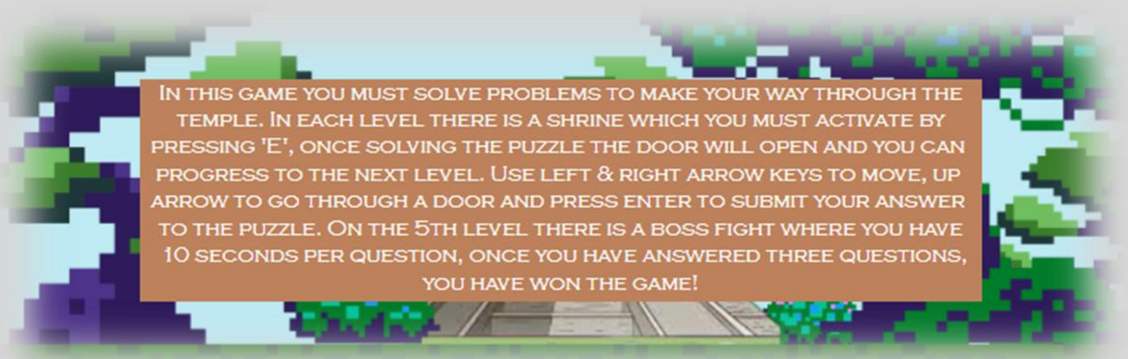
ES2D7 INDIVIDUAL PROJECT

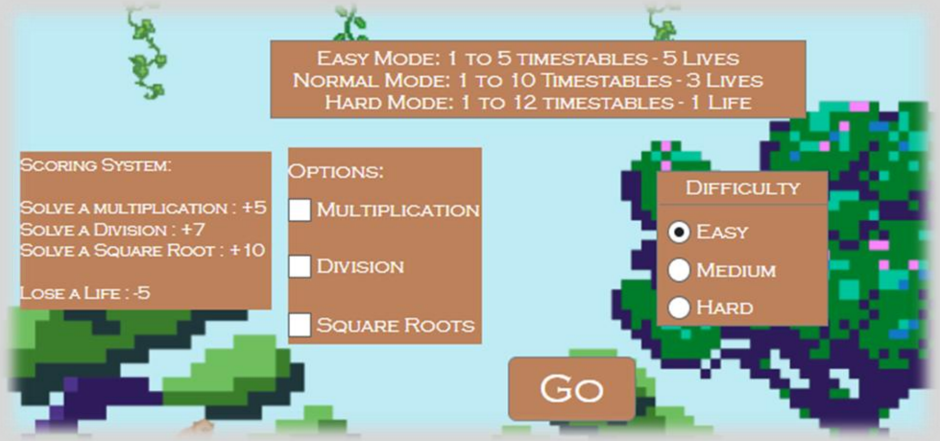
Ethan Stubley

Overview

- The aim of this game was to provide a fun and entertaining way to help primary school children (Ages 7-11) learn and practice times tables. It has different options when you start to select a difficulty level - the difficulty of the timetables can be altered as well as how many lives you start with and the option to test, multiplication, division and square roots. This allows the game to appeal to wider age range.


IN THIS GAME YOU MUST SOLVE PROBLEMS TO MAKE YOUR WAY THROUGH THE TEMPLE. IN EACH LEVEL THERE IS A SHRINE WHICH YOU MUST ACTIVATE BY PRESSING 'E', ONCE SOLVING THE PUZZLE THE DOOR WILL OPEN AND YOU CAN PROGRESS TO THE NEXT LEVEL. USE LEFT & RIGHT ARROW KEYS TO MOVE, UP ARROW TO GO THROUGH A DOOR AND PRESS ENTER TO SUBMIT YOUR ANSWER TO THE PUZZLE. ON THE 5TH LEVEL THERE IS A BOSS FIGHT WHERE YOU HAVE 10 SECONDS PER QUESTION, ONCE YOU HAVE ANSWERED THREE QUESTIONS, YOU HAVE WON THE GAME!

- The Title screen features instructions on how to play the game, which is especially important for younger users who may struggle to understand the game at first.


EASY MODE: 1 TO 5 TIMESTABLES - 5 LIVES
NORMAL MODE: 1 TO 10 TIMESTABLES - 3 LIVES
HARD MODE: 1 TO 12 TIMESTABLES - 1 LIFE

SCORING SYSTEM:

SOLVE A MULTIPLICATION : +5
SOLVE A DIVISION : +7
SOLVE A SQUARE ROOT : +10

LOSE A LIFE : -5

OPTIONS:
- ☐ MULTIPLICATION
- ☐ DIVISION
- ☐ SQUARE ROOTS

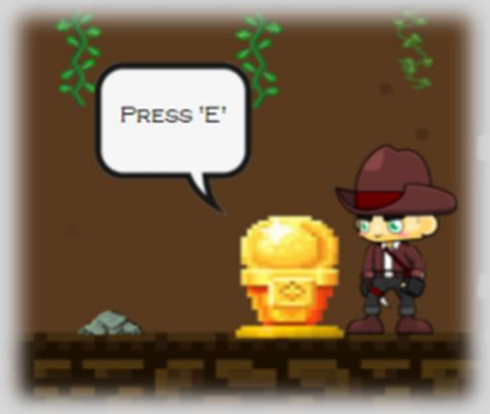
DIFFICULTY
- ● EASY
- ○ MEDIUM
- ○ HARD

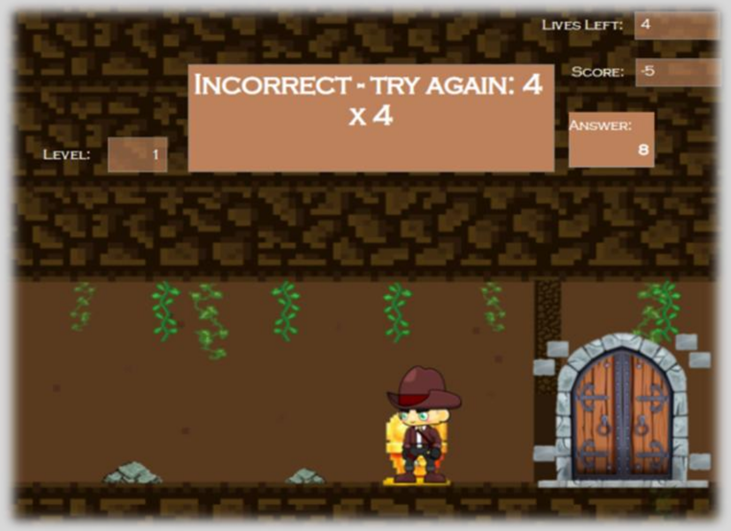
The Game needs to be:

- Easy to understand/play
- Educational
- Entertaining
- Robust/bug-free

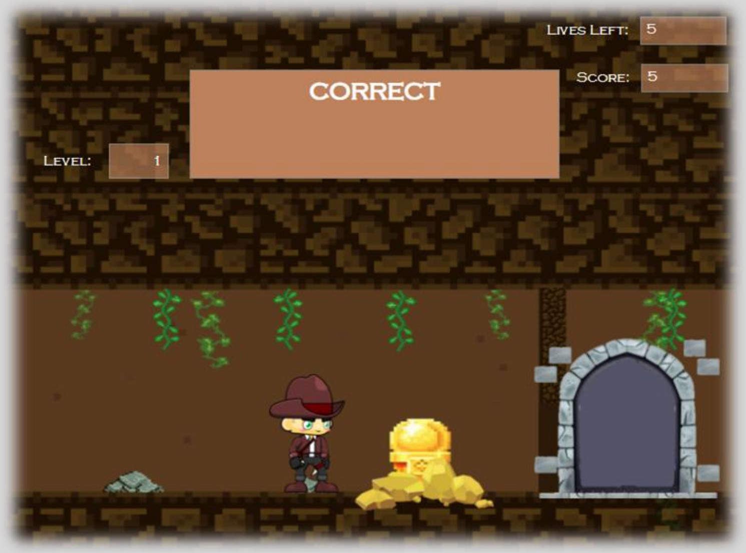
Overview

- The game is a 2D side-scroller style game in which the user must use the left and right arrow keys to navigate the temple. Each level has a shrine/puzzle which can be activated by pressing 'E' when standing near it.
- Upon doing this, a maths question will appear and the user has to input the correct answer to progress to the next level and their score will be increased accordingly, depending on the type of question. If they get the question wrong, a message will be displayed and they will be asked to try again – however they will lose a life.



