Таблица функций и методов в порядке их появления в коде (task_5)

№	Функция/Метод	Назначение и принцип работы	Ссылка на лекцию
1	WinMain	Точка входа в приложение. Инициализирует окно, DirectX, ресурсы и запускает главный цикл.	Лекция 2, стр. 2–4
2	InitWindow	Создаёт и регистрирует окно с заданными размерами и стилем.	Лекция 2, стр. 5–6
3	InitDirectX	Инициализирует DirectX: создаёт устройство, контекст, своп-цепь, растеризатор, буфер глубины D32_FLOAT, состояния глубины и blend states.	Лекция 2, стр. 7–15; Лекция 8, стр. 5–20
4	D3D11CreateDeviceAndSwapChain	Создаёт логическое устройство GPU (ID3D11Device) и своп-цепь для вывода в окно.	Лекция 2, стр. 13–15
5	SetupBackBuffer	Создаёт render target view для back buffer’а своп-цепи и буфер глубины D32_FLOAT.	Лекция 2, стр. 28–29; Лекция 8, стр. 5–6
6	InitCube	Создаёт геометрию куба (вершинный и индексный буферы), компилирует шейдеры, создаёт input layout.	Лекция 3, стр. 5–12; Лекция 6, стр. 11–13
7	D3DCompile	Компилирует шейдер из исходного кода в байт-код DXBC.	Лекция 3, стр. 17–22
8	CreateInputLayout	Описывает структуру вершинного буфера для передачи в вершинный шейдер.	Лекция 3, стр. 26–28
9	InitBuffers	Создаёт константные буферы для матриц модели (m) и вида, а также для второго куба.	Лекция 5, стр. 6–8
10	LoadTexture	Загружает DDS-текстуру, создаёт ресурс текстуры, shader resource view и сэмплер.	Лекция 6, стр. 14–26
11	LoadDDS	Читает DDS-файл, извлекает заголовок, формат, данные и мип-уровни.	Лекция 6, стр. 20–24
12	InitSkybox	Создаёт сферу для skybox, загружает cubemap-текстуру, компилирует шейдеры с reversed depth.	Лекция 6, стр. 34–42; Лекция 8, стр. 31–32
13	CreateSphere	Генерирует вершины и индексы для сферы методом параметризации.	Лекция 6, стр. 34
14	InitTransparentObjects	Инициализирует прозрачные прямоугольники: геометрию, шейдеры, константные буферы, bounding boxes.	Лекция 8, стр. 15–30
15	WndProc	Обрабатывает сообщения окна: изменение размера, вращение камеры, колесо мыши.	Лекция 2, стр. 2–4
16	ResizeSwapChain	Изменяет размер back buffer’а и буфера глубины при изменении окна.	Лекция 2, стр. 28
17	UpdateCamera	Обновляет матрицу вида и проекции	Лекция 5, стр. 21–

№	Функция/Метод	Назначение и принцип работы	Ссылка на лекцию
		(reversed depth), записывает в буфер.	24; Лекция 8, стр. 12–14
18	Render	Основной цикл рендеринга: очистка, настройка конвейера, отрисовка непрозрачных объектов, skybox, прозрачных объектов.	Лекция 3, стр. 52–54; Лекция 6, стр. 28; Лекция 8, стр. 33
19	RenderTransparentObjects	Рендерит прозрачные объекты с сортировкой от дальнего к ближнему, использованием blending и без записи глубины.	Лекция 8, стр. 25–30
20	IASetVertexBuffers / IASetIndexBuffer	Настраивают входной ассемблер: привязывают вершинный и индексный буферы.	Лекция 3, стр. 32
21	VSSetShader / PSSetShader	Устанавливают вершинный и пиксельный шейдеры в конвейер.	Лекция 3, стр. 23–24
22	PSSetShaderResources / PSSetSamplers	Передают текстуры и сэмплеры в пиксельный шейдер.	Лекция 6, стр. 27–28
23	VSSetConstantBuffers	Привязывают константные буферы к вершинному шейдеру.	Лекция 5, стр. 6
24	OMSetBlendState	Устанавливает состояние смешивания для прозрачных объектов.	Лекция 8, стр. 15–20
25	OMSetDepthStencilState	Устанавливает состояние глубины (reversed depth) для разных типов объектов.	Лекция 8, стр. 12–14, 27
26	DrawIndexed	Запускает отрисовку геометрии по индексам.	Лекция 3, стр. 54
27	UpdateSubresource	Копирует данные из CPU в GPU-буфер (например, матрицу модели).	Лекция 5, стр. 9–10
28	Map / Unmap	Отображают GPU-буфер в CPU-память для записи (динамический буфер сцены).	Лекция 5, стр. 43–45
29	Cleanup	Освобождает все созданные DirectX-ресурсы.	—
30	CreateSphere (вспом.)	Генерация геометрии сферы для skybox.	Лекция 6, стр. 34
31	GetBytesPerBlock / DivUp	Вспомогательные функции для работы с DXT-сжатыми текстурами.	Лекция 6, стр. 17–19

