LESSON: CodeX Lucky Charms Project (March)		Time: 50 minutes
Overview: This project uses the graphics capabilities of CodeX to draw clovers using circles and rectangles. It also uses probability and random numbers to create a simple game of chance. Part 1: You can choose to give your students the starter code and follow slides 6-8 (skipping 9-17). This code has the clovers already programmed. Or skip slides 6-8 and follow slides 9-17 to have your students go through the process to draw the clovers. Part 2: Use the clovers and probability to make a simple game of chance – when you press a button, do you get a 3-leaf clover or a 4-leaf clover. Students will create small functions and put everything together in the main program for a simple working game. You can stop here or continue to Part 3. Part 3: Use variables to add more clovers and make the game more difficult. Part 4: Add some extras, like an introduction and ending message. Several ideas for an extension are given on the last slide. These are optional, and instructions are not given for the extensions. They are suggestions for students who want to challenge themselves to do more and apply their learning to this project.		**Coding Objectives:** ● I can use the draw features of CodeX to draw circles and rectangles. ● I can use a random number and probability to determine an outcome. ● I can define and call a function with a return. ● I can define and call a function with parameters. ● I can write code that determines if a game is won. ● I can increment a variable that counts button presses. ● I can use variables for locations on the coordinate grid system to duplicate images drawn with circles and rectangles.
Grades 6-8 CS Standards: **2-CS-03** Systematically identify and fix problems with computing devices and their components. **2-AP-11** Create clearly named variables that represent different data types and perform operations on their values. **2-AP-12** Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. **2-AP-13** Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. **2-AP-14** Create procedures with parameters to organize code and make it easier to reuse. **2-AP-17** Systematically test and refine programs using a range of test cases.	**Grades 9-10 CS Standards:** **3A-CS-03** Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors. **3A-AP-13** Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests. **3A-AP-16** Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions. **3A-AP-17** Decompose problems into smaller components through systematic analysis, using constructs such as procedures, or independent but interrelated programs. **3A-AP-18** Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.	**Grades 11-12 CS Standards:** **3B-AP-10** Use and adapt classic algorithms to solve computational problems. **3B-AP-14** Construct solutions to problems using student-created components, such as procedures, modules and/or objects. **3B-AP-16** Demonstrate code reuse by creating programming solutions using libraries and APIs. **3B-AP-21** Develop and use a series of test cases to verify that a program performs according to its design specifications. **3B-AP-22** Modify an existing program to add additional functionality and discuss intended and unintended implications.

Preparation:

- Download slides
- Be familiar with the final code
- Read through the teaching guide
- Optional: Use the accessibility guide as needed.

In the folder:

- Lucky Charms project slides
- Lucky Charms starter code
- Lucky Charms code solutions for parts 2, 3 and 4.

Agenda:

- Warm-up (5 minutes)
- Complete program using slides (45 minutes) – could be longer with extensions

Teacher Notes:

- Decide if you want to give your students the starter code and skip programming the clovers, or if you want your students to do all the code. Contributing factors include the time you have for the project and the abilities of your students.
- This project can be completed individually or using pair programming.
- Almost all mistakes made by students are typing mistakes. If students get errors when they run their code, first look over the code for spelling, punctuation and especially indenting.

Other Extension Ideas:

- Keep track of the highest number of spins needed to “win” and display in the ending with the lowest number of spins.
- Use a random number for the probability.
- Add a fourth clover to the game (or even more!).
- Add levels to the game. For example, press U for an easy game using high probabilities, or press D for hard using low probabilities.
- Add a third image besides the clovers (maybe a hat). Assign another probability of how often it shows up. If it shows up all three times, you lose – game over!

Cross-curricular Connections:

- **LANGUAGE ARTS:** Have students come up with their own tradition for St. Patrick’s Day and write about it. Or have them come up with their own holiday and explain it orally or in writing.
- **MATH:** Deciding the location of each leaf in the clover can take some math. Work through the math together and see who can calculate their way to a perfect clover.
- **MATH:** This project uses the concept of probability. Have a lesson on probability and do other fun unplugged activities with probability.
- **VISUAL ARTS:** This project shows how to draw using circles and rectangles (and text). Use graph paper and design your own image. Then write the code to display your image.
- **SOCIAL STUDIES:** Students can research the background of St. Patrick’s Day and how it is celebrated across the world.

