Expeditionary Dahlgren Math Education Projects

Each topic is presented in three ways. The Teaching page will introduce the need for some element in a program, but will use terminology and concepts common to programming and math. The calculation page will have the students focus on the math. The programming page will have them apply the concept in a program. In each case the student will be working within a supportive framework that only requires him/her to use the knowledge gained thus far e.g. Only typing in the X axis location rather than both axis. Students can use the link provided to access the challenge and move through all three pages, submitting the final page for review when complete. After completing all challenges, the student should have sufficient trigonometry to aid in programming. Students can move on to a more complicated project. If successful, these students will also be able to apply the skills learned in programming to math classwork.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | Topic(s) | Teaching | Calculating | Programming |
| 1 | X axis; distance in 1 variable  [[Link](https://bsdillon.github.io/cs200_Spring2020/GoldField/alternate/cartesian1.html)] | Students are introduced to the X axis and motion in the horizontal direction described in terms of numbers and an X variable. | Students use a single-variable distance formula to calculate the distance between the hero and a gold bar. | Replicate the hero on the field. |
| 2 | Y axis; boundary conditions  [[Link](https://bsdillon.github.io/cs200_Spring2020/GoldField/alternate/cartesian2.html)] | Revision of the same topics from project #1 but for the Y axis. Y axis follows mathematical, not computer, conventions. | Use < and > to express boundary conditions based on character locations. | Replicate the hero on the vertical field. |
| 3 | Cartesian plane; ordered pairs  [[Link](https://bsdillon.github.io/cs200_Spring2020/GoldField/alternate/cartesian3.html)] | Using both axis on a plane, describe position of the character in terms of both variables. | Use ordered pairs to describe the location of various figures. | Replicate the plane, create figures, and set them in specific locations. |
| 4 | Component vectors and slope | Consider distance in 1 variable for both X and Y axis. Use these calculations to describe the slope of a line between two figures. | Correctly identify and insert x and y values into a formula to calculate the slope. | Place to figures on the plane and then describe and add all the component vectors. |
| 5 | Magnitude, Pythagorean theorem, distance | Revisit the concept of magnitude/distance in single variables and then the Pythagorean theorem to calculate the same quantities in two variables. | Calculate the distance between various figures on the page. | Revisit the challenge from #4, adding distance calculations to various figures. |
| 6 | Trig functions | Manipulate the Pythagorean theorem to derive sine cosine, and tangent functions. | Use these functions to reconstruct component vectors or magnitudes based on incomplete information. |  |
| 7 | Unit circle; +/- θ | Using figures on the screen, consider the unit circle in terms of sensing radius and introduce positive and negative measures of angles in degrees. | Resize the arc angle of a sector to measure the angle from one figure to another. | Add a sensing ring around each figure. |
| 8 | Realism: Y inversion | Introduce students to the concept that computers invert the y dimension. Keep the mathematical conventions visually, exposing the true computer values and the conversion to/from standard mathematical notation. | Revise calculations of distance in Y dimension, distance in two dimensions, and location from ordered pairs. |  |
| 9 | Realism: Radians | Introduce the concept of radians in the unit circle as these are used by standard trig functions in JavaScript. | Revise the concepts in #7 and translation to/from radians. | Use radians to describe the distance and bearing from one figure to all others. |
| 10 | Find θ from component vectors | Use the inverse tangent function to solve for theta and find an angle based on component vectors. | Manipulate the equation to solve for X, Y, or θ. |  |
| 11 | Actors, Sensing; if logic; increase granularity of field. | Introduce the concept of an actor/agent with OODA loop reactions with simple control structures. | Use distance formula and tangent to calculate which of several monsters can sense the hero. |  |
| 12 | Optimal movement; off by one errors | Use theta to predict the optimal direction for one actor to “chase” another. Use <, >, ≤, and ≥ to discuss off by one errors in terms of sensing distance and theta. | Determine the use of comparisons to create certain actor behaviors. |  |
| 13 | Interval, while | Unify sensing and movement into the OODA loop controlled by a time interval and a while loop. |  | Alter the interval to increase the apparent “speed” of a chase. |
| 14 | Velocity components | For a given velocity or step size and direction, calculate the x and y component vectors using sign. | Use a stepwise process for the monster to “catch” the hero. |  |
| 15 | Randomness, polar and rectangular coordinates. | Discuss the nature of pseudo-randomness. Revise use of random numbers in previous assignments. | Use base and range values for both variables to create a target distribution of figures inside a target square, rectangle, circle, or sector. |  |
| 16 | Arrays, for loops | Discuss the concept of an array, use of indices and the close relationship with for loops. | Use a for loop to find the “center” of a group of figures. |  |