Contents

1 Lab 7 Overview

Lab 7 is a version of the video game Q'bert(Gottlieb 1982) written in ARM assembly language. Q'bert is depicted by the character 'Q' on a two dimmensional pyramid. Three types of enemies can be on the pyramid with Q'bert. Balls and snake balls behave the same with the exception that the snake ball becomes a snake when it reaches the bottom of the pyramid. There will not be more than two balls, and 1 snake on the pyramid at a time. Each time Q'bert moves to a new square it is cleared of '///' characters and awards the player points. Once all squares have been cleared the game begins again on a new level with the uncleared squares and an increased game speed. Q'bert begins the game with four lives. A life is lost whenever Q'bert jumps off the pyramid or occupies the same square as a ball or snake. The game ends after two minutes of gameplay. The objective of the game is for the player to score as many points before lives are lost or the game ends due to two minute timer. Ten points are awarded for each square cleared, and 100 points for each level reached. The player also recieves 25 bonus points for each life remaining at time out.

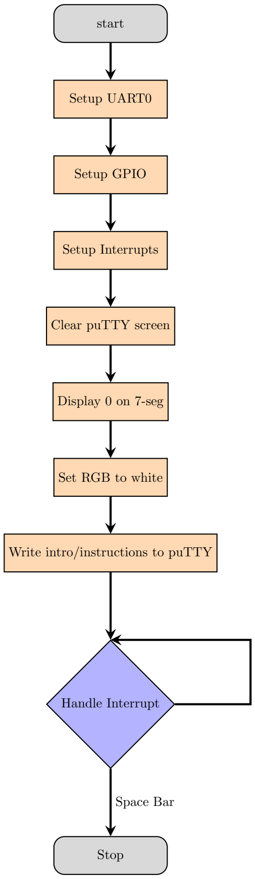
1.1 Lab 7 Usage

The UART, GPIO, and interrupts are all initialized when the game is started. The puTTY terminal is cleared, a 0 is displayed on the seven-segment display, and the RGB LED is set to white. An intro with instructions on how to play the game is written to the cleared puTTY terminal. The program then waits for interrupts, specifically a 'g' key press to start a new game, or a space bar press to quit the game.

1.2 Division of Work

David was the sole member for this lab, and is responsible for all code and documentation.

1.3 Lab 7 Flow Chart



2 Subroutines

All subroutines are stored in the library file. Each routine is imported at the beginning of the file.

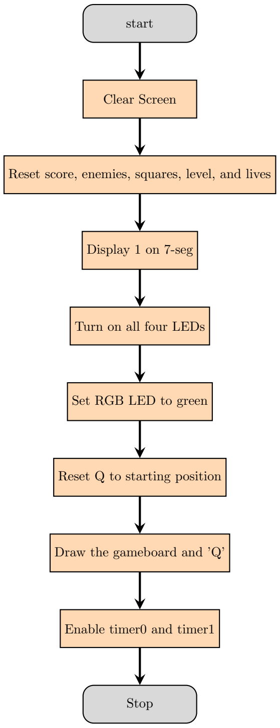
1. newGame
2. gameOver
3. resetAllSquares
4. increaseLevel
5. removeQ
6. redrawQ
7. FIQ_Handler
8. pauseGame
9. handleMoves
10. spawnEnemy
11. moveBall1
12. moveBall2
13. moveSnakeBall
14. moveSnake
15. randomNum

2.1 newGame Routine

This routine starts a new game. All values are reset for the new game. This routine is predominately intended to be used from the intro/instruction screen, or the game over screen but can be used at anytime the game is not paused.

Usage
The newGame routine first clears the screen of what is currently being displayed. The score us reset to zero. The number of enemies currently on the pyramid is set to zero. All squares are set to uncleared. The level is set to one, and the lives are set to four. A one is displayed on the seven-segment display to illustrate the current level. All four LEDs are turned on to represent that the player has four lives left. The RGB LED is set to green to show that the game is currently running. Q'bert's('Q') position is set to the starting position at the top of the pyramid, and the game board is drawn with 'Q' at top. Timer0 and timer1 are enabled(started).

