# HYPE Game- Clear the Court!

HOPE FOR YOUTH THROUGH PROVIDING EDUCATION

**STEAM Connection:** Why is it important to give very clear instructions? Have you ever been given unclear instructions and ended up doing the wrong thing?
In this game, the players will have to debug the program by using a code to give clear instructions.

## Suggested Gameplay

Have fun in physical play game mode where you use a large grid or the ground such as:

- An outside painted chess board.
- Grids in your floor tiles at home.
- Make your own masking tape grids one the floor in your living room or den.
- Draw a chalk grid outside.
- Want to play without the space? Use a board from another game or the handout included!

Have 2 friends or family members join you! Determine who will have each role for the first round. Not sure how to pick? Try Paper, Scissors, Rock. If you are playing with 2 players instead of 3, either player can be the Developer.
**Role 1:** The Developer (who writes the program) - Your developer will setup the bug(s) before gameplay.
**Role 2:** The Tester (who instructs the Bot and looks for bugs) - Your tester will create the code to remove bugs.
**Role 3:** The Bot (who runs the program) - Your bot will follow the code to remove bugs.

## Objective:

- To use code to navigate the grid and "debug" the program. Clearing the court of all the balls means the programmer has successfully debugged the program!

## To Win:

- A Tester must reach 5 points to win!

## Resources Needed:

### Physical Play

- Bugs: Basketball Balls (or any small object)
- Program Grid: Masking tape, chalk floor tiles or outdoor board (5x9 suggested)
- Clipboards
- Handheld whiteboards/Paper
- Whiteboard marker/Pens

### Handout Play

- Your Court: Program Grid Handout
- Your Game Pieces Handout
- Scissors

- Handheld whiteboards/Paper
- Whiteboard marker/Pens

## Vocabulary

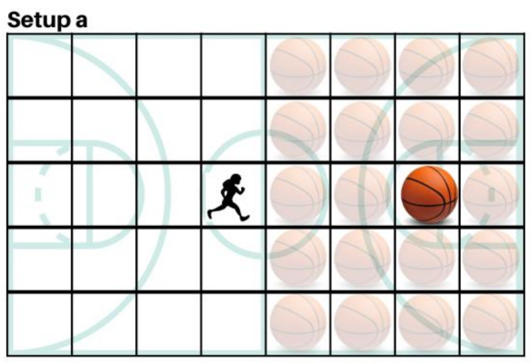
1. **Developer**- an individual that builds and create software and applications
2. **Programmer**- make, test and troubleshoot the coding languages within a software application to make sure it runs successfully
3. **Bot**- a software application programmed to execute specific tasks as part of another computer program or to simulate human activity
4. **Debugging**- the process of finding and fixing errors or bugs in the source code of any software
5. **Programming language**- a language that allows people to write specific commands to be executed on a computer

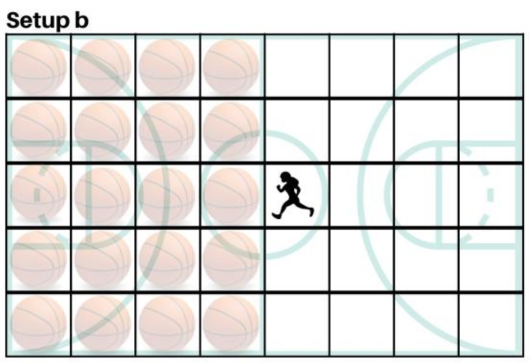
## Pre-Game Reflection

1. Have you ever used a computer program or app that had a glitch or bug? What did that look like? How did you know there was a problem or an error? What did you (or an adult) have to do to clear the problem or fix the glitch?
2. Sometimes programmers create updates to add new features or functions to a program or app. Updates can also debug a program. What was the last program or app you used that needed an update?
3. What happens if a program is not debugged or you don't update a program?
4. Look up the specs of your favorite digital game or app (the app store or any device is a great spot to look). Review the information to find update history. When was the program created? How many updates have there been since the program's release?