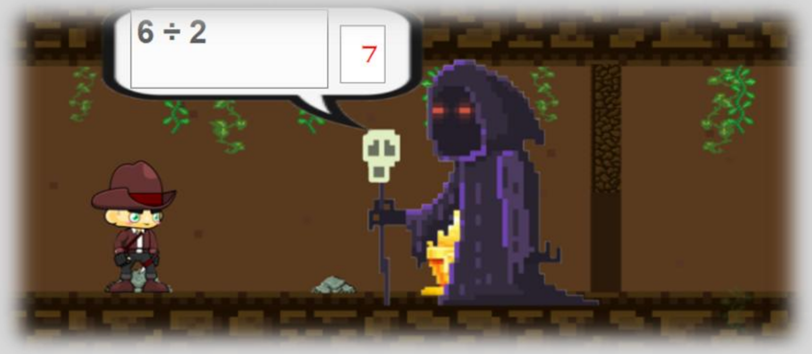


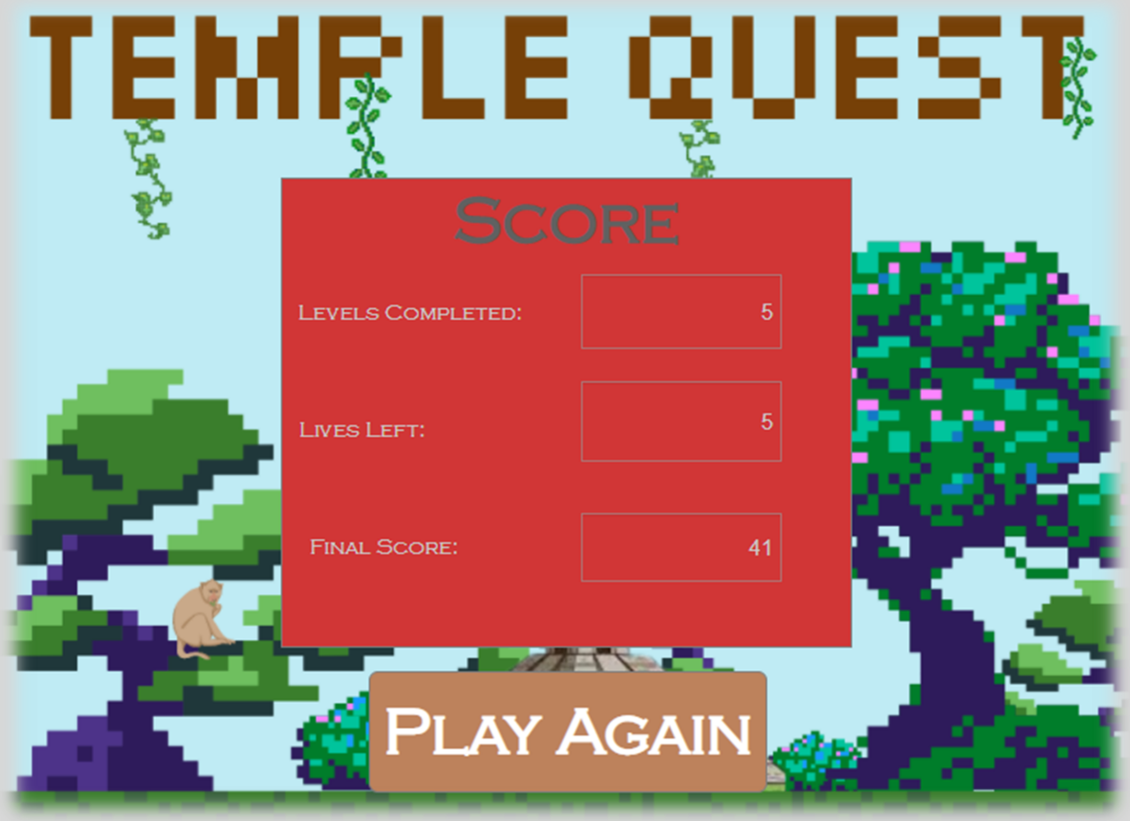
- If the user gives the correct answer then the shrine will 'break' and the door will open allowing them to progress to the next level



Overview

- Once the user reaches the fifth and final level of the game they will be confronted with a boss fight.
- In the boss fight the user will have to answer three correct questions to defeat the boss, they must answer each question within ten seconds otherwise they will lose a life.
- Once the user has defeated the boss – or if they run out lives they will be presented with a scoreboard which gives them a score based on the question types answered and how many they got correct.





Score

Levels Completed: 5

Lives Left: 5

Final Score: 41

- There is also a Play Again button which allows the user to play the game again

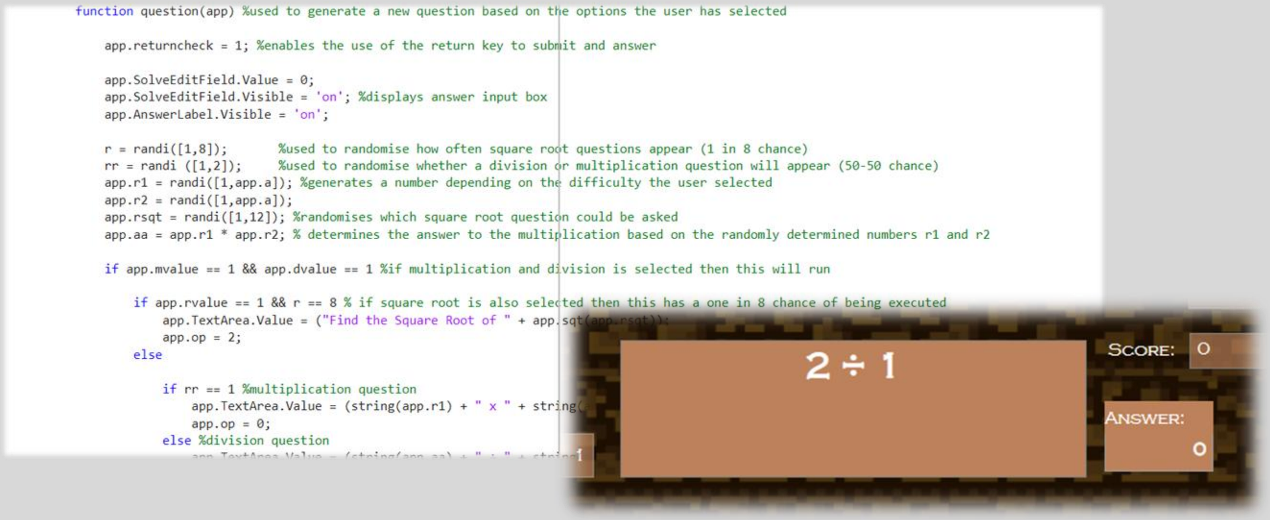
Development & Design Process

Function/Feature	Priority (1-5)	Time estimation (mins)
Generate Question	5	60
Checking Input	5	30
Menu and Instructions	5	50
Character Animation Movement and Animation	4	90
Level Design	4	20
Moving through the Door to the next Level	3	30
Scoring System	5	30
Lives	5	20
Boss Fight	2	150

- I came up with a type of product backlog to help breakdown the features I wanted to include in the game, how long they would take to implement and how important they were.

