____.

Правильный ответ: рекурсивном повторении разбиения отрезков / многократном делении отрезков на основе контрольных точек / поэтапном делении контрольных отрезков до построения кривой / повторном разбиении отрезков до достижения точности кривой

Компетенции (индикаторы): ПК-2

4. Закончите утверждение.

В случае, когда длина отрезка меньше пикселя, отображение в одном пикселе, игнорирование отрезка, сглаживание и изменение цвета пикселя характерны для алгоритма _____.

Правильный ответ: средней точки / Брезенхейма / Брезенхема

Компетенции (индикаторы): ПК-2

Задания открытого типа с развернутым ответом

1. Что произойдёт с отрезком согласно алгоритма Коэна-Сазерленда, концы которого заданы кодом 0101 и 1000? Привести расширенное поэтапное решение с пояснением работы алгоритма.

Время выполнения – 25 мин.

Ожидаемый результат:

Алгоритм Коэна-Сазерленда используется для отсеечения отрезков, которые не попадают в видимую область (окно) на плоскости. Этот алгоритм работает с прямоугольной областью, которая делит плоскость на 9 регионов, каждому из которых присваивается 4-битный код. Код определяет положение точки относительно границ окна. Каждый бит в 4-битном коде соответствует одной из границ окна:

- Бит 1 (верхний): Точка выше верхней границы окна.
- Бит 2 (нижний): Точка ниже нижней границы окна.
- Бит 3 (правый): Точка правее правой границы окна.
- Бит 4 (левый): Точка левее левой границы окна.
- Если точка находится внутри окна, её код будет 0000.

Проанализируем данные коды концов предлагаемого отрезка. 0101 - точка находится ниже нижней границы (бит 2) и левее левой границы (бит 4). 1000 - Точка находится выше верхней границы (бит 1).

Алгоритм работает следующим образом:

- проверка тривиальной видимости: Если оба кода равны 0000, отрезок полностью видим. В нашем случае это не так.
- проверка тривиальной невидимости: Если логическое И (побитовое AND) кодов концов отрезка не равно 0000, отрезок полностью невидим. Вычисляем: $0101 \text{ AND } 1000 = 00000101 \text{ AND } 1000 = 0000$. Результат равен 0000, поэтому отрезок может быть частично видимым.
- определение пересечений: Если отрезок не тривиально видим и не тривиально невидим, алгоритм вычисляет точки пересечения отрезка с границами окна. В нашем случае один конец отрезка находится выше окна (код 1000), а другой — ниже и левее окна (код 0101), следовательно, отрезок пересекает верхнюю и левую границы окна.
- отсеечение: алгоритм вычисляет точку пересечения отрезка с верхней границей окна и заменяет конец отрезка с кодом 1000 на эту точку. Затем вычисляет точку пересечения с левой границей окна и заменяет конец

отрезка с кодом 0101 на эту точку. После этого отрезок будет полностью видимым внутри окна.

Ответ: отрезок с кодами концов 0101 и 1000 пересекает окно. Алгоритм Коэна-Сазерленда вычисляет точки пересечения с границами окна и отсекает невидимые части отрезка. В результате отрезок будет видимым внутри окна.

Компетенции (индикаторы): УК-1

2. В представленном программном коде определить, что выполняется и является результатом его работы. Привести комментарии к каждой выполняемой строке кода.

Время выполнения – 30 мин.

```
//
function initShaders()
{
    var fragmentShader = getShader(gl.FRAGMENT_SHADER, 'shader-fs');
    var vertexShader = getShader(gl.VERTEX_SHADER, 'shader-vs');
    shaderProgram = gl.createProgram();
    gl.attachShader(shaderProgram, vertexShader);
    gl.attachShader(shaderProgram, fragmentShader);
    gl.linkProgram(shaderProgram);
    if (!gl.getProgramParameter(shaderProgram, gl.LINK_STATUS))
    {
        alert("Не удалось установить шейдеры");
    }
    gl.useProgram(shaderProgram);
    shaderProgram.vertexPositionAttribute =
gl.getAttribLocation(shaderProgram, "aVertexPosition");
    gl.enableVertexAttribArray(shaderProgram.vertexPositionAttribute);
}
//
function getShader(type,id)
{
    var source = document.getElementById(id).innerHTML;
    var shader = gl.createShader(type);
    gl.shaderSource(shader, source);
    gl.compileShader(shader);
    if (!gl.getShaderParameter(shader, gl.COMPILE_STATUS))
    {
```

```

        alert("Ошибка компиляции шейдера: " +
gl.getShaderInfoLog(shader));
        gl.deleteShader(shader);
        return null;
    }
    return shader;
}

```

Критерии оценивания – полное содержательное соответствие приведённому ниже пояснению.

