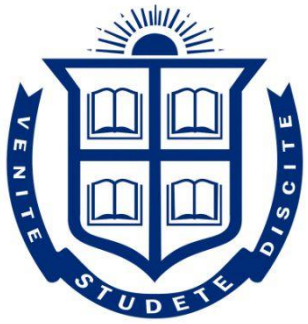


BLAIR ACADEMY

Advanced Survey: Computer Science Portfolio 2025-26

Xavion Mirchandani
Class of 2029

Project Description

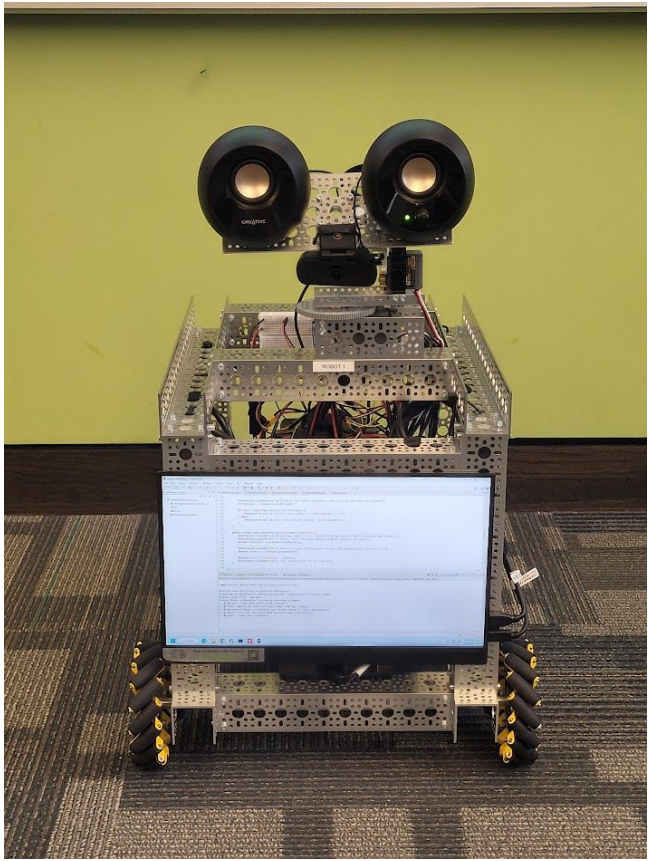
The portfolio assignment was to write write a larger version of MyProgram for the robot. The program was supposed to demonstrate the use of all of the robot methods in the three classes we've developed together (RobotSpeech, RobotSound, and Robot). The portfolio was supposed to demonstrate as many of the coding concepts covered this year as possible. We were allowed to review assignments and presentations in our Google classroom, but we had to think of things that we had done together, such as prompting a user to choose from a menu, looping elements of code multiple times, etc. We also had to think creatively about what we wanted the robot to do, and use the speech capabilities to interact with the user. The assignment is deliberately open ended, and we could have the robot do whatever you wanted, within the limits of the material covered in class this year.

Project Design

My aim with this portfolio assignment was to create a robot that would allow the users to play Jeopardy and would act as a game show host. My aim was to have to robot show the users the game board (which updates as the rounds go on) and allow them to select the category and point value, then calculate their score based on the answer and who answered. I chose this topic because Jeopardy is something that my family and I watch, especially with my grandmother, and it is a great social activity that everyone finds fun. Thus, I wanted to create this interactive game so that everyone in my community can also have fun.

Hardware Framework

The code for the project was loaded onto a robot in the classroom. The robot used a PC running Windows. The code was written in Java, and compiled using Eclipse. The mechanics were operated by an Arduino board attached to the PC using a USB cable. Several Java libraries were loaded onto the robot, and students also wrote and developed their own Java class files.