Ключевые моменты для защиты (task_5):

1. **Reversed Depth и D32_FLOAT** – борьба с z-fighting, повышенная точность на дальних расстояниях.
2. **Прозрачные объекты и Blending** – настройка blend state для альфа-смешивания, порядок отрисовки.
3. **Сортировка прозрачных объектов** – необходимость сортировки от дальнего к ближнему для корректного смешивания.
4. **Разделение состояний глубины** – разные состояния для непрозрачных объектов, skybox и прозрачных объектов.
5. **Порядок рендеринга** – непрозрачные объекты → skybox → прозрачные объекты (с сортировкой).

Сравнение кодов: что нового добавилось в task_5 (по сравнению с task_4)

1. Буфер глубины D32_FLOAT и Reversed Depth

Компонент	task_4 (старый)	task_5 (новый)	Что изменилось
Формат глубины	DXGI_FORMAT_D24_UNORM_S8_UINT	DXGI_FORMAT_D32_FLOAT	Повышенная точность, борьба с z-fighting
Depth Test	D3D11_COMPARISON_LESS	D3D11_COMPARISON_GREATER (reversed depth)	Обратный порядок сравнения глубины
Очистка глубины	Значение 1.0	Значение 0.0	Для reversed depth очистка в 0
Матрица проекции	Обычная перспективная проекция	Reversed depth проекция (near и far swapped)	Более равномерное распределение точности

2. Прозрачные объекты (Transparent Objects)

Компонент	task_4 (старый)	task_5 (новый)	Что изменилось
Геометрия прозрачных объектов	Нет	Добавлены два прямоугольника: m_pRectVertexBuffer, m_pRectIndexBuffer	Поддержка прозрачности
Шейдеры прозрачных объектов	Нет	Добавлены: m_pRectVertexShader, m_pRectPixelShader, m_pRectInputLayout	Отдельные шейдеры для прозрачных объектов
Blend State	Нет	Добавлены: m_pTransBlendState (для смешивания), m_pOpaqueBlendState (выключено)	Система смешивания цветов
Состояние глубины для прозрачных	Нет	m_pTransDepthState (без записи глубины)	Прозрачные объекты не влияют на Z-буфер
Константные буферы	Нет	m_pRectGeomBuffer1, m_pRectGeomBuffer2 (цвет и матрица)	Управление цветом и позицией

3. Улучшения Skybox для Reversed Depth

Компонент	task_4 (старый)	task_5 (новый)	Что изменилось
Depth State Skybox	D3D11_COMPARISON_LESS_EQUAL	D3D11_COMPARISON_GREATER_EQUAL	Адаптировано под reversed

Компонент	task_4 (старый)	task_5 (новый)	Что изменилось
			depth
Вершинный шейдер	Обычное преобразование	output.pos.z = 0.0 (принудительно)	Skybox всегда на максимальной глубине

4. Второй кубик и сортировка прозрачных объектов

Компонент	task_4 (старый)	task_5 (новый)	Что изменилось
Количество кубов	Один куб	Два куба (m_pGeomBuffer, m_pGeomBuffer2)	Усложнение сцены
Сортировка прозрачных	Нет	Сортировка по расстоянию от камеры	Правильный порядок смешивания
Bounding Boxes	Нет	m_boundingRects[2] для расчёта расстояний	Вычисление дистанции для сортировки

5. Новые функции

Функция	task_4 (старый)	task_5 (новый)	Назначение
InitTransparentObjects	Нет	Да	Инициализация всех ресурсов для прозрачных объектов
RenderTransparentObjects	Нет	Да	Отрисовка прозрачных объектов с сортировкой
OMSetBlendState	Нет	Да	Установка состояния смешивания

6. Изменения в Render()

