Course Title: SDE 104 Game Programming and Development I

Course Leader: David Maruszewski

Expected Learning Outcomes for Course

- Demonstrate problem solving skills through verbal and written media
- Apply principles of video game design and issues designers face in game creation
- Define variables, functions and random events
- Create narrative environments, stories and characters associated with games
- Define game genres, and the differences between them, with reference to creating each
- Demonstrate game balancing
- Demonstrate fundamentals of gaming and simulation design in the stages of concept and development
- Work productively in a team environment
- Adeptly simulate in 2 dimensions and 3 dimensions
- Analyze, select and apply tools appropriate for a specific solution
- Logically formulate scripts and/or programs to solve problems
- Understand and articulate interactivity in the gaming industry, including the connectivity between computer art and programming
- Apply programming theory in practical applications
- Demonstrate problem solving skills through verbal and written media
- Apply rudimentary Physics and Trigonometry principles

Assessment

Students are required to complete a final project which was created to test skills gained throughout the course. An assessment is used to evaluate the students' progress.

Three exams are issued to help confirm the findings of the project grade.

Validation

Currently, all evaluations are held for two semesters and composite data is used to show trends. Certain chosen questions on exams should help verify or contradict findings.

Results

1. Students can improve verbally, but on the whole, do much better than writing. Many students seem timid expressing themselves in writing.
2. Students will take more opportunity to do complete work with software than with any other venue.
3. Gaming industry knowledge is easily understood by students, although sometimes hard to retain.
4. Students enjoy the creative process, often getting involved in their section of a project.
5. Students can work well in groups but some will take advantage of the group. Some of the people in the group will be timid in making those individuals accountable.
6. Students like to start from scratch. They like to reinvent the wheel. They also like to create their own material.

7. The text has been changed many times since the inception of this course. There is a limited amount of books that can be used. This translates into confusion as sometimes the students don't see the "big picture."
8. Game theory in conjunction with a plausible design is a hard to grasp for students. They often come up with juvenile or too simple of ideas to work through, and then lack implementation skills. This is a function of their critical thinking.
9. Programming was too heavy in the course and repelled those who wanted to learn about making games but didn't want to opt for this elective. We added Development into the title and content.

Follow-up

1. More writing assignments were implemented and worked. Explanations were more guided and had continued success. Giving students clear goals is good as well. In one project they must write instructions on how to use their project. Understanding the pitfalls of people not being able to do their directions has impact on their writing.
2. Game engine software was integrated into the teaching model to reach students better. The only problem with this is that it can distract from their other needed work. Segregating times in class has improved the distraction.
3. Slides were used to generate discussion. We then implemented these ideas in an ongoing game that we are creating. This is ongoing. Having students get through first the necessary components needed to game program helped when we entered into the creation phase. Then, they could employ their knowledge.
4. Projects definitely help the student get interested and learn more. Having a working project brought to fruition is even better. We've continued working with Unity. Giving student specific tasks in programming and development has improved learning.
5. Peer reviews were created to grade all class mates. Still some were reticent to potentially hurt others grades. A different rubric is used and has helped, but students still are not fully honest. Having student comments clears things. I need to be the final word on grade after reading all of the reviews. That seems to be the most fair of grading policies.
6. This class will focus on coming in on a project and working on it. Students will have to look at past design documents and work. They will also have to use past resources. Students don't like this but it is an industry necessity. I get a little resistance, but the point gets across.
7. A comprehensive teaching model should be sought where books can be applied to aid the teaching. This is as opposed to doing the opposite where the book drives the content. This was employed and worked moderately well. There is really no best book or accepted book for that matter. A frame work that I create from notes and slides guides the students better.
8. I will lead with initial ideas for project work including theme, and then teach game theory and design from that theme. Students can mature existing ideas better than come up with their own most times. This has been working well. Keeping them focused is now the issue. Irrelevant work can come up because they don't want to make the game created. Clear grading components can help aid this.
9. This had a good effect as students were engaged. Also, when you focus on programming, the game making becomes secondary. This change allowed for these parts to be treated equally. My outline for the class could now have a better integration of the game creation.

Budget Justification

- A modern and relevant game engine is a great tool to get students interested and understand the theory and rudimentary application that we do in class.
- Game theory works well with programming. It allows them something to program. The game engine and other support software like Microsoft Visual Studio are useful for the learning experience.
- Disk space to work on in class that will not be deleted is needed to use open source software.
- A curriculum/content creation aid would be helpful.