Teaching Guide

Warm-up (5 minutes)

Slide 2 – This project is a fun way to bring in St. Patrick’s Day and some of its traditions. You can warm up by:

- Making a list of traditions or ways people celebrate the holiday.
- People often talk about the luck of the Irish, or the lucky pot of gold at the end of a rainbow, or finding a lucky 4-leaf clover.
- As a transition, use the question on the slide “Are you feeling lucky?”

Create/Run the Program (45 minutes)

💻 This project can be completed individually or with pair programming. Students should already have an account in CodeSpace.

💡 **Teaching tip:**

The instructions for the project are not included in the interactive textbook of CodeSpace. Download and follow the slides. They include step-by-step instructions as well as code segments to guide students through the program code creation.

This project can be completed as a class by showing the slides on a large screen. Or you can give students the slides and let them work at their own pace. It is also a great project for students who finish work early.

💻 **Slides 3-4**

These slides give instructions on getting started. Students log in and use the sandbox. They need to create a new file and import the codex library, plus sleep and randrange.

PART 1: Drawing the clovers

You can either:

- Give students the starter code with clovers already programmed (slides 5-7) –OR–
- Skip to slide 8 and have students work through slides to draw their own clovers.

💻 **Slides 5-7: Step 1 & 2**

For this option, give students the starter code to copy and paste into their own file. They run the code and can modify the clovers as needed.

💻 **Slides 8-16: Step 1 & 2**

For this option, students work through the slides step-by-step to design their own 3-leaf and 4-leaf clover. Example code is given. They will only construct one clover of each type.

PART 2: Coding the game

💻 **Slides 18-19: Step 3**

The slides introduce probability and the basic algorithm for the game.

💻 **Slides 20-23: Step 3**

Students define and call a function that uses a random number and probability. It returns a Boolean value that is displayed on the screen only for testing purposes.

💻 **Slides 24-27: Step 4**

Students use the Boolean value in another function that will display either the 4-leaf or the 3-leaf clover. During testing they should find that the clovers just draw on top of each other.

💻 **Slides 28-29: Step 5**

The fix! Students clear the display at the beginning of each button press. This is a good place for testing different probabilities. Students can decide how easy or difficult their game is by selecting an appropriate probability.

💻 **Slides 30-33: Step 6**

Students define another function that will flash pixels when they "win" the game. Example code is given, but really students should be creative and design their own flashy win. They can even include sound!

💻 **Slides 34-36: Step 7**

Final step of basic game: add a counter. The number of button presses, or "spins" is counted and then displayed when the pixels are flashing.

You can stop here, or continue with the next part. You can also skip to Part 4 and do the extras without Part 3.

PART 3: Intermediate Level – adding more than 1 clover

💻 **Slides 38-42: Step 8**

Students are given step-by-step instructions on adding variables to the clovers so they can be moved and/or duplicated. The biggest problem in this step is missing code or adding it where it doesn't belong. The first number in each circle/rectangle needs to have x+ added to it. The second number in each circle/rectangle needs to have y+ added to it. No other changes. Also, don't forget to add the parameters for each function.

Make sure students test their code to see the clovers before moving on. Testing is on slide 42. They can press the button several times to see all the clovers in many positions to be sure.

💻 **Slides 43-45: Step 9**

Students call the functions a second time to display a second clover. This may require a little bit of math. They need to think about the size of the clover. The radius is 20, so each leaf has a diameter of 40. They can think about the math to determine where the next clover should be, or just trial and error. They can keep y at 0 for the second clover. A suggestion of the second x value is given (110).

Students also need to modify the condition for winning. There are two conditions now instead of one.

💻 **Slides 46-48: Step 10**

Students follow the same steps for displaying a third clover. They also modify the condition for winning.

At this point, students have a working game with three clovers. You can stop here, or if time permits to to Part 4.

PART 4: Intermediate Level – extras: introduction and ending

💻 **Slides 50-52: Step 11**

Students make the game more user-friendly by adding an introduction with instructions. A sample intro is given, but students can use their creativity to come up with their own.

💻 **Slides 53-60: Step 12**

Students add an ending message to the game. They also use a variable to keep track of the lowest number of spins needed to win. This value is displayed as part of the ending message.

Finished! Enjoy the game!

💻 **Slide 62: Extensions**

Several ideas are given for students who want to challenge themselves and apply their learning. Instructions are not given for the extensions.

Wrap-up / Optional (5 minutes)

💻 The project doesn't require a wrap-up. You can use an extra-curricular activity from the list above, or have students reflect on the project and something they learned.