2.1.1 newGame Flow Chart

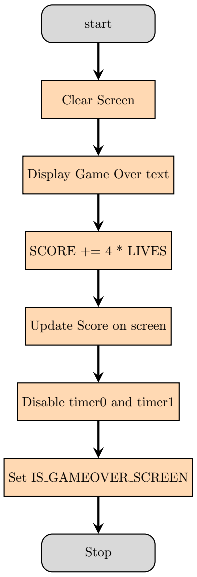


2.2 gameOver Routine

Displays the game over text when two minute timer is up or when all lives are lost. The player my either press 'g' for a new game, or space bar to quit.

Usage
This routine clears the current puTTY display and shows the game over text. The number of lives is multiplied by 25 and added to the current score. This final score is then displayed at the top of the screen. Both timers are disabled. The IS_GAMEOVER_SCREEN is set. This is checked by other routines in the FIQ handler to prevent certain key interrupts during the game over screen.

2.2.1 gameOver Flow Chart



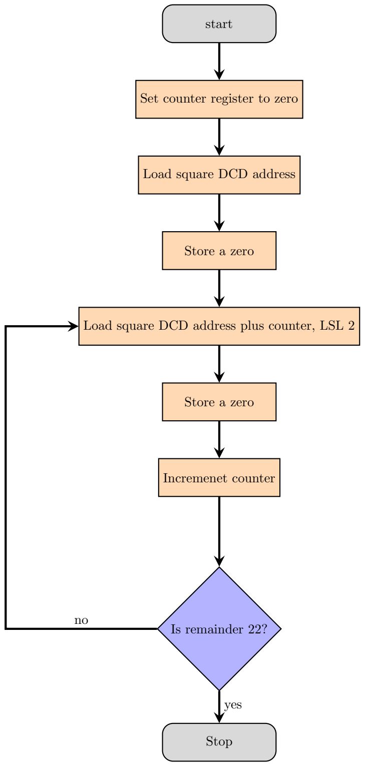
2.3 resetAllSquares Routine

Squares are set when Q'bert has explored them. They need to be cleared at the start of a new game, or when proceeding to the the next level.

Usage

A coutner register is set to zero. The first square DCD address is loaded. The value in that address is cleared. Move to next. Store a zero and increment counter by one. Check if the counter is equal to 22, and move to next address if not.

2.3.1 resetAllSquares Flow Chart

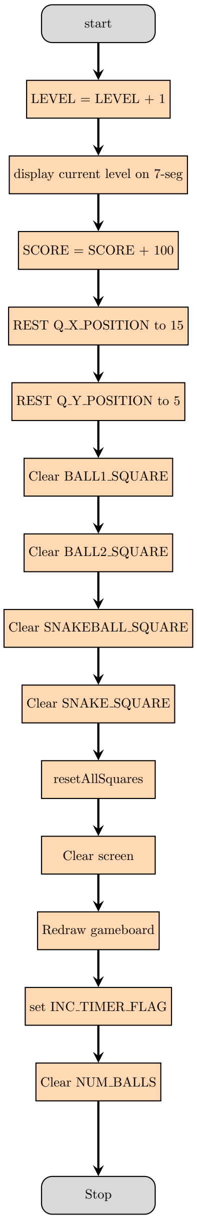


2.4 increaseLevel Routine

The level is increased to the next one whenever the player explores and clears all the squares on the pyramid. Q'bert is moved back to the starting positon, and all enemies are reset. The speed of the game is increased with each level, not to exceed two player, or one enemy, moves every 0.1 seconds.

Usage
The LEVEL is increased by one. The new value is displayed on the 7-segment display. One-hundred points are added to the score. Q'bert's X and Y positions are set to the starting square at the top of the pyramid. BALL1_SQUARE, BALL2_SQUARE, SNAKEBALL_SQUARE, and SNAKE_SQUARE are all set to zero. The screen is cleared and a new GAME_BOARD is displayed. INC_TIMER_FLAG is set and NUM_BALLS is cleared to zero.