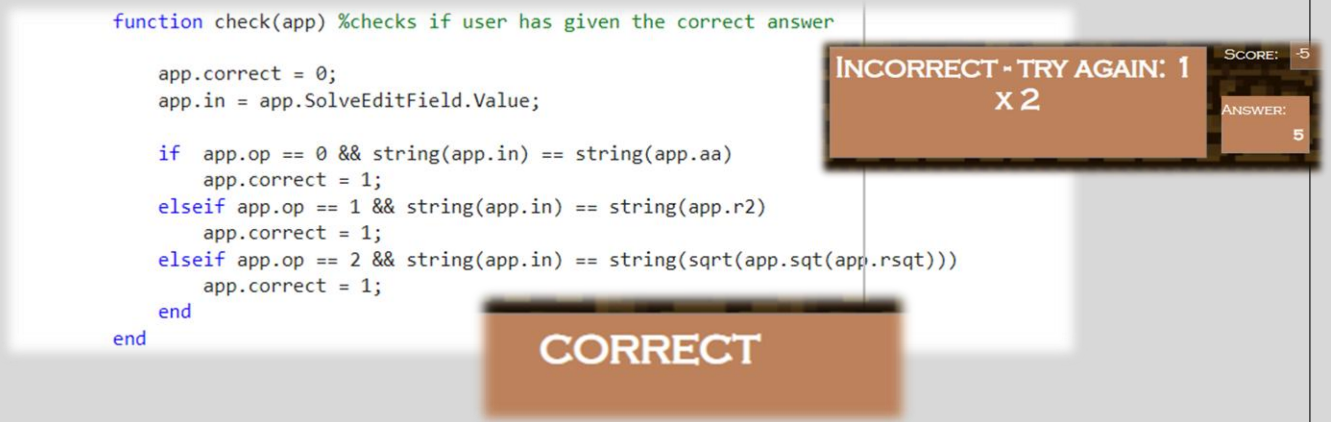
Development & Design Process

- To start, the first part of the game to be designed and coded was the question generator which was a function that would take into account selected options from the user, this was important to develop first as it would provide the foundation to build the rest of the game around.



```
function question(app) %used to generate a new question based on the options the user has selected

    app.returncheck = 1; %enables the use of the return key to submit and answer

    app.SolveEditField.Value = 0;
    app.SolveEditField.Visible = 'on'; %displays answer input box
    app.AnswerLabel.Visible = 'on';

    r = randi([1,8]);       %used to randomise how often square root questions appear (1 in 8 chance)
    rr = randi ([1,2]);     %used to randomise whether a division or multiplication question will appear (50-50 chance)
    app.r1 = randi([1,app.a]); %generates a number depending on the difficulty the user selected
    app.r2 = randi([1,app.a]);
    app.rsqt = randi([1,12]); %randomises which square root question could be asked
    app.aa = app.r1 * app.r2; % determines the answer to the multiplication based on the randomly determined numbers r1 and r2

    if app.mvalue == 1 && app.dvalue == 1 %if multiplication and division is selected then this will run

        if app.rvalue == 1 && r == 8 % if square root is also selected then this has a one in 8 chance of being executed
            app.TextArea.Value = ("Find the Square Root of " + app.sqt(app.rsqt));
            app.op = 2;
        else

            if rr == 1 %multiplication question
                app.TextArea.Value = (string(app.r1) + " x " + string(
                app.op = 0;
            else %division question
```

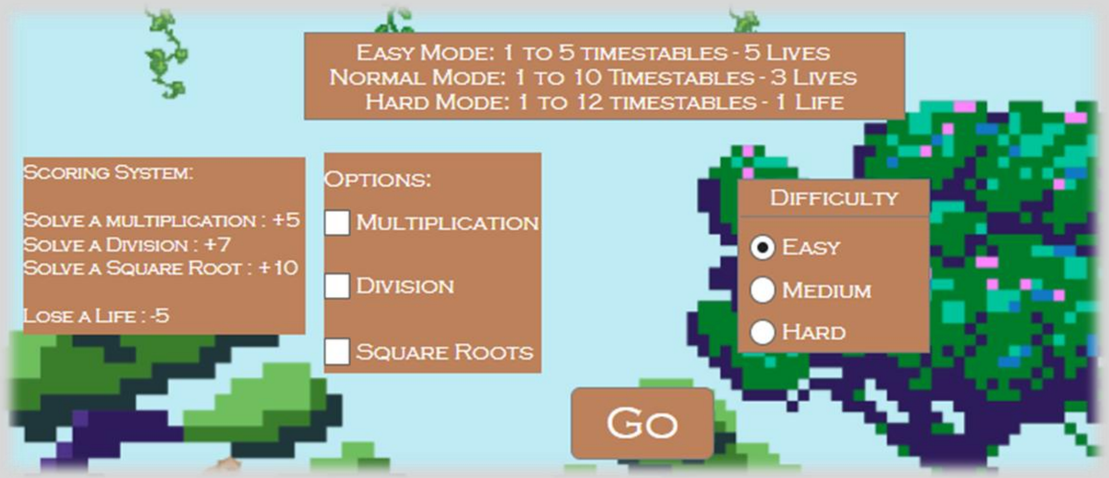
- The next part I worked on was the part of the program that could verify whether the user had given a correct answer and display an appropriate message in response



```
function check(app) %checks if user has given the correct answer

    app.correct = 0;
    app.in = app.SolveEditField.Value;

    if  app.op == 0 && string(app.in) == string(app.aa)
        app.correct = 1;
    elseif app.op == 1 && string(app.in) == string(app.r2)
        app.correct = 1;
    elseif app.op == 2 && string(app.in) == string(sqrt(app.sqt(app.rsqt)))
        app.correct = 1;
    end
end
```

Development & Design Process

- After creating the core of the game I implemented a menu at the beginning of the game and started experimenting with different ways to create a character that could be moved and animated.



Easy Mode: 1 to 5 timestables - 5 Lives
Normal Mode: 1 to 10 Timestables - 3 Lives
Hard Mode: 1 to 12 timestables - 1 Life

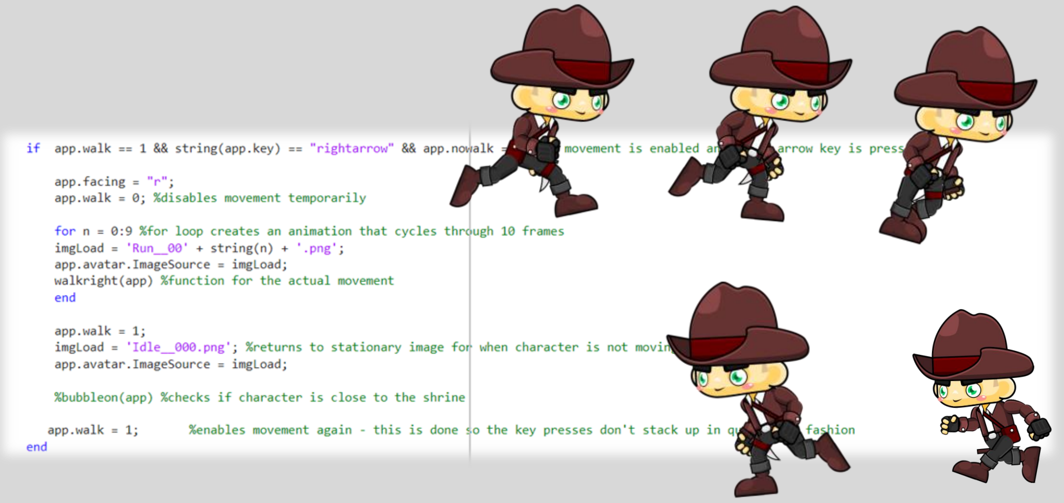
Scoring System:

Solve a multiplication : +5
Solve a Division : +7
Solve a Square Root : +10

Lose a Life : -5

Options:
☐ Multiplication
☐ Division
☐ Square Roots

Difficulty
[✓ Easy]
Medium
Hard

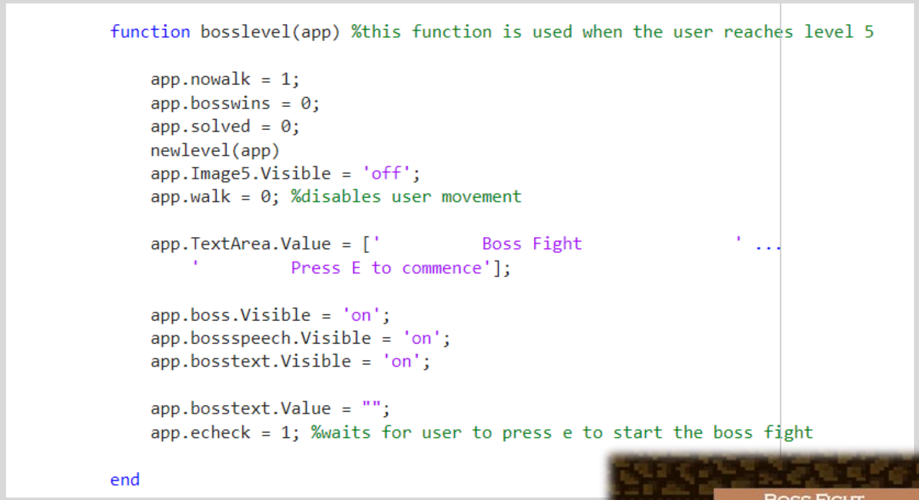


```
if  app.walk == 1 && string(app.key) == "rightarrow" && app.nowalk =        movement is enabled an        arrow key is press

    app.facing = "r";
    app.walk = 0; %disables movement temporarily

    for n = 0:9 %for loop creates an animation that cycles through 10 frames
    imgLoad = 'Run__00' + string(n) + '.png';
    app.avatar.ImageSource = imgLoad;
    walkright(app) %function for the actual movement
    end

    app.walk = 1;
    imgLoad = 'Idle__000.png'; %returns to stationary image for when character is not movin
    app.avatar.ImageSource = imgLoad;

    %bubbleon(app) %checks if character is close to the shrine

   app.walk = 1;       %enables movement again - this is done so the key presses don't stack up in qu        fashion
end
```

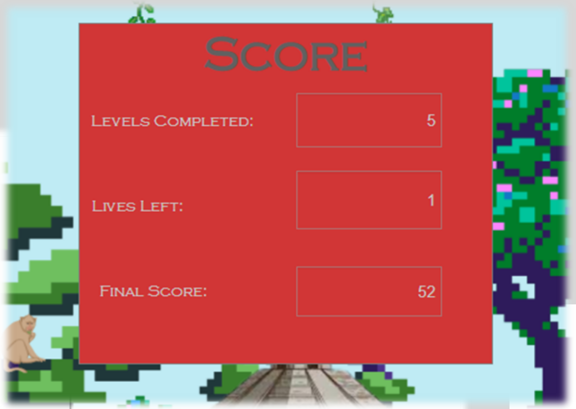
- I found a set of animation frames online which I could change quickly to give the effect of the character moving
- At this stage I also created some backgrounds for the menu and levels

Development & Design Process

- At this point the game was playable but I wanted to implement some more features so added a proper score section at the end of the game and implemented a boss level on the fifth and final level, making use of a lot of similar functions as the normal levels but including a timer so the questions had a time limit.



```
function bosslevel(app) %this function is used when the user reaches level 5

    app.nowalk = 1;
    app.bosswins = 0;
    app.solved = 0;
    newlevel(app)
    app.Image5.Visible = 'off';
    app.walk = 0; %disables user movement

    app.TextArea.Value = ['          Boss Fight                ' ...
        '          Press E to commence'];

    app.boss.Visible = 'on';
    app.bossspeech.Visible = 'on';
    app.bosstext.Visible = 'on';

    app.bosstext.Value = "";
    app.echeck = 1; %waits for user to press e to start the boss fight

end
```