Порядок рендеринга	task_4 (старый)	task_5 (новый)
1. Очистка	Цвет + глубина (1.0)	Цвет + глубина (0.0) для reversed depth
2. Skybox	Рендерится первым	Рендерится после непрозрачных объектов
3. Куб	Один куб	Два куба
4. Прозрачные объекты	Нет	Рендерятся последними с сортировкой

7. Ключевые концептуальные добавления в task_5

1. **Reversed Depth** – использование буфера глубины D32_FLOAT с обратным порядком значений для повышения точности.
2. **Alpha Blending** – поддержка прозрачности через blend states и правильный порядок отрисовки.
3. **Сортировка прозрачных объектов** – необходимость рендеринга от дальнего к ближнему для корректного смешивания.
4. **Улучшенный Skybox** – адаптация под reversed depth (z = 0, GREATER_EQUAL).
5. **Расширенная сцена** – два кубика, два прозрачных прямоугольника, более сложный порядок рендеринга.

8. Порядок рендеринга в task_5

1. Очистка буферов (глубина = 0.0)
2. Рендеринг непрозрачных объектов (два куба) с reversed depth (GREATER)
3. Рендеринг Skybox с состоянием глубины GREATER_EQUAL (z = 0)
4. Рендеринг прозрачных объектов с blending и без записи глубины, отсортированных от дальнего к ближнему
5. Present

9. Зависимости от лекций:

- **Лекция 8 "Глубина и порядок отрисовки"** – объясняет все новые концепции:
  - Буфер глубины D32_FLOAT (стр. 4–6)
  - Reversed depth (стр. 11–14)
  - Z-fighting (стр. 8–10)
  - Blend states и прозрачные объекты (стр. 15–20)
  - Сортировка прозрачных объектов (стр. 25–30)
  - Skybox с reversed depth (стр. 31–32)

Итог: task_5 добавляет продвинутую систему работы с глубиной, поддержку прозрачности и правильный порядок рендеринга, что значительно улучшает визуальное качество и корректность сцены по сравнению с task_4.

Детальный конспект (task_5)

1. Инициализация приложения

Область кода	Что делает	Как работает	Теория
WinMain()	Точка входа Windows приложения	1. Инициализирует окно 2. Инициализирует DirectX 3. Загружает ресурсы (два куба, текстуры, skybox, прозрачные объекты) 4. Запускает главный цикл сообщений 5. При завершении освобождает ресурсы	Лекция 2, стр. 2–4
InitWindow()	Создает и регистрирует окно	1. Регистрирует класс окна (WNDCLASSEX) 2. Вычисляет размеры с учетом рамок (AdjustWindowRect) 3. Создает окно (CreateWindow) 4. Показывает окно (ShowWindow)	Лекция 2, стр. 5–6
WndProc()	Обрабатывает сообщения окна	Обрабатывает: - WM_SIZE: изменение размера окна → ResizeSwapChain() - WM_RBUTTONDOWN/UP: вращение камеры мышью - WM_MOUSEWHEEL: зум камеры - WM_DESTROY: завершение приложения	Лекция 2, стр. 2–4

2. Инициализация DirectX 11 с Reversed Depth

Область кода	Что делает	Как работает	Теория
InitDirectX()	Инициализирует графический конвейер	1. Создает устройство и своп-цепь (D3D11CreateDeviceAndSwapChain) 2. Создает растеризатор (CreateRasterizerState) 3. Создает состояния глубины для skybox, непрозрачных и прозрачных объектов 4. Создает blend states для прозрачных и непрозрачных объектов 5. Настраивает back buffer и буфер глубины D32_FLOAT (SetupBackBuffer)	Лекция 2, стр. 7–30; Лекция 8, стр. 5–20
D3D11CreateDeviceAndSwapChain()	Создает основные объекты DirectX	Создает: - ID3D11Device: логическое представление GPU - ID3D11DeviceContext: контекст для команд - IDXGISwapChain: цепочка буферов для вывода в окно	Лекция 2, стр. 13–15, 26–27
SetupBackBuffer()	Настраивает цели рендеринга	1. Получает back buffer из своп-цепи (GetBuffer) 2. Создает Render Target View	Лекция 2, стр. 28–30;

Область кода	Что делает	Как работает	Теория
		(CreateRenderTargetView) 3. Создает текстуру буфера глубины D32_FLOAT (CreateTexture2D) 4. Создает Depth Stencil View (CreateDepthStencilView)	Лекция 8, стр. 5–6