2.4.1 inreaseLevel Flow Chart

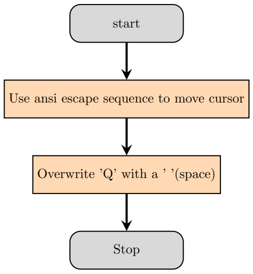


2.5 removeQ Routine

Replace 'Q' on the screen with a ' '(space).

Usage
Use ansi excape secquence, ESC[#;#f to move to line #, and column #. Output a ' ' space to overwite the 'Q' that is currently there.

2.5.1 removeQ Flow Chart

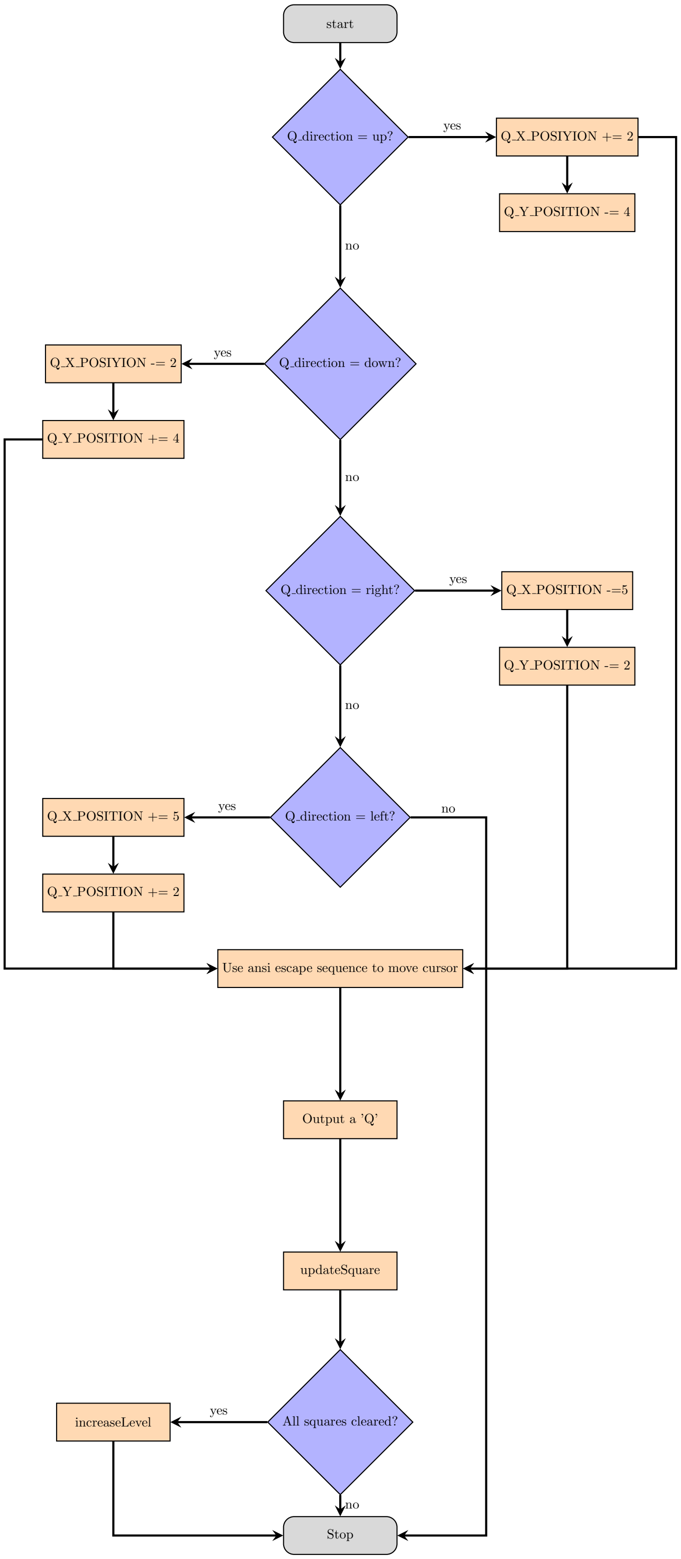

start
Use ansi escape sequence to move cursor
Overwrite 'Q' with a ' '(space)
Stop

2.6 redrawQ Routine

Routine that draws Q'bert ('Q') in his new position of the pyramid based on key inputs from the player. Called twice every second for the first level, twice every 0.9 seconds for the second level and 0.1 seconds less for every level after. It will stop decrasing at 0.1 seconds.