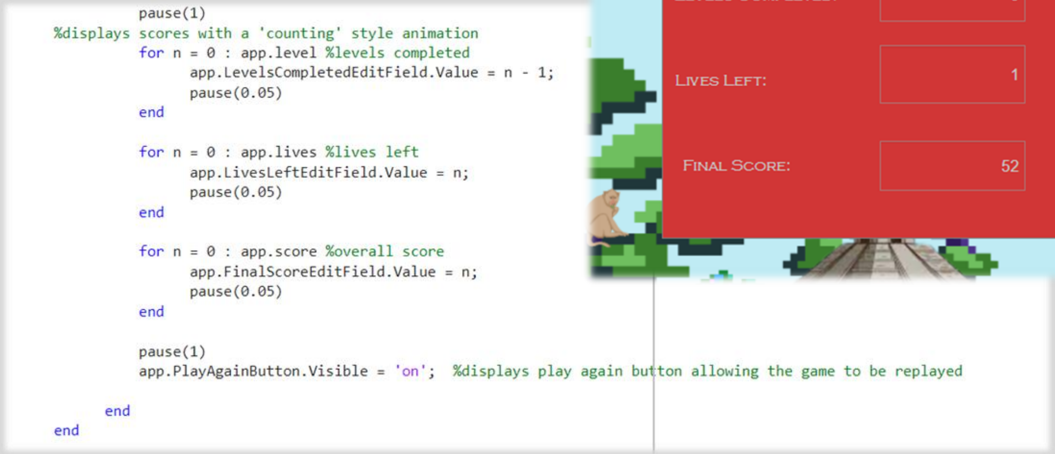


SCORE

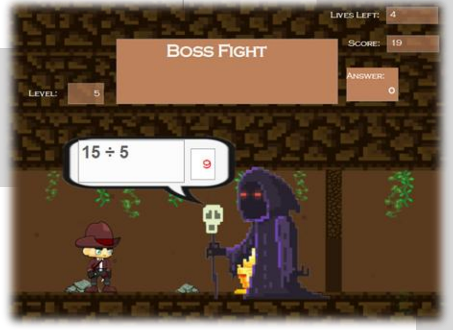
LEVELS COMPLETED: 5

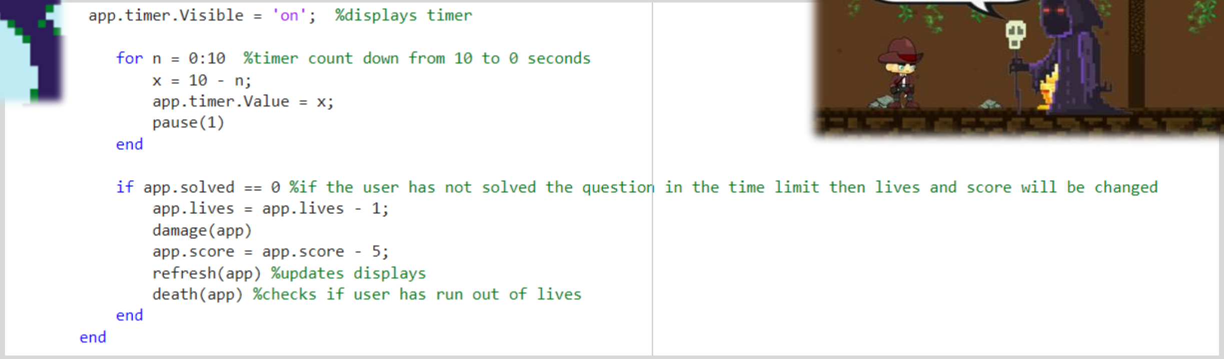
LIVES LEFT: 1

FINAL SCORE: 52



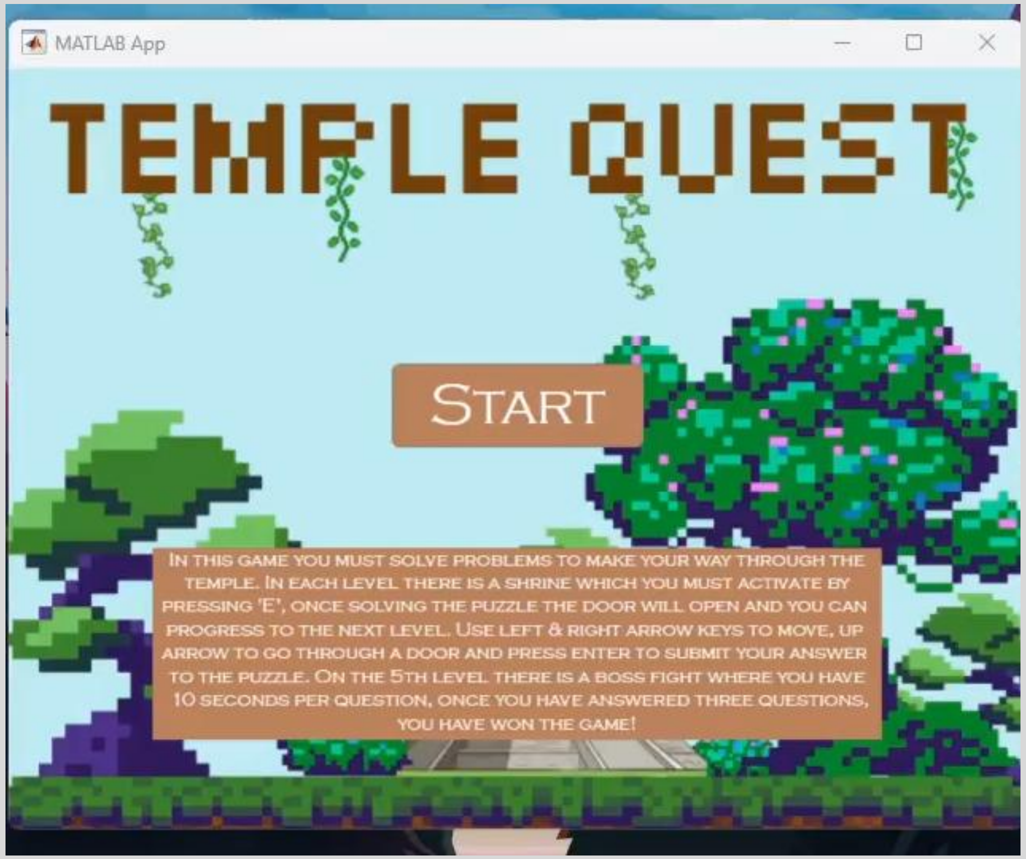
```
        pause(1)
%displays scores with a 'counting' style animation
        for n = 0 : app.level %levels completed
              app.LevelsCompletedEditField.Value = n - 1;
              pause(0.05)
        end

        for n = 0 : app.lives %lives left
              app.LivesLeftEditField.Value = n;
              pause(0.05)
        end

        for n = 0 : app.score %overall score
              app.FinalScoreEditField.Value = n;
              pause(0.05)
        end

        pause(1)
        app.PlayAgainButton.Visible = 'on';  %displays play again button allowing the game to be replayed

    end
end
```





```
app.timer.Visible = 'on';  %displays timer

    for n = 0:10  %timer count down from 10 to 0 seconds
        x = 10 - n;
        app.timer.Value = x;
        pause(1)
    end

    if app.solved == 0 %if the user has not solved the question in the time limit then lives and score will be changed
        app.lives = app.lives - 1;
        damage(app)
        app.score = app.score - 5;
        refresh(app) %updates displays
        death(app) %checks if user has run out of lives
    end
end
```

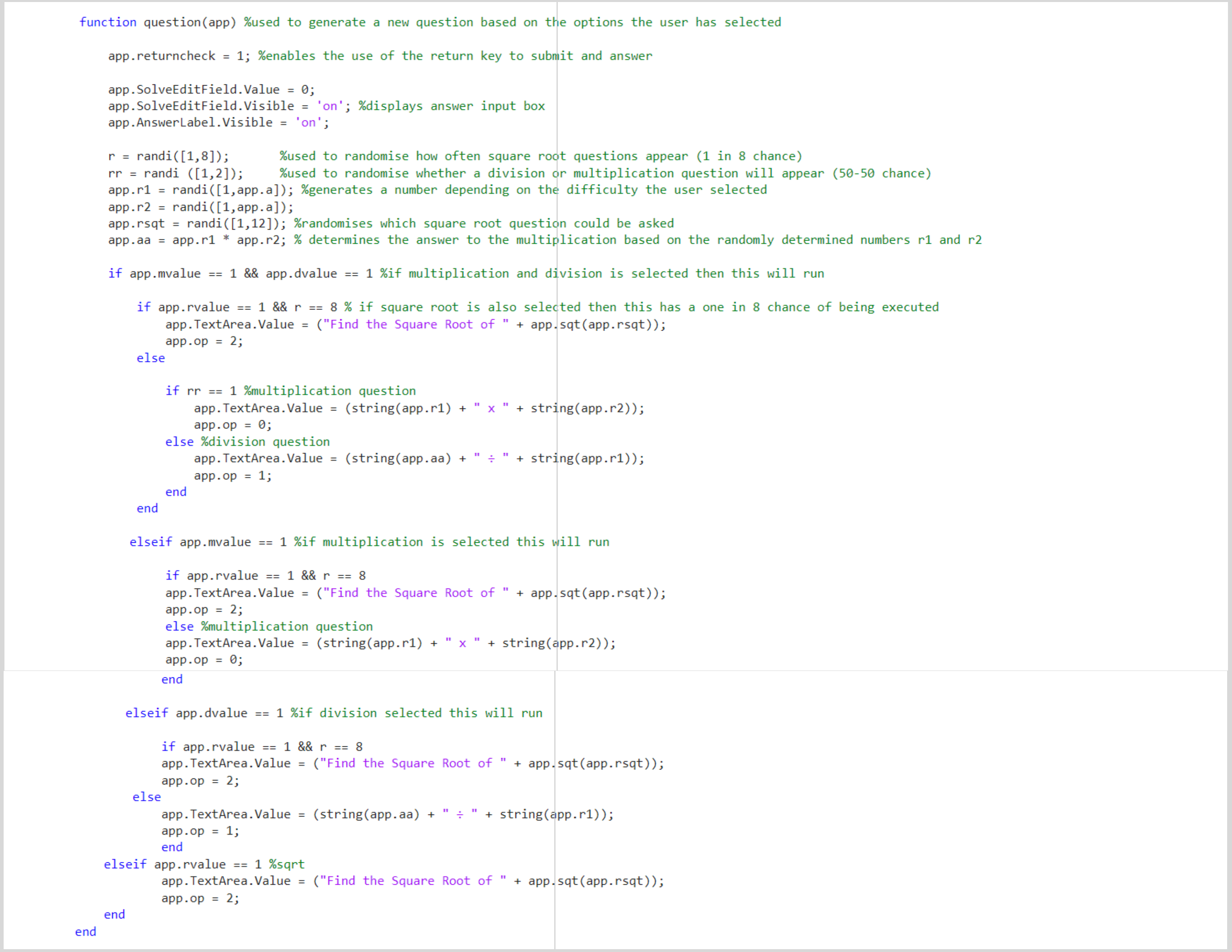
Demonstration



In this game you must solve problems to make your way through the temple. In each level there is a shrine which you must activate by pressing 'E', once solving the puzzle the door will open and you can progress to the next level. Use left & right arrow keys to move, up arrow to go through a door and press enter to submit your answer to the puzzle. On the 5th level there is a boss fight where you have 10 seconds per question, once you have answered three questions, you have won the game!

