The Maze Game — Mini Project 1: Detect Which Keys are Pressed

In this first mini-project, you will learn how to use the **getKey** function to return the value of a pressed key. You will use the displayed values to determine the numeric value of all the keys on the keyboard. In order to create your maze game, you'll need to know the numeric value for all four arrow keys.

Objectives:

- Use getKey to return key press values
- Learn the number value of each key
- Learn how to insert lines of code

The MAZE Game Project Overview:

After completing a series of 7 mini-projects, you will have a maze game similar to the one on the right. Projects 1 and 2 will provide skills needed to code movement in the maze game. Projects 3-6 will create code you'll import using the recall method and use in your final project.

Mini-project Order:

1. **<u>Detect which keys are pressed</u>**
2. Use key presses to move a word stored in a string value
3. Draw objects using pixels
4. Move objects using keypresses and variables
5. Create a specific Maze
6. Randomize maze attributes
7. Create the final maze project.

Teacher Tip:
Students will ultimately import code from projects 3-6 to project 7. Make sure students save each project as they work through the mini-projects.

1. The getKey command returns the value of the key that is pressed.
   To find the getKey command, press prgm then select the I/O menu for input/output.

   The line: getKey→K
   Stores the value of the key into a variable named K.

   Write the following lines of code:

   ```
   :ClrHome
   :getKey→K
   :Disp K
   ```

   If you execute the program now, it will execute and display a 0 automatically. The program doesn't wait for you to press a key. If it doesn't detect a button push it displays 0.

```
PROGRAM:KEYPRESS
:ClrHome
:getKey→K
:Disp K
```

```
PROGRAM:KEYPRESS
:ClrHome
:0→K
:While K≠45
:getKey→K
:Disp K
:End
```

2. Insert the two lines between ClrHome and the getKey line.
   (alpha graph will give you an option to insert a blank line)
   :0 → K
   :While K ≠45

   At the end of your code add
   :End
   The while loop will continue to execute the code to get and display the values of keys pressed until the clear key is pressed.

   Run your program.

   What type of values are displayed when you press a key?
   What type of values are displayed when you don't press a key?

Teacher Tip:
Pressing a key results in a value from 11 to 105. If a button isn't pressed, 0 is displayed.
When debugging, make sure students use the ≠ sign.

```
PROGRAM:KEYPRESS
:ClrHome
:0→K
:While K≠45
:getKey→K
:If K≠0
:Then
:Disp K
:End
:End
```

3. Insert the following If statement before the display line.
   :If K ≠ 0
   :Then
   :Disp K
   :End

   Before running your modified program, predict how the new lines of code will change the program.

Teacher Tip: Only the numbers from the buttons pushed will show. The 0's will no longer display on the screen when a button isn't pushed.

4. Use your program to explore the keypad.

What are the values of the arrow keys?

What is the value of the enter key?

What value is the clear key?

Use the blank template on the right to write down the numeric value of each key. Do you notice the pattern?

Teacher Tip:
The clear key and arrow keys will be used multiple projects. Students might want to save a paper copy for reference. Students could also rerun this program later to determine key values when needed in projects. Notice the row number is the first number on the key, the column number is the second number on the key. The enter key is in the 10th row in the 5th column, so its number is 105. The math key is in the 4th row in the 1st column so it is number 41.

Project Code:

```
NORMAL FLOAT AUTO REAL RADIAN MP
EDIT MENU: [alpha] [f5]
PROGRAM:KEYPRESS
:ClrHome
:0→K
:While K≠45
:getKey→K
:If K≠0
:Then
:Disp K
:End
:End
```