Usage
Check which direction Q'bert will be moved. Adjust Q'bert's X and Y position and update his square to refelect new position. Clear unexplored square and check if all squares have been explored. Incease level if all have been explored.

2.6.1 redrawQ Flow Chart

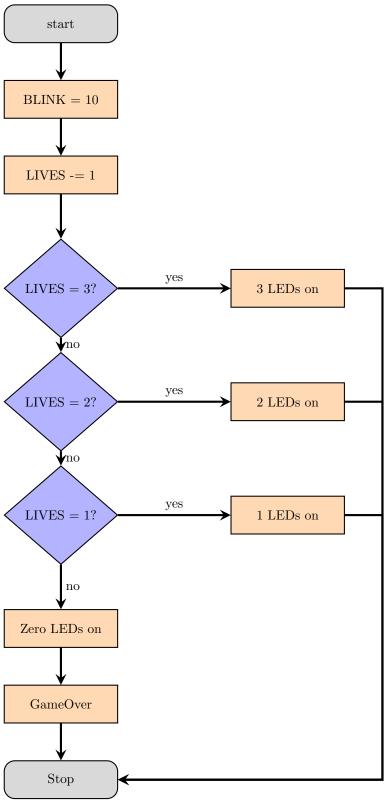


2.7 removeLife Routine

This routine doubles the speed that the character moves across the puTTy terminal.

Usage
Load the current value in the timer match1 register. Divide the value in half and store the new value in the timer match1 register.

2.7.1 removeLife Flow Chart

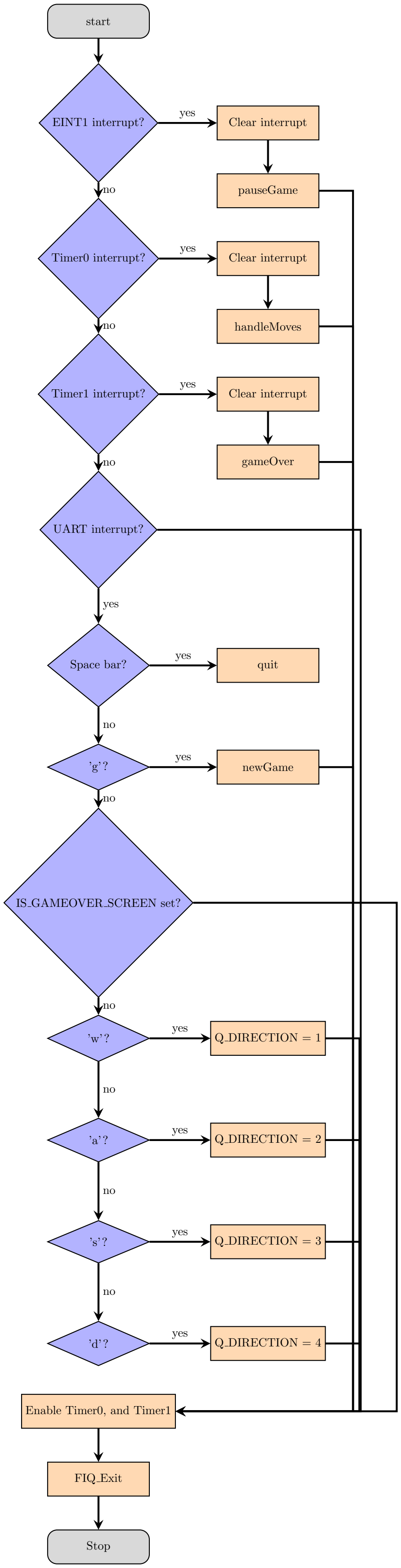


2.8 FIQ_Handler Routine

Code that handles the fast interrupts for Edge Sensitive Push Button EINT1, keyboard UART0, Timer0, and Timer1.

