AP Physics 2 Project: Physics Computer Based App/Sim

Objective: Design and build a working physics based application/simulation that demonstrates general physics concepts or specific concepts.

Rules and Specifications:

- Teams- may be done individually or in pairs.
- The app/sim must use physics knowledge and skills learned and developed during the school year.
- The app/sim should have at least 3 variables that can be changed so that the effect of those changes can be observed and analyzed using the application/simulation. A "Help" menu with basic instructions or troubleshooting tips should be included.
- Project is due **before homeroom, Monday, June 12, 2017**. Presentations and peer evaluation will be conducted in class during the following classes (schedule to be made).

Project will consist of the following main parts:

The App/Sim
Instructions
Presentation of the App/Sim
Evaluation

I. The Application/Simulation (40% of grade)

The application/simulation code and user interface must be structurally sound/stable, neat and easy to follow.

The idea for the application/simulation may be taken from already made ones, but must be adjusted to fit the objective and built/coded by the team. Credit must be given to original designer in the Instructions for the game. (Example: A physics application based off of energy, Rube Goldberg Machines and the game "Mouse Trap" must be made to center around physics and credit should be given to the Ideal Toy Company.)

Code used in building of game may stem from open source code, however the overall code must be written/built by the students. **Make sure all components necessary to run the program is available on due date.**

II. Instructions and Code (30% of grade)

Instructions for the application/simulation must include:

Title of the app/sim with team member names (any credit to be given to existing app/sim)
Purpose of the application/simulation
Basic instructions of use/Help menu
Responses to game reflection questions

III. Presentation (20% of grade)

Presentation should be done by team (each must participate in presentation- absent students will have to present after school to get their portion of the presentation grade). Presentation must be between 3 & 7 minutes in length. The presentation should be done as a "sales" pitch to try to get the rest of the class to want to "buy" your team's application/simulation. The presentation may be done using PowerPoint, Prezi, or similar computer based presentation format and should include the purpose of the application/simulation, pictures/screenshots of application/simulation, brief description of the app/sim, highlights from the simulation/selling features, etc.

IV. Peer & Code Evaluation (10% of grade)

After all presentations have been given, peers will be able to try out the application/simulation and rate the app/sim based on first impressions, being user-friendly, playability/difficulty, and creativity. The code will also be evaluated by a computer software engineer.

Due Date for the Game- Monday June 12th (Before Homeroom)

Hints, Tips & Suggestions

Make roles for each team member. Some suggested roles that can play off of different strengths/personalities:

- Visionary- design and purpose of app/sim
- Graphic Artist- user interface
- Software Engineer- coding
- Salesperson- presentation

NOTE: All members should be equally involved in all parts, however each part can be overseen by a different student leader.

Programmed computer games that run on common platforms (such as Java, HTML5, Python, etc.) are recommended.

Stuck for ideas? Think of the components of one of your favorite games and try to incorporate and adjust them to fit the project's requirements. Just remember, though, that knowledge of physics is essential to play, advance and complete your game!

Your app/sim will be played by other students who wants to be WOWed by your game. Take care in the making of your app/sim. This should be something that you will be proud of when the day is done. Once you get the basics, start paying attention to the details.

REFLECTION QUESTIONS

Game Name: [Your Game's Name Here]

Overall Description: Part of 20% instructional booklet

General Description
[What style is it (First Person Shooter, Turn Based, Real Time Strategy, etc…)? What is the objective? How many people can play?]

Rules
[How do you play? How can you win? How can you lose? Can somebody cheat?]

4 Points: Design

Software Design Approach:
[Describe how you thought about the design of the software. What techniques (if any) did you employ: stepwise refinement/functional decomposition, Object-Oriented Design, trial-and-error, cowboy coding? How did the coding go? What factors were critical to have sorted out prior to the start of development?]

2 Points: Requirements Choice

Software:
[Do you have any operating system requirements? Browser requirements (is it web based)?]

Hardware:
[Do you have any specific hardware requirements? What platforms can it run on (eg: mobile, desktop/laptop only)?]

5 Points: Implementation

Language:
[What programming language did you use and why did you choose that? What were some pro's and con's for this language?]

Frameworks:
[Did you leverage any existing game design frameworks (DirectX, OpenGL, etc..) or is this completely home built? What factors lead to your decisions?]

Algorithms:
[Describe any notable algorithm implementations used. Examples: quicksort, merge sort, binary search, or something of your own? Why did you use them?]

2 Points: Strategy

Strategies:
[Describe some limitations to the application. How did you program them? How did you balance things so the application or simulation could be used to accurately represent a particular physics concept without including too many concepts?]

Artificial Intelligence:
[Do you have any AI? If so, describe how it does its “thinking”]

5 Points

Testing:
[Did you do any unit testing? What parts were the most critical for testing and why?
Did you do any play testing? How did users respond afterwards? What changes (if any) did you make based upon user feedback?]

Memorable Bug:
[Describe some background for one of the more memorable programming bugs you encountered. What kind of issue was it: “off-by-1”, “typo”, “cut-and-paste”, oversight, logic error, etc.?]

[How did you discover it? How did you isolate and fix it? DID you fix it? :) How do you know it’s fixed?]

2 Points

Next Time:
[What would you do differently if you had to do it again? What learning experiences did you take away from this? How will this help you on future projects?]

RUBRIC:

		Max Points
GAME	On time (10pts) Requires physics knowledge and skills to play (10 pts) Game is playable (10 pts) GUI is well designed, neat and presentable (5 pts) Game time fits into a 45-minute time frame (5 pts)	**40**
INSTRUCTIONS	Title of the game with team member names with credit if necessary (2pts) Number of players/teams needed to play the game & playing time (2pts) Objective of the game (2pts) Rules and how to win- clear, concise and accomplishable goals (4pts) *Design Approach (4pts) *Requirement Choice (2pts) *Implementation (5pts) *Strategy (2pts) *Reflection (7pts)	**30**
PRESENTATION	Team Participation (5pts) – any absent student will need to present after school to get their 5 points. Length: 3-7 minutes (3pts) Information: objective of the game (2pts) action shot (2pts) brief description of rules (3pts) highlights/selling features (2pts) Presentation is well thought out, clear, appropriate for audience, fluid (3pts)	**20**
EVALUATION	Peer Evaluation- Averaged score of peer evaluations on presentation and game (5pts) *Code Evaluation (5pts) *Evaluated/Assessed by computer software engineer, Mr. Rudowski	**10**

Note: Some portions of the rubric that have a larger point value may have an additional breakdown when being scored by the teacher.