МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«Дальневосточный федеральный университет»
(ДВФУ)

ШКОЛА ЕСТЕСТВЕННЫХ НАУК

«СОГЛАСОВАНО»
Руководитель ОП
д.ф.-м.н., профессор, академик РАН, Гузев М.А.

_____
(подпись) (Ф.И.О. рук. ОП)
«23» июня 2017 г.

«УТВЕРЖДАЮ»
Заведующая (ий) кафедрой
информатики, математического и компьютерного моделирования
(название кафедры)

_____ Чеботарев А.Ю.
(подпись) (Ф.И.О. зав. каф.)
«23» июня 2017 г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ (РПУД)

Анимационное проектирование

09.03.03 «Прикладная информатика»

*Форма подготовки **очная***

Школа естественных наук ДВФУ
Кафедра информатики, математического и компьютерного моделирования
Курс 4 семестр 8
лекции 34(час.)
практические занятия (час.)
семинарские занятия - час.
лабораторные работы –34 час.
консультации 36
всего часов аудиторной нагрузки 68 (час.)
самостоятельная работа 52(час.)
реферативные работы (количество)
контрольные работы – 5 (количество)
зачет -
экзамен - 8семестр

Рабочая программа составлена в соответствии с требованиями образовательного стандарта, самостоятельно установленного ДВФУ, принятого решением Ученого совета Дальневосточного федерального университета, протокол от 28.01.2016 № 01-16, и введенного в действие приказом ректора ДВФУ от 18.02.2016 № 12-13-235.
Рабочая программа обсуждена на заседании кафедры информатики, математического и компьютерного моделирования, протокол №22 «23» июня 2017 г.
Заведующий кафедрой информатики, математического
и компьютерного моделирования А.Ю. Чеботарёв

Составитель: И.П.Яровенко

АННОТАЦИЯ

РП по дисциплине «Анимационное проектирование» для направления ООП «09.03.03 – прикладная информатика» соответствует рабочей программе дисциплины и требованиям ГОС ВПО.

Изучаемая дисциплина формирует у студентов положительную мотивацию на использование современных методов в разработке и проектировании компьютерных игр и программируемой анимации.

РП, предназначенный для организации учебной работы по дисциплине, содержит основной теоретический материал, маршрутную схему изучения и путеводитель по темам дисциплины, задания для самостоятельной работы и рекомендации по их выполнению, описание контрольных работ с методическими указаниями, глоссарий, каталог образовательных ресурсов в сети Интернет, средства педагогического контроля.

Целью изучения дисциплины «Анимационное проектирование» является ознакомление с основами программирования анимации и компьютерных игр, а также основам ООП Javascript. Javascript выбран в качестве методического языка программирования, т.к. отвечает, как критериям современности, так и востребованности на рынке.

Студент должен овладеть основными методами ООП, программирования анимации, игровыми алгоритмами.

По результатам выполненных самостоятельно каждым студентом работ и активности студента на занятиях выставляется итоговая отметка.

При подготовке к практическим занятиям следует пользоваться настоящими указаниями, лекционным материалом, представленным студентам в электронном виде и рекомендуемой литературой.

Полученные навыки по курсу «Анимация и компьютерные игры» в дальнейшем могут быть применены профессионально, как в области программирования, так и в области разработки компьютерных игр.

СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

Анимационное проектирование

1. **История компьютерных игр.** История и развитие компьютерных игр, сортированная по датам и платформам.
2. **Введение в Javascript, ООП в Javascript.** Основы Javascript. Основы ООП в Javascript: прототипирование, инкапсуляция, наследование и перегрузка.

3. **DOM.** История развития, структура DOM. Элементы, свойства и аттрибуты. Javascript методы по работе с DOM.
4. **События. События клавиатуры.** Введение в событийную модель Javascript. Прослушивание и перехват событий. Виды и перехват событий клавиатуры.
5. **События. События мыши. Drag & Drop.** Виды и перехват событий мыши. Основы техники Drag&Drop.
6. **Canvas.** Обзор HTML элемента Canvas. Javascript методы по работе с Canvas: рисование, трансформации.
7. **JS анимация.** Основные принципы Javascript анимации.