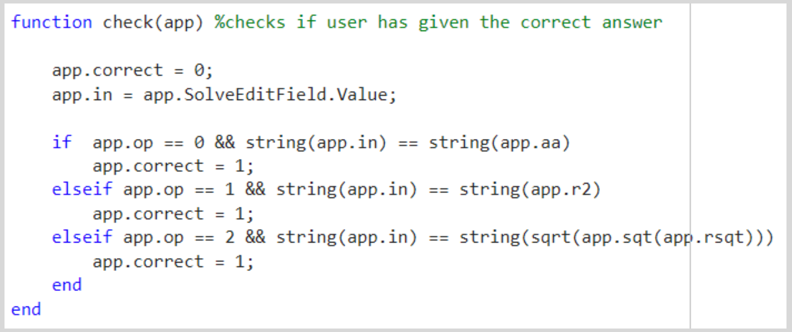
Explanation of Code

- Question Function



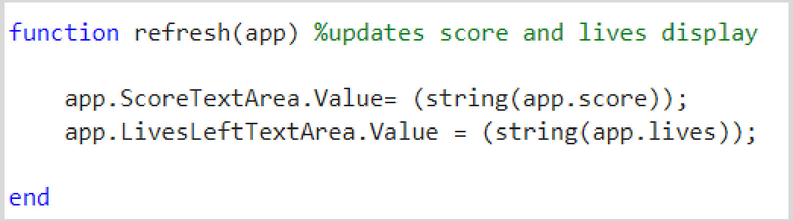
```
function question(app) %used to generate a new question based on the options the user has selected

    app.returncheck = 1; %enables the use of the return key to submit and answer

    app.SolveEditField.Value = 0;
    app.SolveEditField.Visible = 'on'; %displays answer input box
    app.AnswerLabel.Visible = 'on';

    r = randi([1,8]);       %used to randomise how often square root questions appear (1 in 8 chance)
    rr = randi ([1,2]);     %used to randomise whether a division or multiplication question will appear (50-50 chance)
    app.r1 = randi([1,app.a]); %generates a number depending on the difficulty the user selected
    app.r2 = randi([1,app.a]);
    app.rsqt = randi([1,12]); %randomises which square root question could be asked
    app.aa = app.r1 * app.r2; % determines the answer to the multiplication based on the randomly determined numbers r1 and r2

    if app.mvalue == 1 && app.dvalue == 1 %if multiplication and division is selected then this will run

        if app.rvalue == 1 && r == 8 % if square root is also selected then this has a one in 8 chance of being executed
            app.TextArea.Value = ("Find the Square Root of " + app.sqt(app.rsqt));
            app.op = 2;
        else

            if rr == 1 %multiplication question
                app.TextArea.Value = (string(app.r1) + " x " + string(app.r2));
                app.op = 0;
            else %division question
                app.TextArea.Value = (string(app.aa) + " ÷ " + string(app.r1));
                app.op = 1;
            end
        end

       elseif app.mvalue == 1 %if multiplication is selected this will run

            if app.rvalue == 1 && r == 8
            app.TextArea.Value = ("Find the Square Root of " + app.sqt(app.rsqt));
            app.op = 2;
            else %multiplication question
            app.TextArea.Value = (string(app.r1) + " x " + string(app.r2));
            app.op = 0;
           end

      elseif app.dvalue == 1 %if division selected this will run

           if app.rvalue == 1 && r == 8
           app.TextArea.Value = ("Find the Square Root of " + app.sqt(app.rsqt));
           app.op = 2;
       else
           app.TextArea.Value = (string(app.aa) + " ÷ " + string(app.r1));
           app.op = 1;
           end
   elseif app.rvalue == 1 %sqrt
           app.TextArea.Value = ("Find the Square Root of " + app.sqt(app.rsqt));
           app.op = 2;
   end
end
```

- Check Function



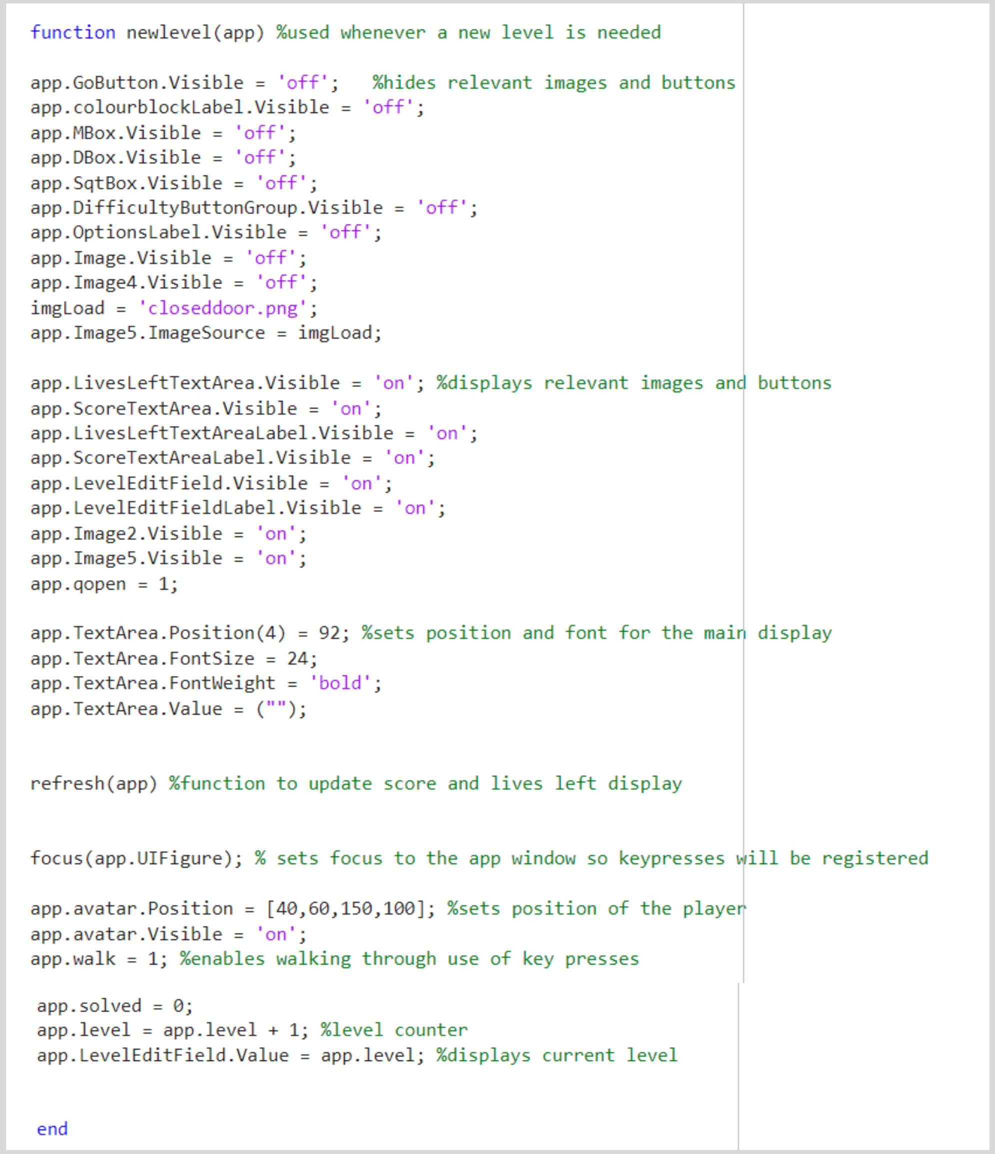
```
function check(app) %checks if user has given the correct answer

    app.correct = 0;
    app.in = app.SolveEditField.Value;

    if  app.op == 0 && string(app.in) == string(app.aa)
        app.correct = 1;
    elseif app.op == 1 && string(app.in) == string(app.r2)
        app.correct = 1;
    elseif app.op == 2 && string(app.in) == string(sqrt(app.sqt(app.rsqt)))
        app.correct = 1;
    end
end
```