3. Reversed Depth: буфер D32_FLOAT и обратная глубина

Область кода	Что делает	Как работает	Теория
Формат буфера глубины	Использует D32_FLOAT вместо D24_UNORM	Формат DXGI_FORMAT_D32_FLOAT обеспечивает повышенную точность за счет плавающей запятой, уменьшая z-fighting	Лекция 8, стр. 4–6
Reversed depth проекция	Меняет near и far плоскости местами	В матрице проекции far передается как near, near как far. Это обеспечивает более равномерное распределение значений глубины	Лекция 8, стр. 11–14
DepthFunc для непрозрачных объектов	Использует D3D11_COMPARISON_GREATER	Вместо стандартного LESS, для reversed depth используется GREATER: пиксель проходит тест, если его глубина больше текущей в буфере	Лекция 8, стр. 12–14
Очистка буфера глубины	Очистка значением 0.0	В reversed depth дальняя плоскость соответствует 0.0, поэтому очистка выполняется в 0.0 (а не в 1.0)	Лекция 8, стр. 12

4. Геометрия и шейдеры кубов

Область кода	Что делает	Как работает	Теория
InitCube()	Создает два 3D куба с текстурами	1. Определяет 24 вершины с UV-координатами 2. Определяет 36 индексов для 12 треугольников 3. Создает вершинный и индексный буферы 4. Компилирует шейдеры из строкового кода 5. Создает Input Layout	Лекция 3, стр. 5–12; Лекция 6, стр. 11–13
Структура TextureVertex	Описывает формат вершины	Содержит: - float x, y, z: позиция в пространстве - float u, v: текстурные координаты (0.0–1.0)	Лекция 6, стр. 4–7
Вершинный шейдер куба	Преобразует вершины	1. Умножает вершину на матрицу модели (m) 2. Умножает на матрицу вида-проекции (vp) 3. Передает UV-координаты в пиксельный шейдер	Лекция 5, стр. 4–5, 21–24

Область кода	Что делает	Как работает	Теория
Пиксельный шейдер куба	Применяет текстуру	1. Берет текстуру (Texture2D) и сэмплер 2. Читает цвет по UV-координатам (Sample) 3. Возвращает цвет пикселя	Лекция 6, стр. 27

5. Система прозрачных объектов и Blending

Область кода	Что делает	Как работает	Теория
InitTransparentObjects()	Инициализирует прозрачные прямоугольники	1. Создает геометрию прямоугольников (вершины + индексы) 2. Создает bounding boxes для сортировки 3. Компилирует шейдеры для прозрачных объектов 4. Создает константные буферы с цветом и матрицей	Лекция 8, стр. 15–30
Blend State для прозрачных объектов	Настраивает смешивание цветов	Формула: `Color_dest = SrcAlpha * Color_src + (1 - SrcAlpha) * Color_dest` SrcBlend = D3D11_BLEND_SRC_ALPHA DestBlend = D3D11_BLEND_INV_SRC_ALPHA BlendOp = D3D11_BLEND_OP_ADD	Лекция 8, стр. 15–20
Состояние глубины для прозрачных объектов	Отключает запись глубины	DepthWriteMask = ZERO (не записывает глубину) DepthFunc = GREATER (проверка глубины включена, но без записи)	Лекция 8, стр. 27
Сортировка прозрачных объектов	Рендерит от дальнего к ближнему	1. Вычисляет расстояние от камеры до bounding box каждого объекта 2. Сортирует объекты по убыванию расстояния 3. Рендерит в порядке от дальнего к ближнему	Лекция 8, стр. 25–30

6. Skybox с поддержкой Reversed Depth

Область кода	Что делает	Как работает	Теория
InitSkybox()	Инициализирует небесную сферу	1. Генерирует геометрию сферы (CreateSphere) 2. Создает вершинный и индексный буферы 3. Компилирует шейдеры для skybox с reversed depth 4. Загружает 6 текстур для cubemap 5. Создает cubemap текстуру и SRV	Лекция 6, стр. 31–42; Лекция 8, стр. 31–32
Вершинный шейдер skybox	Преобразует сферу с reversed depth	1. Масштабирует сферу, добавляет позицию камеры 2. Вычисляет позицию в пространстве 3. Принудительно выставляет	Лекция 8, стр. 32

Область кода	Что делает	Как работает	Теория
		`output.pos.z = 0.0` для reversed depth	
Состояние глубины для skybox	Skybox рисуется "на бесконечности"	DepthWriteMask = ZERO (не записывает глубину) DepthFunc = GREATER_EQUAL (для reversed depth) Позволяет skybox рисоваться позади всего, но не влиять на буфер глубины	Лекция 8, стр. 31–32