Robot Commands

```
Sample command set:

*Robot Mechanical Actions*
Robot robby = new Robot();         // opens serial port on Arduino board, resetting servos to 'initial position'
robby.continueOperating(2000);     // time in milliseconds
robby.rotateHeadHorizontally(45); // Range 0-180, straight 90, default 90
robby.rotateHeadVertically(59);    // Range 53-129, level 68, default 92
robby.setWheel1Speed(20);          // Range 0-100, min 20, recommend 25
robby.setWheel2Speed(20);          // Range 0-100, min 20, recommend 25
robby.setWheel3Speed(20);          // Range 0-100, min 20, recommend 25
robby.setWheel4Speed(20);          // Range 0-100, min 20, recommend 25
robby.changeWheelSpeed(20);        // Range 0-100, min 12, recommend 20
robby.reverseWheels(20);           // Range 0-100, min 12, recommend 20
robby.stopWheels();                // Stops all 4 wheels
robby.resetHeadPosition();         // Positions head to look straight forward  ('default position')
robby.strafeLeft(20);              // Range 0-100, min 20, recommend 25
robby.strafeRight(20);             // Range 0-100, min 20, recommend 25
robby.turnLeft(20);                // Range 0-100, min 15, recommend 20
robby.turnRight(20);               // Range 0-100, min 15, recommend 20

*Robot Speech Synthesis*
RobotSpeech.say("Hello World!");
RobotSpeech.sayAndPrint("Hello World!");

*Robot Sound Production*
RobotSound.playSound("filename"); // Must be in .wav format, .wav automatically appended
```

The robot has a set of commands used to interact with it. The image shown to the left list out these commands and allow for us to work with the robot to make it do things.

Coding Concepts

When coming up with the project, although I had to incorporate all of the coding concepts we had learned up until the Integer and Double classes, some were more prevalent than others. I mainly used classes and methods to create objects of different roles and to separate code blocks and allow for more organized coding. Additionally, I used interfaces to label/plan out classes, and I used print statements and the Keyboard Class to display the information for the game and to receive user input.

Here is a link to a full list of coding concepts: Link

Project Code

The code is linked here and I have provided detailed comments in it to help with readability. I have also provided a snippet of code from the Board Class on the right.

Here is the link to the full code:

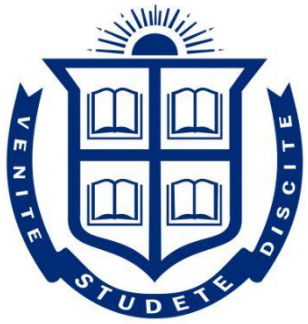
Link

src > computerScienceProject > Board.java > Board > readCategories()

```
1   package computerScienceProject;
2
3   public class Board implements BoardInterface {
4       public boolean[][] boardOpened;
5       public int[][] pointValues;
6       private Question[][] boardValues;
7       private String[] categories;
8
9       private String dashes;
10      private int boardMultiplier = 1;
11      private long maxPoints;
12
13      // Initialization
14      // Initializes the board with the given stage for standard play
15      public Board(int stage) {···
19
20      // Initializes the board with the given stage and a point multiplier
21      public Board(int stage, int multiplier) {···
25
26      // Initializes a custom-sized board with a point multiplier
27      public Board(int boardMultiplier, int xSize, int ySize) {···
30
31      // Sets up the internal arrays and variables for a custom-sized board
32      private void setBoard(int boardMultiplier, int xSize, int ySize) {···
55
56      // Sets up the internal arrays and variables for a standard 5x6 board
57      private void setBoard(int boardNumber, int multiplier) {···
76
77      // Loads predefined categories and questions for specific board stages
78      private void loadBoard(int boardNumber) {···
149
150     // Used Methods
151     // Prints the current state of the board including categories and available point values
152     public void printBoard() {···
191
192     // Checks if all questions on the board have been answered
193     public boolean isBoardEmpty() {···
203
204     // Marks all questions on the board as answered to effectively clear it
205     public void clearBoard() {···
212
213     // Retrieves the question text for a specific point and category index
214     public String getQuestion(int pointIndex, int categoryIndex) {···
217
218     // Retrieves the correct answer for a specific point and category index
219     public String getAnswer(int pointIndex, int categoryIndex) {···
222
223     // Retrieves the name of the category at the specified index
224     public String getCategory(int categoryIndex) {···
227
228     // Returns the total number of categories on the board
229     public int getCategoryLength() {···
232
233     // Uses the robot to audibly read out all the categories on the board
234     public void readCategories() {···
```

Possible Next Steps

Upon conclusion of the project, we were asked how we could improve our projects. If I was to further work on this project, I would incorporate the Gemini API so that users don't have to determine whether their answers are correct and instead they write their answer and Gemini determines for them. Additionally, I would incorporate saving of games and boards which would allow users to save and load their games (so if they are in the middle of a game they can stop and come back to it) and which would allow users to create and save their custom boards, allowing them to reuse their boards and creating fun and customizable jeopardy games for each user.



BLAIR ACADEMY

Advanced Survey: Computer Science Portfolio 2025-26

Xavion Mirchandani
Class of 2029