LIMITLESS SPACE INSTITUTE

Programming a PIP Party: A Scratch Design Challenge Mission Brief

Subject: Computational Thinking/Coding

Grade: Kindergarten through 8th Grade

Time: 90 minutes

Learning Objectives:

- Apply programming concepts and problem-solving skills to design and create a dance party featuring PIP in Scratch or Scratch, Jr.
- Foster creativity and critical thinking through the development of program mechanics, obstacles, backgrounds, and sprites.
- Encourage collaboration and peer feedback during the playtesting and iteration process.

Standards and Competencies:

NGSS Elementary (K-5)

- 3-5-ETS1-1. Define a simple design problem reflecting a need or a want that includes specified criteria for success and constraints on materials, time, or cost.
- 3-5-ETS1-2. Generate and compare multiple possible solutions to a problem based on how well each is likely to meet the criteria and constraints of the problem.
- 3-5-ETS1-3. Plan and carry out fair tests in which variables are controlled and failure points are considered to identify aspects of a model or prototype that can be improved.

NGSS Middle School (6-8)

- MS-ESS1-3. Analyze and interpret data to determine scale properties of objects in the solar system.
- MS-ETS1-1. Define the criteria and constraints of a design problem with sufficient precision to ensure a successful solution, taking into account relevant scientific principles and potential impacts on people and the natural environment that may limit possible solutions.
- MS-ETS1-2. Evaluate competing design solutions using a systematic process to determine how well they meet the criteria and constraints of the problem.
- MS-ETS1-3. Analyze data from tests to determine similarities and differences among several design solutions to identify the best characteristics of each that can be combined into a new solution to better meet the criteria for success.
- MS-ETS1-4. Develop a model to generate data for iterative testing and modification of a proposed object, tool, or process such that an optimal design can be achieved.

Background Knowledge:

- Read the book “PIP Goes Incredibly Fast”
- Review the Engineering Design Process, as needed, using your preferred steps.
- Review Scratch Basics
  - Recap key Scratch concepts, including sprites, backdrops, and basic programming blocks (e.g., motion, sensing, conditionals).
  - If necessary, provide a brief demonstration or refresher on using Scratch's interface and block system.

Teacher Challenge Preparations:

- Familiarize yourself with Scratch:
  - Explore the Scratch website (www.scratch.mit.edu) and review the available resources (https://scratch.mit.edu/educators/) , tutorials, and project examples (https://resources.scratch.mit.edu/www/guides/en/EducatorGuidesAll.pdf)
  - Create a Scratch account if you don't have one, as it allows you to save and share projects.
  - Spend time experimenting with Scratch yourself to gain hands-on experience and become comfortable with the interface and basic programming concepts.

Materials:

- “PIP Goes Incredibly Fast” Storybook
- Computers or laptops with Scratch installed (Scratch can be accessed online at www.scratch.mit.edu)
- Projector or screen for demonstration purposes (optional)
- Scratch tutorial handouts (optional)
- Planning paper (optional)

Procedure:

Introduction (10 minutes)

- Welcome the students and provide an overview of the design challenge.
- Explain the objective of creating a party simulation using Scratch.
- Emphasize the importance of applying programming concepts and problem-solving skills.

Design the Program (20 minutes)

- Discuss the components of the party, such as the space backdrop, sprite (must include P.I.P), movements, and obstacles.
- Encourage students to brainstorm ideas for the party simulation design, player controls, and win conditions, if needed.
- Remind students to consider how to handle collisions, movement restrictions, and other project mechanics.

Hands-on Coding (30 minutes)

- Allow students to work individually or in pairs to start implementing their program design.
- Provide support and guidance as needed, answering questions and offering suggestions.
- Remind students to use programming concepts like conditionals, loops, and event blocks to create program functionality.

Playtesting and Peer Feedback (10 minutes)

- Allocate time for students to playtest their programs.
- Encourage them to exchange programs and provide constructive feedback to their peers.
- Facilitate a class discussion on the strengths and areas for improvement in each program.

Iteration and Refinement (10 minutes)

- Encourage students to incorporate feedback and make necessary adjustments to their programs.
- Discuss the importance of iterating and refining designs based on feedback and testing.
- Remind students to test their updated programs to ensure the changes have the desired effect.

Final Presentations (15 minutes)

- Provide an opportunity for students to showcase their completed programs to the class.
- Each student or pair should explain their program mechanics, controls, and sprites.
- Encourage the audience to ask questions and provide positive feedback.

Assessment:

Reflection and Wrap-up (5 minutes)

- Lead a short discussion on the design process, challenges faced, and lessons learned.
- Ask students to reflect on how they applied programming concepts and problem-solving skills during the challenge.
- Have students identify a “Glow” (positive comment) and “Grow” (constructive feedback) for themselves or peers.
- Wrap up the lesson by acknowledging students' efforts and encouraging them to continue exploring Scratch and simulation design.

Additional Resources:

Extension Activities

- Challenge students to create additional levels or increase the complexity of their programs.
- Introduce advanced programming concepts, such as custom blocks, variables, or advanced sensing blocks.
- Explore the possibility of incorporating sounds, animations, or power-ups into the programs.
- Encourage students to share their completed programs with others or publish them on the Scratch community platform.

**Note: The duration and complexity of the design challenge can be adjusted based on the students’ age, skill level, available materials, and time constraints.