7. Матрицы и камера с Reversed Depth

Область кода	Что делает	Как работает	Теория
UpdateCamera()	Обновляет матрицы камеры	1. Из сферических координат вычисляет позицию камеры 2. Создает матрицу вида (XMMatrixLookAtLH) 3. Создает матрицу проекции с reversed depth (XMMatrixPerspectiveLH с swapped near/far) 4. Перемножает и транспонирует матрицы 5. Записывает в буфер через Map/Unmap	Лекция 5, стр. 21–24; Лекция 8, стр. 12–14
Reversed depth проекция	Создает матрицу с обратными near/far	`XMMatrixPerspectiveLH(halfW*2, halfH*2, far, near)` far передается как near, near как far для reversed depth	Лекция 8, стр. 13–14

8. Рендеринг (главный цикл с правильным порядком)

Область кода	Что делает	Как работает	Теория
Render()	Рендерит один кадр	1. Очищает back buffer и буфер глубины (0.0 для reversed depth) 2. Устанавливает viewport 3. Обновляет камеру 4. Рендерит непрозрачные объекты (два куба) 5. Рендерит skybox 6. Рендерит прозрачные объекты с сортировкой 7. Отображает кадр (Present)	Лекция 3, стр. 52–54; Лекция 8, стр. 33
RenderTransparentObjects()	Рендерит прозрачные объекты	1. Устанавливает blend state и состояние глубины без записи 2. Вычисляет расстояния от камеры до объектов 3. Сортирует объекты от дальнего к ближнему 4. Рендерит в правильном порядке	Лекция 8, стр. 25–30
Порядок рендеринга	Минимизирует overdraw	1. Непрозрачные объекты (можно сортировать от ближнего к дальнему) 2. Skybox (без записи глубины) 3. Прозрачные объекты (от дальнего к ближнему)	Лекция 8, стр. 33

9. Обработка изменения размера окна

Область кода	Что делает	Как работает	Теория
ResizeSwapChain()	Изменяет размер буферов	1. Сбрасывает текущие RTV и DSV 2. Изменяет размер своп-цепи (ResizeBuffers) 3. Пересоздает back buffer и буфер глубины 4. Обновляет радиус skybox (зависит от aspect ratio)	Лекция 2, стр. 28
WM_SIZE сообщение	Обработка изменения размера окна	При изменении размера окна Windows отправляет WM_SIZE. WndProc вызывает ResizeSwapChain для пересоздания буферов с новым размером	Лекция 2, стр. 2–4

10. Вспомогательные функции

Область кода	Что делает	Как работает	Теория
Cleanup()	Освобождает ресурсы	Вызывает Release() для всех COM-объектов DirectX в порядке, обратном созданию	Принципы COM
SAFE_RELEASE макрос	Безопасное освобождение	Проверяет указатель на nullptr перед вызовом Release(), затем обнуляет указатель	Лекция 2, стр. 9
GetBytesPerBlock()	Размер блока DXT сжатия	BC1/DXT1: 8 байт на блок 4×4 BC2/DXT3, BC3/DXT5, BC4-7: 16 байт на блок 4×4	Лекция 6, стр. 17–19
DivUp()	Деление с округлением вверх	`(a + b - 1) / b`. Используется для расчета количества блоков в сжатых текстурах (размер должен быть кратен 4)	Лекция 6, стр. 22

11. Ключевые концепции для защиты (task_5):

1. **Reversed Depth и D32_FLOAT** – борьба с z-fighting, повышенная точность на дальних расстояниях, реализация через обратную матрицу проекции и GREATER для depth test.
2. **Blending для прозрачных объектов** – формула смешивания, настройка blend state, важность отключения записи глубины для прозрачных объектов.
3. **Сортировка прозрачных объектов** – необходимость рендеринга от дальнего к ближнему, вычисление расстояний через bounding boxes.
4. **Skybox с reversed depth** – принудительное выставление z=0 в шейдере и использование GREATER_EQUAL для depth test.
5. **Правильный порядок рендеринга** – непрозрачные → skybox → прозрачные (с сортировкой), минимизация overdraw.

Итог: task_5 добавляет продвинутую систему работы с глубиной (reversed depth + D32_FLOAT), поддержку прозрачности с правильным blending и сортировкой, а также усложненную сцену с двумя кубами и прозрачными объектами, что значительно улучшает визуальную корректность и качество рендеринга.