Course Outcome Guide (COG)

Course Title: SDE 205 Game Programming and Development II

Course Leader: Audra Martenot

Expected Learning Outcomes for Course

- Understand principles of video game production
- Create event driven programming environments and algorithms
- Develop game events through the use of C++
- Level and Mod design analysis
- Understand the differences in programming and developing of different genre games
- 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 and artistic theory in practical applications
- Demonstrate problem solving skills through verbal and written media
- Apply rudimentary Physics and Trigonometry principles

Assessment

(How do students demonstrate achievement of these outcomes?)

Students are required to complete their second project which was created to test skills gained throughout the course. The project is then graded with a "grade sheet" which looks at skills and outcomes. A full assessment rubric may be created in the future.

A supplemental (3rd of the semester) exam is issued to help confirm the findings of the project grade.

Validation

(What methods are used to validate your assessment?)

Currently, all grades sheets 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

(What does the data show?)

1. Error trapping, and creating flawless algorithms is challenging to the student
2. Debugging for some students is very easy, but others seem to be difficult. The split is about 50/50.
3. Being able to explain their programming is something students do not do well. They don't give results. They give commentary.
4. Much material is in SDE 104/205. Students may feel overloaded or we may not get through enough helpful material.
5. Students struggle with the software (UDK).
6. Books are insufficient to drive content especially in outline and framework.

7. 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

(How have you used the data to improve student learning?)

1. Students will be put in teams in order to help them with debugging and error trapping. This has initially worked well. Initial assessment of this is good.
2. I pair up stronger students with weaker students to help them out. This can be beneficial to both if done correctly. However, it drains class time. I went with a more to be completed at home atmosphere this semester and it worked, but I need to test this more.
3. Initially this has improved. Students tend to be trained to be opinionated in their pasts. A little of un-training is needed. However, getting them to focus on what they are trying to achieve in their code works well. I had good students this semester, so the good results may not be indicative of other students groups in the future.
4. We dropped the C++ component of this course, and taught more to the UnrealScript and generic script reading. The artificial intelligence component was kept. This worked much better, but students don't like the artificial intelligence. Most indicators point to the fact that they find it too hard (scores and words.) Integrating all theory into the practice of game making helped and it appears that outcomes were achieved.
5. Unity was used and overall better understood. C# was used for scripting. Since it is taught in our Intro to OOP class (SDE 130) students feel more comfortable with it. This fixed a lot of issues.
6. We use book as a support for only the most general of topics. Tutorials and web searches are added to garner individual learning. This allows students to work in the direction of their future career requirements.
7. 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

(What resources are necessary to improve student learning?)

1. A game engine is used in this class and helps to get the students interested in programming issues.
2. Thawspace would be beneficial for downloading of Open Source material.
3. Webinar software would work well in this course. A completely online course will be eventually developed.