## Connectivity Activity

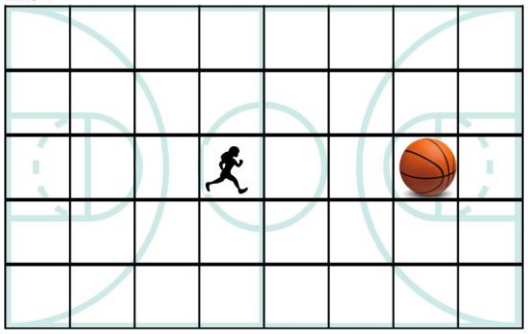
1. Setup: The Developer(s) will be tasked to place your bug (ball) anywhere on either side of the court. The starting position for the bot (player) should be at the midcourt line of the grid (the court) on the side *opposite* of the ball (see setup examples below). [If you are playing using the Handouts, cut the game pieces out to play accordingly.]





2. The tester should create a code, only using the programming language of arrows, to help the bot navigate to the bugs to remove them. The tester has 30 seconds to write a code down.

Ex. 1



Your code of arrows for Ex. 1 should look something like this:

➡ ➡ ➡

There are multiple codes that will navigate the bot to the bug on the program grid. Below you can see alternative codes that would also debug this program. Use your finger to follow your player from the starting position to the ball and test out these alternative codes!

**Other Possible Codes:**

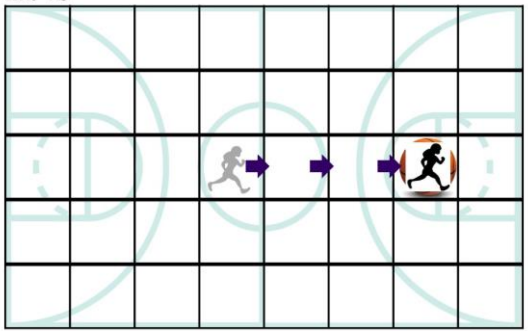
⬆ ➡ ➡ ➡ ⬇

⬇ ➡ ➡ ➡ ⬆

⬅ ⬆ ➡ ➡ ➡ ➡ ⬇

3. Standing at the midcourt starting position, the player that is assigned to be the bot should be given the written code. Only using the instructions given, the bot should follow the arrows of the code to navigate to the bug. If the bot end in the same square as the bug, pick it up to clear the court! See below how the first code navigates to the bug in 3 moves.

Ex. 1b

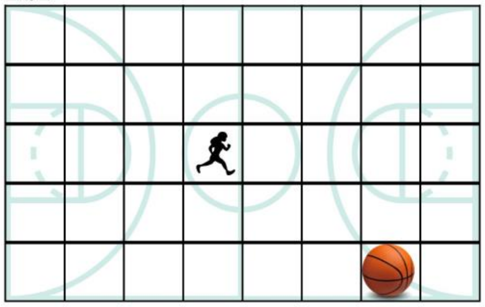


*The tester cannot change the code once the bot starts to move!

4. Did you clear the bug? Each correct code is a point! If the bot doesn't end in the same square as the bug, the tester does not get the point.
5. Now, take turns! Rotate roles. The next tester must come up with a new code in 30 seconds to clear the court. If a code is repeated, the tester's turn is over. The bug and bot should start in the same place until a tester reaches 5.