Usage
If the push button was pressed, the interrupt is cleared, and the pauseGame subroutine is called. If Timer0 caused the interrupt, it is cleared and handleMoves subroutine is called. Timer1 interrupt is a the two minute timer which calls gameOver. UART interrupts are automatically cleared. Quit is called if the space bar was pressed. NewGame is called if the 'g' was pressed. The 'a', 'w', 's', and 'd' keys set Q_DIRECTION. After all interrupts are handled both timer0 and time1 are re-enabled and FIQ_Exit is called.

2.8.1 FIQ_Handler Flow Chart

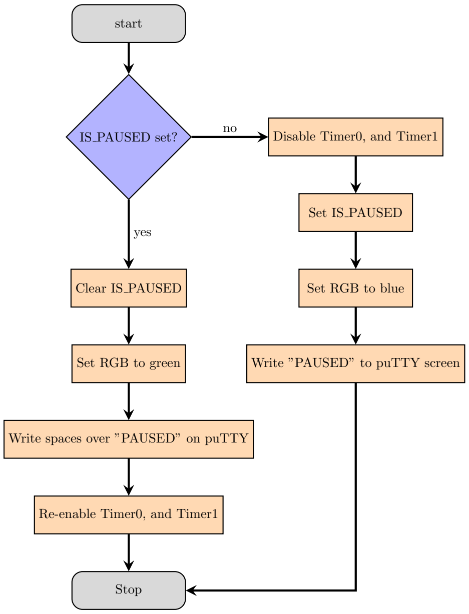


2.9 pauseGame Routine

This routine pauses, or unpauses the game when the EINT1 push button is pressed. No keyboard interrupts are handled when the game is paused.

Usage
IS_PAUSED is loaded and checked to see if the game needs to be paused, or unpaused. Whn paused, IS_PAUSED is set, the RGB is set to blue, "PAUSE" is displayed on puTTY, and both timers are stopped. Unpausing the game sets the RGB back to green, clears IS_PAUSED writes spaces over the "PAUSED" message on puTTY, and renanbles both timers.