- Refresh Function



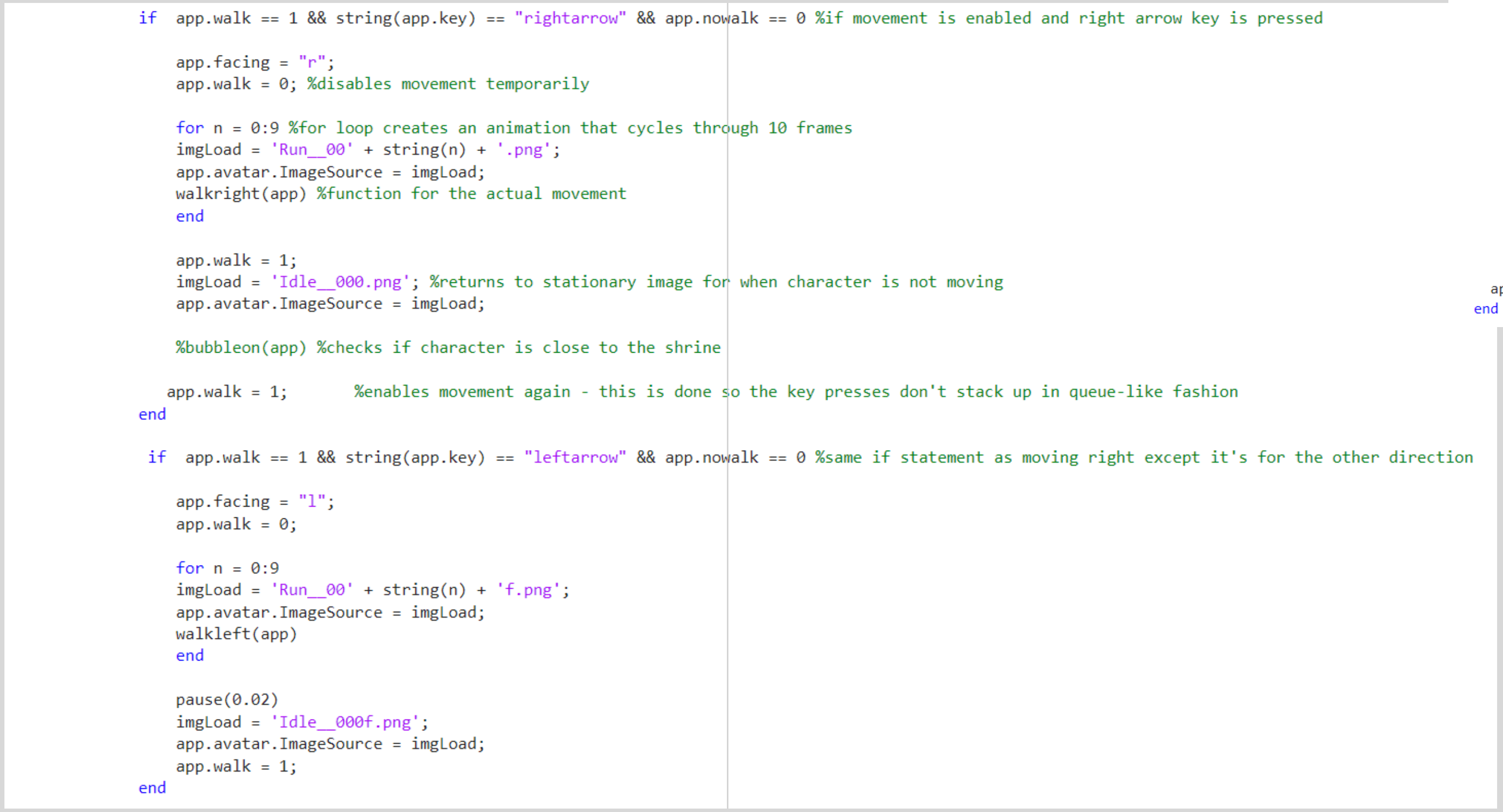
```
function refresh(app) %updates score and lives display

    app.ScoreTextArea.Value= (string(app.score));
    app.LivesLeftTextArea.Value = (string(app.lives));

end
```

- New Level Function

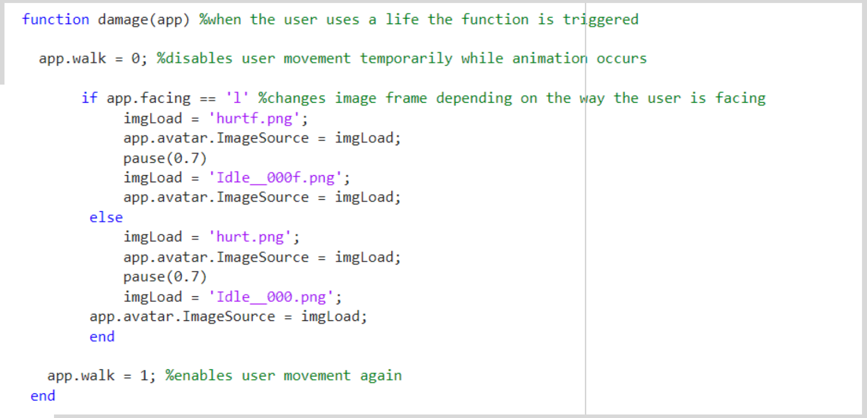


```
function newlevel(app) %used whenever a new level is needed

app.GoButton.Visible = 'off';   %hides relevant images and buttons
app.colourblockLabel.Visible = 'off';
app.MBox.Visible = 'off';
app.DBox.Visible = 'off';
app.SqtBox.Visible = 'off';
app.DifficultyButtonGroup.Visible = 'off';
app.OptionsLabel.Visible = 'off';
app.Image.Visible = 'off';
app.Image4.Visible = 'off';
imgLoad = 'closeddoor.png';
app.Image5.ImageSource = imgLoad;

app.LivesLeftTextArea.Visible = 'on'; %displays relevant images and buttons
app.ScoreTextArea.Visible = 'on';
app.LivesLeftTextAreaLabel.Visible = 'on';
app.ScoreTextAreaLabel.Visible = 'on';
app.LevelEditField.Visible = 'on';
app.LevelEditFieldLabel.Visible = 'on';
app.Image2.Visible = 'on';
app.Image5.Visible = 'on';
app.qopen = 1;

app.TextArea.Position(4) = 92; %sets position and font for the main display
app.TextArea.FontSize = 24;
app.TextArea.FontWeight = 'bold';
app.TextArea.Value = ("");


refresh(app) %function to update score and lives left display


focus(app.UIFigure); % sets focus to the app window so keypresses will be registered

app.avatar.Position = [40,60,150,100]; %sets position of the player
app.avatar.Visible = 'on';
app.walk = 1; %enables walking through use of key presses

 app.solved = 0;
 app.level = app.level + 1; %level counter
 app.LevelEditField.Value = app.level; %displays current level


 end
```

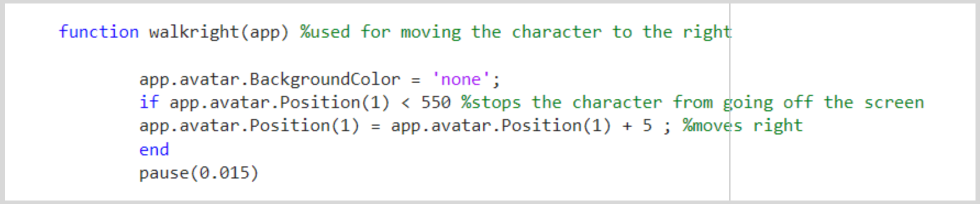
- Character Movement



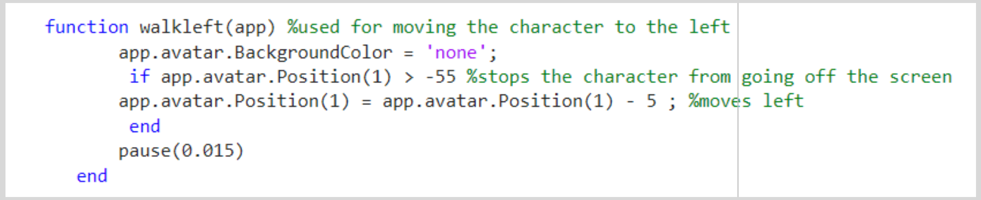
```
if  app.walk == 1 && string(app.key) == "rightarrow" && app.nowalk == 0 %if movement is enabled and right arrow key is pressed

    app.facing = "r";
    app.walk = 0; %disables movement temporarily

    for n = 0:9 %for loop creates an animation that cycles through 10 frames
    imgLoad = 'Run__00' + string(n) + '.png';
    app.avatar.ImageSource = imgLoad;
    walkright(app) %function for the actual movement
    end

    app.walk = 1;
    imgLoad = 'Idle__000.png'; %returns to stationary image for when character is not moving
    app.avatar.ImageSource = imgLoad;

    %bubbleon(app) %checks if character is close to the shrine

   app.walk = 1;       %enables movement again - this is done so the key presses don't stack up in queue-like fashion
end

 if  app.walk == 1 && string(app.key) == "leftarrow" && app.nowalk == 0 %same if statement as moving right except it's for the other direction

    app.facing = "l";
    app.walk = 0;

    for n = 0:9
    imgLoad = 'Run__00' + string(n) + 'f.png';
    app.avatar.ImageSource = imgLoad;
    walkleft(app)
    end

    pause(0.02)
    imgLoad = 'Idle__000f.png';
    app.avatar.ImageSource = imgLoad;
    app.walk = 1;
end
```



```
function damage(app) %when the user uses a life the function is triggered

  app.walk = 0; %disables user movement temporarily while animation occurs

      if app.facing == 'l' %changes image frame depending on the way the user is facing
          imgLoad = 'hurtf.png';
          app.avatar.ImageSource = imgLoad;
          pause(0.7)
          imgLoad = 'Idle__000f.png';
          app.avatar.ImageSource = imgLoad;
       else
          imgLoad = 'hurt.png';
          app.avatar.ImageSource = imgLoad;
          pause(0.7)
          imgLoad = 'Idle__000.png';
      app.avatar.ImageSource = imgLoad;
      end

   app.walk = 1; %enables user movement again
 end
```