Данный фрагмент программного кода отвечает за компиляцию и проверку шейдеров; создание программы шейдеров, их подключение и компоновку; установку атрибутов для передачи данных в шейдеры.

// Функция инициализации шейдера

```

function initShaders()
{
    var fragmentShader = getShader(gl.FRAGMENT_SHADER, 'shader-fs'); //
загрузка и компиляция фрагментного шейдера, который отвечает за цвет и
отображение пикселей
    var vertexShader = getShader(gl.VERTEX_SHADER, 'shader-vs'); //
загрузка и компиляция вершинного шейдера, который отвечает за обработку
геометрии вершин
    shaderProgram = gl.createProgram(); // создание объекта программы
шейдеров, который будет содержать оба шейдера (вершинный и
фрагментный)
    gl.attachShader(shaderProgram, vertexShader); // подключение
вершинного шейдера к программе
    gl.attachShader(shaderProgram, fragmentShader); // подключение
фрагментного шейдера к программе
    gl.linkProgram(shaderProgram); // компоновка (объединение) программы
шейдеров в единую программу для дальнейшего использования
    if (!gl.getProgramParameter(shaderProgram, gl.LINK_STATUS))
    {
        alert("Не удалось установить шейдеры");
    } // проверка успешности компоновки программы: если программа
не скомпонована корректно, выводится сообщение об ошибке
    gl.useProgram(shaderProgram); // установка созданной программы
шейдеров в качестве активной программы, которая будет использоваться для
рендеринга
    shaderProgram.vertexPositionAttribute =
gl.getAttribLocation(shaderProgram, "aVertexPosition"); // получение

```

местоположения атрибута `aVertexPosition` из программы шейдеров, который будет использован для передачи координат вершин

```
gl.enableVertexAttribArray(shaderProgram.vertexPositionAttribute);    //  
включение полученного атрибута для использования в рендеринге  
}  
// Функция создания шейдера  
function getShader(type,id)  
{  
    var source = document.getElementById(id).innerHTML; // загрузка  
исходного кода шейдера из элемента HTML с указанным id  
    var shader = gl.createShader(type); // создание нового объекта шейдера  
заданного типа  
    gl.shaderSource(shader, source); // привязка исходного кода шейдера к  
созданному объекту шейдера  
    gl.compileShader(shader); // этап компиляции шейдера, на котором  
исходный код преобразуется в исполняемый код  
    if (!gl.getShaderParameter(shader, gl.COMPILE_STATUS)) // проверка  
успешности компиляции  
    {  
        alert("Ошибка компиляции шейдера: " +  
gl.getShaderInfoLog(shader)); // если произошла ошибка, выводится  
сообщение с деталями ошибки  
        gl.deleteShader(shader); // и шейдер удаляется  
        return null;  
    }  
    return shader; // возврат скомпилированного шейдера  
}
```

Компетенции (индикаторы): УК-1

Экспертное заключение

Представленный комплект оценочных материалов по дисциплине «Программирование WebGL» соответствует требованиям ФГОС ВО.

Предлагаемые оценочные материалы адекватны целям и задачам реализации основной профессиональной образовательной программы по направлению подготовки 09.03.03 Прикладная информатика.

Виды оценочных средств, включённые в представленный фонд, отвечают основным принципам формирования ФОС.

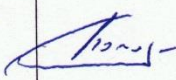
Разработанные и представленные для экспертизы оценочные материалы рекомендуются к использованию в процессе подготовки обучающихся по указанному направлению.

Председатель учебно-методической комиссии
института компьютерных систем
и информационных технологий



Ветрова Н. Н.

Лист изменений и дополнений

№ п/п	Виды дополнений и изменений	Дата и номер протокола заседания кафедры (кафедр), на котором были рассмотрены и одобрены изменения и дополнения	Подпись (с расшифровкой) заведующего кафедрой (заведующих кафедрами)
1.	Дополнен комплект оценочных материалов	протокол заседания кафедры компьютерных систем и сетей № <u>8</u> от <u>10.03.2025</u>	 С.В. Попов