## More Gameplay Examples

**Ex. 2**

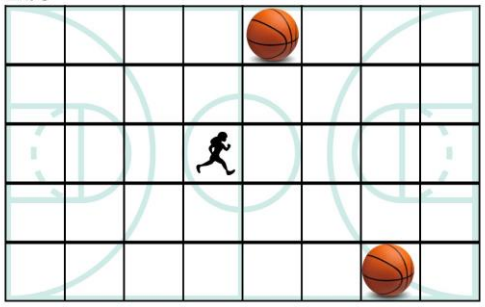


**Possible Codes:**

→ → → ↓ ↓

↓ ↓ → → →

**Ex. 3**

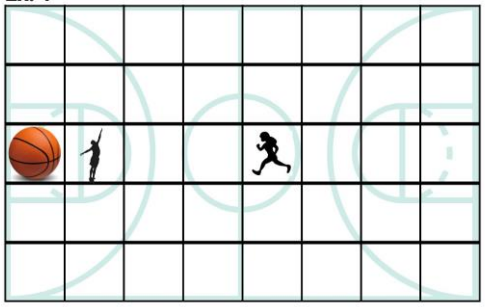


**Possible Codes:**

↑ ↑ → → → ↓ ↓ ↓ ↓

→ → → ↓ ↓ ← ← ↑ ↑ ↑ ↑

**Ex. 4**

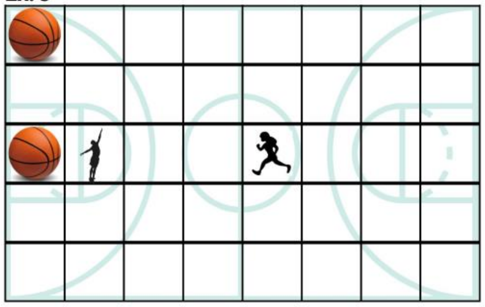


**Possible Codes:**

← ← ↓ ← ← ↑

← ← ↑ ← ← ↓

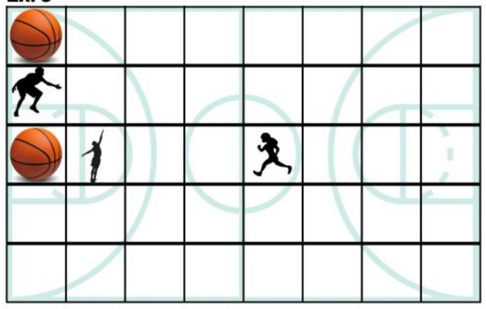
**Ex. 5**



**Possible Code:**

← ← ↓ ← ← ↑ ↑ ↑

Ex. 6



**Possible Code:**

⬅⬅⬇⬅⬅⬆⬇➡➡⬆⬆⬆⬅⬅

## Level Up!

**Is this *too* easy? Try any of these ideas to make the game more challenging.**

Add obstacles.

- Opponents or obstacles represent stationary items that can go anywhere on your grid. Add opponents to the court for your bot to navigate around. The more obstacles you place, the more challenging the code needs to be.

Increase the number of bugs

- Place more than 1 bug. See if you can clear all of the bugs with one long code. Can you do it on your first try?

Playing solo? Time yourself.

- Using multiple bugs, a tester should time their first round of debugging. Keeping the bugs in the same locations, challenge yourself to create a code the clears the bugs out faster.

Who's faster?

- With 3+ players, designate the Developer. Setup your program grid with multiple bugs and/or obstacles. Have all other players start writing a debugging code simultaneously. The Developer should be watching to determine the order that they finish writing the code. Starting with the code that was written the fastest, have a bot test each code. Which successful code was the fastest one written?

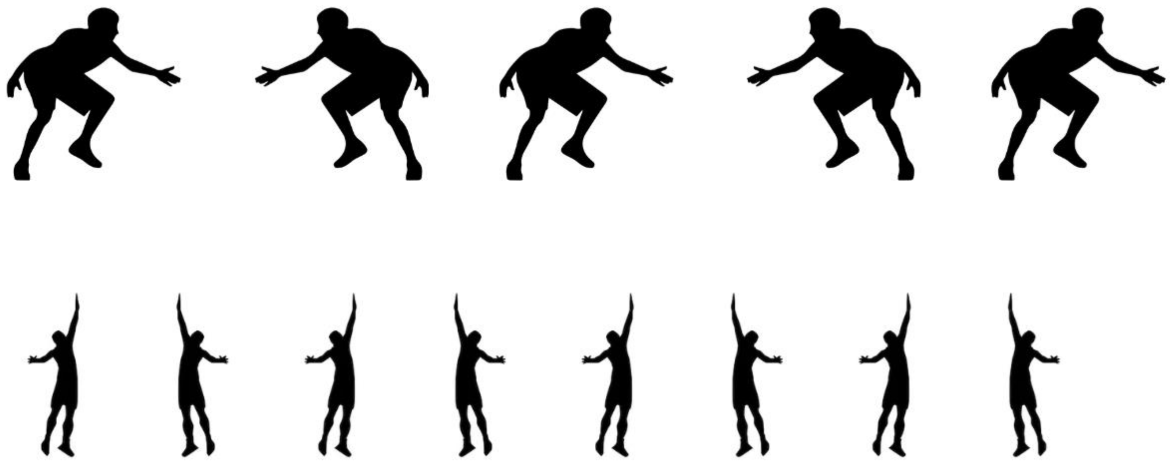
## Your Game Pieces

Playing on paper? Cut out these pieces for endless rounds of debugging!

The Player (The Bot)

The Ball (The Bug)

Opponents (Obstacles)



Intentionally Blank

## The Court: Your Program Grid