2.9.1 pauseGame Flow Chart

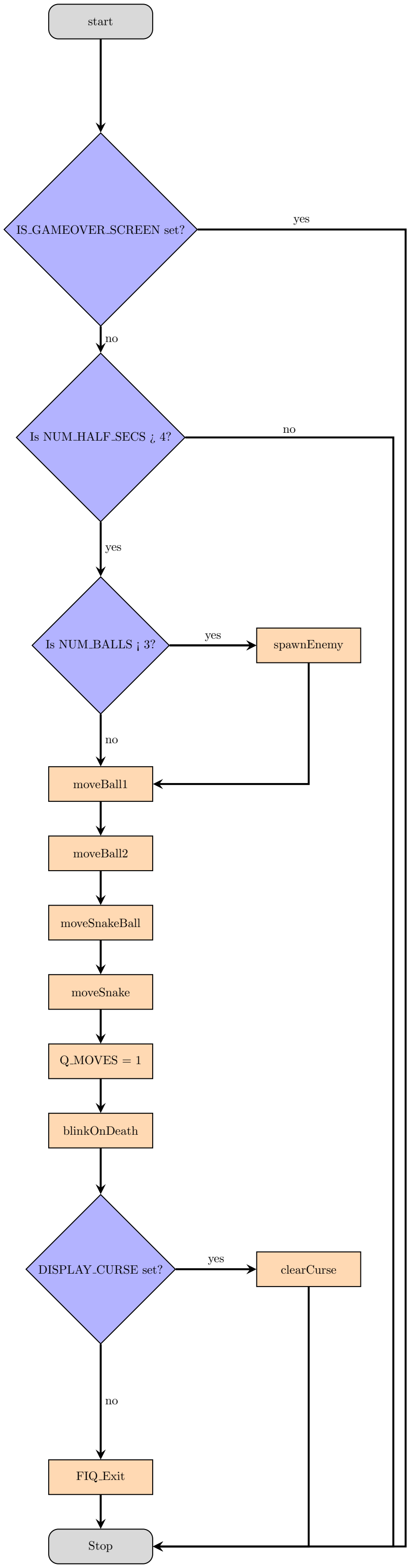

start
IS_PAUSED set?
no
Disable Timer0, and Timer1
Set IS_PAUSED
Set RGB to blue
Write "PAUSED" to puTTY screen
yes
Clear IS_PAUSED
Set RGB to green
Write spaces over "PAUSED" on puTTY
Re-enable Timer0, and Timer1
Stop

2.10 handleMoves Routine

This routine determines when enemies should be spawned and when they should be moved. It also controls how often the player can move Q'bert. Curses are also removed by from this subroutine. blinkOnDeath is called from here and will turn RGB red or off if the player recently died.

Usage
If IS_GAMEOVER_SCREEN is set then then no moves need to be handled, and routine goes straight to FIQ_Exit. If the game has not been running for at least two seconds then no enemies will be spawned. If there are aleady three enemies on the pyramid then no more need to be spawned. ALl currently enemies are moved. Q'bert is allowed another move. blinkOnDeath is called and RGB blinks red if a life was recently lost.

2.10.1 handleMoves Flow Chart

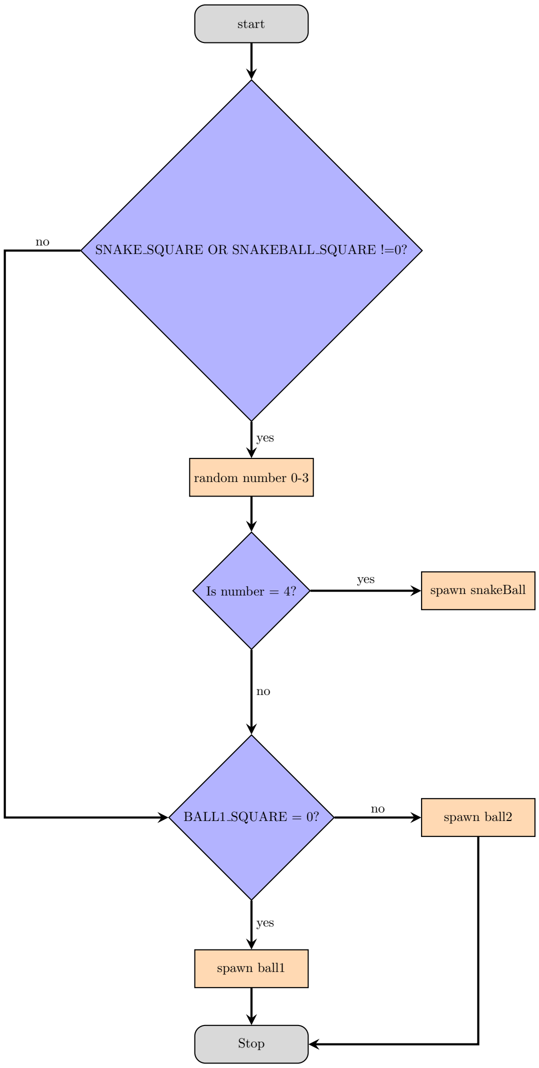


2.11 spawnEnemy Routine

Spawns a new enemy depending on which enemies are already spawned as well as random numbers.

Usage
If a snake or snakeBall already occupy a square on the pyramid, a snakeball will not spawn. If they are not already spawned there is a one in four chance to spawn one. Otherwise if there is no ball1 then ball1 will spawn. if there is already a ball1, then a ball2 will spawn.