СПИСОК ЛИТЕРАТУРЫ

1. Грибов Д.Е. Macromedia Flash 8. Интерактивная web-анимация. - М. ДМК. 2009. - 672 с.
2. Малев В.В. Общая методика преподавания информатики: Учебное пособие. – Воронеж: ВГПУ, 2005. – 271 с.
3. Франклин Д., Паттон Б. Flash 8. Анимация в интернете. - СПб. Символ Плюс. 2010.-464 с.
4. Мильберн Кен, Крото Джон. Внутренний мир Flash 7 для дизайнеров.
5. К: Диасофт. 2007. - 496 с. Петров М.Н., Молочков В.П. Компьютерная графика. Учебник. – СПб.: Питер, 2004. 736с.
6. Коцюбинский А., Горшев С. Компьютерная графика. М.: Технолоджи. 2009. 320с.
7. Stephen J. Ethier, Christine A. Ethier, «3D Studio MAX in Motion: Basics Using 3D Studio MAX 4.2»
8. Верстак, Владимир, «Анимация в 3ds Max 8. Секреты мастерства (+ CD-ROM)»
9. Тозик Вячеслав , Меженин Александр , Звягин Кирилл, «3ds Max. Трехмерное моделирование и анимация на примерах»
10. Тозик Вячеслав , Меженин Александр , Звягин Кирилл, «3ds Max. Трехмерное моделирование и анимация на примерах (+ CD-ROM)»
11. http://cyberleninka.ru/article/n/razrabotka-multimediynyh-uchebnikov-effektivnost-i-podvodnye-kamni Шахов Э. К., Надеев А. И., Акинин В. В., Голобокова Е. М. РАЗРАБОТКА МУЛЬТИМЕДИЙНЫХ УЧЕБНИКОВ:

ЭФФЕКТИВНОСТЬ И «ПОДВОДНЫЕ КАМНИ» // Прикладная информатика. 2006. №1

12. http://learn.javascript.ru Илья Кантор Современный учебник JavaScript, 2007
13. https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Canvas_tutorial «Canvas Tutorial — Web developer guide»
14. http://window.edu.ru/resource/325/65325 Переверзев С.И. Анимация в Macromedia Flash MX. - 2-е изд. - М.: БИНОМ. Лаборатория знаний, 2009. - 374 с. - (Практикум)
15. http://window.edu.ru/resource/311/77311 Шишкин, В.В. Трехмерное моделирование в среде Blender: учебное пособие / В.В. Шишкин, С.Т. Гераськина, О.Ю. Шишкина. - Ульяновск: УлГТУ, 2010. - 185 с.
16. http://window.edu.ru/resource/091/64091 Майстренко Н.В., Майстренко А.В. Мультимедийные технологии в САПР: учебное пособие. - Тамбов: Изд-во Тамб. гос. техн. ун-та, 2008. - Ч. 1. - 80 с.
17. George Maestri, Sanford Kennedy, Ralph Frantz, Steve Burke, Jason Greene, Eric Greenleif, Jeremy Hubbell, Paul Kakert, Randy Kreitzman, Bob Lamb, «Inside 3d Studio Max: Animation (Inside 3D Studio MAX)»
18. Иваницкий К. А., Комягин В. Б., Каменский П. А., Резников Ф. А., «100% самоучитель. Трехмерная графика и анимация 3ds Max 2009»
19. Аллен, Дамиан, «Motion 3. Дизайн и анимация графики в Final Cut Studio 2 (+ DVD-ROM)»
20. Потапкин, А.; Кучвальский, Д., «3D studio MAX»
21. Mark Williamson, «Inside 3D Studio MAX 2»

Практические задания

1. Цель работы Усовершенствовать ИИ **компьютерных** противников для стратегической **компьютерной игры**.
2. Задание: 1. Выбрать **игру**, изучить её исходный код и архитектуру. 2. Проанализировать существующий ИИ, выявить его недостатки.

3. Цель работы Усовершенствовать ИИ **компьютерных** противников для **компьютерной игры** Wesnoth. Задание на курсовую работу заключается в следующем: 1. Изучить исходный код и архитектуру **игры** Wesnoth. 2. Проанализировать существующий ИИ, выявить его недостатки.

4. Доработка **компьютерной игры** HedgeWars

5. Социальная сеть любителей **компьютерных игр** Want2Play (клиентская часть)

6. HedgeWars -- open- source аналог известной **компьютерной игры** Worms. Задачи: 1) Доработка генерации ландшафта 2) Доработка искусственного интеллекта

7. Карточная **игра** с веб- интерфейсом

Контрольно-измерительные материалы

Геометрия. Создать следующие объекты на JavaScript, моделирующие геометрические:

- Примитив "точка", характеризующаяся координатами.
- Простой двумерный геометрический объект: прямоугольник, окружность, треугольник. Каждый такой объект характеризуется точками на плоскости (и радиусом, если это окружность). Добавить метод вычисления площади.
- Простой трехмерный объект, характеризующийся двумерным объектом и дополнительными параметрами. Например, параллелепипед может быть охарактеризован прямоугольником и высотой. Добавить метод вычисления объема.

1. **Снег на DOM.** Используя нативные методы и свойства для работы с DOM, реализовать "снег". А именно. Формируется квадратное поле из ячеек (элементы-контейнеры DOM). Случайным образом в любой из ячеек верхней границы поля появляются "снежинки" (элемент DOM) и по определенному промежутку времени начинают "падать" к нижней границе поля, последовательно перемещаясь из одной ячейки в другую. По достижении нижней границы "снежинка замирает". Если следующая "снежинка" попадает на любую лежащую "снежинку", то пробует с нее "скатиться", т.е. пробует продолжить движение по диагонали вниз (вниз и влево или вниз и вправо). Если же при движении по диагонали "снежинке" не встречаются свободные ячейки, то она замирает. Если некоторые участки поля забиты "снежинками" снизу доверху, то снежинки продолжают появляться там, где еще имеется свободна верхняя граница поля. Таким образом, поле NxN в процессе "падения" N^2 "снежинок" должно быть полностью покрыто "снегом".
2. **Тетрис на DOM.** На основе предыдущего задания. На поле 10 на 20 клеток реализовать игру "Тетрис". Требования:
   - Реализовать из точек предыдущего задания классические фигуры игры. Фигуры передвигаются сверху вниз, появляясь в центре поля.