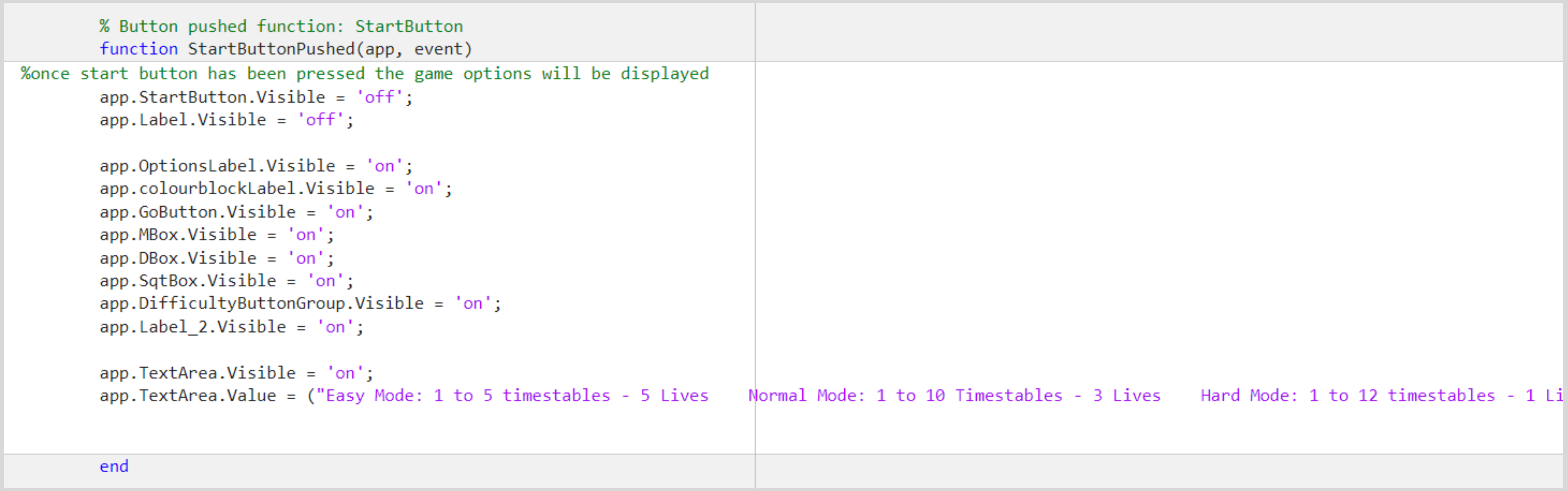


```
function walkright(app) %used for moving the character to the right

        app.avatar.BackgroundColor = 'none';
        if app.avatar.Position(1) < 550 %stops the character from going off the screen
        app.avatar.Position(1) = app.avatar.Position(1) + 5 ; %moves right
        end
        pause(0.015)
```



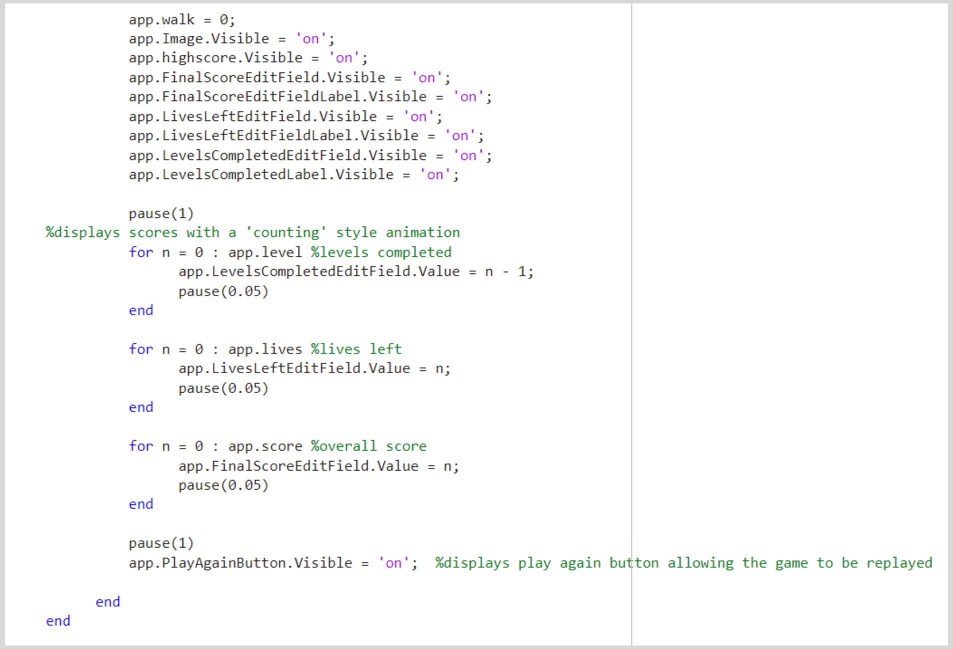
```
function walkleft(app) %used for moving the character to the left
        app.avatar.BackgroundColor = 'none';
         if app.avatar.Position(1) > -55 %stops the character from going off the screen
        app.avatar.Position(1) = app.avatar.Position(1) - 5 ; %moves left
         end
        pause(0.015)
    end
```

- Menu Call backs



```
        % Button pushed function: StartButton
        function StartButtonPushed(app, event)
%once start button has been pressed the game options will be displayed
        app.StartButton.Visible = 'off';
        app.Label.Visible = 'off';

        app.OptionsLabel.Visible = 'on';
        app.colourblockLabel.Visible = 'on';
        app.GoButton.Visible = 'on';
        app.MBox.Visible = 'on';
        app.DBox.Visible = 'on';
        app.SqtBox.Visible = 'on';
        app.DifficultyButtonGroup.Visible = 'on';
        app.Label_2.Visible = 'on';

        app.TextArea.Visible = 'on';
        app.TextArea.Value = ("Easy Mode: 1 to 5 timestables - 5 Lives    Normal Mode: 1 to 10 Timestables - 3 Lives    Hard Mode: 1 to 12 timestables - 1 Li


        end
```

- Scoreboard



```
          app.walk = 0;
          app.Image.Visible = 'on';
          app.highscore.Visible = 'on';
          app.FinalScoreEditField.Visible = 'on';
          app.FinalScoreEditFieldLabel.Visible = 'on';
          app.LivesLeftEditField.Visible = 'on';
          app.LivesLeftEditFieldLabel.Visible = 'on';
          app.LevelsCompletedEditField.Visible = 'on';
          app.LevelsCompletedLabel.Visible = 'on';

          pause(1)
%displays scores with a 'counting' style animation
          for n = 0 : app.level %levels completed
                app.LevelsCompletedEditField.Value = n - 1;
                pause(0.05)
          end

          for n = 0 : app.lives %lives left
                app.LivesLeftEditField.Value = n;
                pause(0.05)
          end

          for n = 0 : app.score %overall score
                app.FinalScoreEditField.Value = n;
                pause(0.05)
          end

          pause(1)
          app.PlayAgainButton.Visible = 'on';  %displays play again button allowing the game to be replayed

      end
end
```

Thanks for watching!