- Количество разных фигур должно быть не менее 10, учитывая, что фигура и аналогичная, повернутая на 90 градусов, в данном задании принимаются за разные.
- Фигуры можно передвигать влево и вправо во время их движения, используя клавиатуру. Как только фигура достигла позиции, ниже которой движение невозможно - управление фигурой становится невозможным.
- Новые фигуры выбираются случайно и начинают свое движение, тогда, когда окончательно остановилась предыдущая.
- При заполнении полной строки клеток - строка очищается, верхние заполненные клетки сдвигаются вниз.
- С помощью клавиатуры должна быть возможность поставить игру на паузу и наоборот. В режиме паузы фигуры не двигаются, и управление ими невозможно.

3. **Drag&Drop. Puzzle.** Реализовать головоломку, заключающуюся в том, чтобы из различных частей собрать целое изображение. Игрок указывает изображение (не важно, каким методом: через document.location.hash, элементы формы и т.п.), которое разбивается на равные фрагменты. Фрагменты помещаются в специальную область "кучу". Рядом с кучей доступно поле в виде сетки, по размерам совпадающее с исходным изображением. Каждая ячейка сетки по размеру совпадает с каждым фрагментом из кучи. Используя мышку, игрок может перетащить фрагмент из кучи в любую область экрана, причем: фрагмент можно переместить в любую точку кучи (при захвате фрагмента (зажали кнопку мыши над ним), он визуально начинает отображаться выше всех остальных фрагментов в куче); если фрагмент переместили таким образом, что он находится, как за пределами кучи, так и за пределами поля - фрагмент возвращается на то место, откуда его взяли; если фрагмент попадает в область поля, он занимает ближайшую к нему ячейку сетки (недопустима ситуация, когда фрагмент, находясь в области

поля, не находится целиком ни в одной ячейке). Фрагменты можно перемещать из ячейки в ячейку. Таким образом, на сетке из фрагментов собирается исходное изображение. Программа должна определить правильное расположение фрагментов на сетке - тем самым выдать сообщение об успешном завершении игры.

4. **Canvas. Heartbeat.** Используя canvas отобразить анимацию бьющегося сердца, которое при этом совершает обороты на 360 градусов вокруг себя. Требования:
   - Для отрисовки сердца использовать кривые и/или пути; желательно кривые Безье.
   - Анимация сердцебиения должна заключаться в периодическом увеличении изображения сердца до определенных размеров с последующим уменьшением до, так же, определенных размеров.
   - При этом, с момента максимального размера изображения сердца и до определенного момента в течение его уменьшения от изображения сердца должен отделяться ореол по форме совпадающий с исходным изображением, который плавно растворяется, увеличиваясь в размерах.
   - Наконец, анимационное изображение должно вращаться вокруг себя. Скорость, направления, цвета и размеры задаются произвольно.
5. **Canvas. Пушка.** Реализовать игру, заключающуюся в поражении случайно появляющихся целей ядром, выпущенным из пушки. Пушка позиционируется в нижнем левом углу игрового поля. Имеет возможность поворачивать дуло на 90 градусов (от вертикального положения до горизонтального). Пушка имеет силу залпа, которая регулируется игроком в процессе стрельбы (например, при зажатии клавиши происходит увеличение силы залпа, и при отпускании клавиши происходит выстрел). Углом поворота дула следует управлять с клавиатуры. После выстрела сила залпа обнуляется. Ядро представлено окружностью. Ядро имеет массу. При выстреле ядро в горизонтальном направлении движется прямолинейно, в вертикальном направлении -

равноускоренно (учесть силу тяжести). Само собой необходимо связать массу ядра и силу залпа: чем меньше сила залпа - тем медленнее полетит ядро. Пока ядро находится в движении, произвести выстрел нельзя. При достижении границ игрового поля ядро прекращает свое движение. Исключение составляет верхняя граница поля - ядро может свободно вылетать за нее. Предполагается, что за пределами верхнего поля ядро движется аналогично, т.е. рано или поздно оно должно вернуться в поле видимости (конечно, с учетом угла полета). Дополнительные баллы за реализацию автомасштабирования поля, чтобы ядро всегда было на виду. Реализация цели может быть какой угодно. Главное, при попадании ядра в цель, последняя исчезает, ядро прекращает движение, игроку присваиваются очки (их нужно отображать), генерируется новая цель. Следует самостоятельно решить и реализовать, что будет, если ядро упадет на пушку.