2.11.1 spawnEnemy Flow Chart

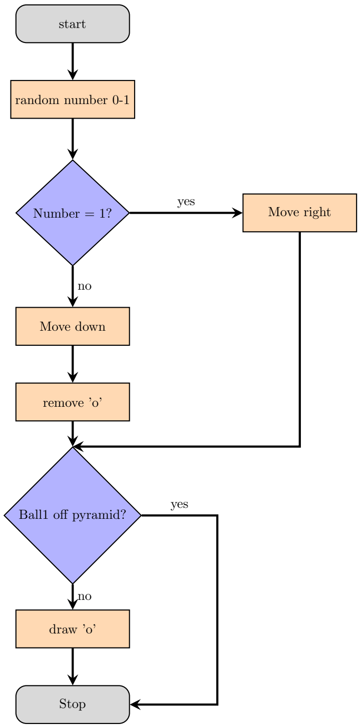


2.12 moveBall1 Routine

Moves ball1 randomly down,or right.

Usage
Get a random number between 0-1. If it is a 0, move down, otherwise move right. Remove the 'o'. Check if ball1 has moved off the pyramid. If not, draw 'o' in new square.

2.12.1 moveBall1 Flow Chart



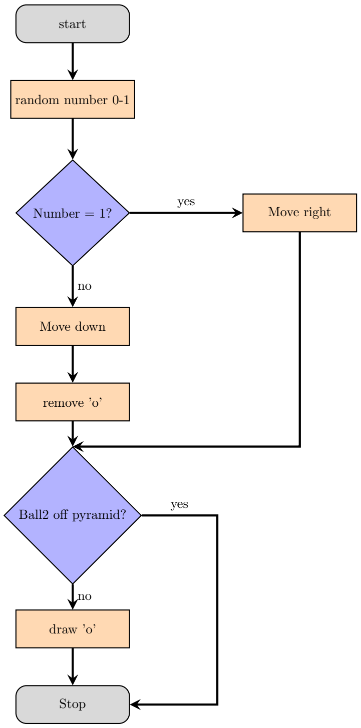
2.13 moveBall2 Routine

Moves ball2 randomly down,or right.

Usage

Get a random number between 0-1. If it is a 0, move down, otherwise move right. Remove the 'o'. Check if ball2 has moved off the pyramid. If not, draw 'o' in new square.

2.13.1 moveBall2 Flow Chart

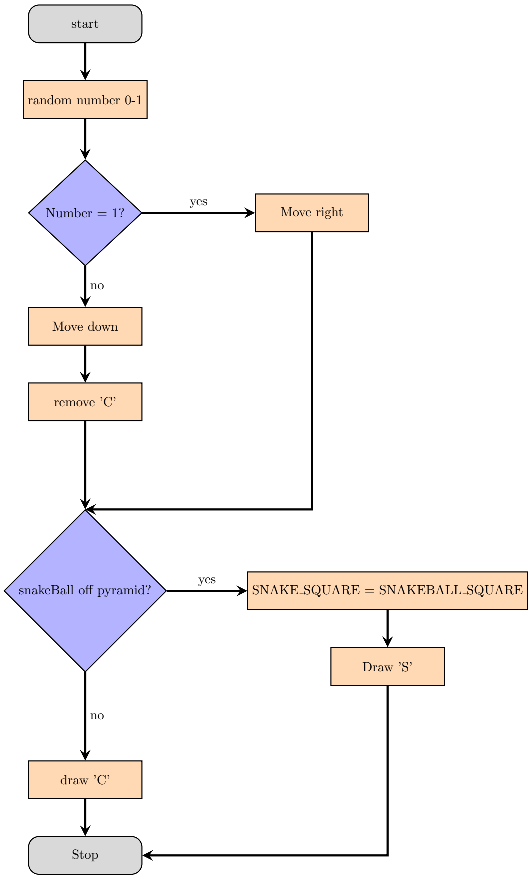


2.14 moveSnakeBall Routine

Moves snakeBall randomly down,or right. Spawns a snake when it reaches bottom of pyramid

Usage
Get a random number between 0-1. If it is a 0, move down, otherwise move right. Remove the 'C'. Check if the snakeBall has moved off the pyramid. If not, draw 'C' in new square. If it has moved off the pyramid replace the 'C' in the bottom square with a 'S' for snake enemy.

2.14.1 moveSnakeBall Flow Chart

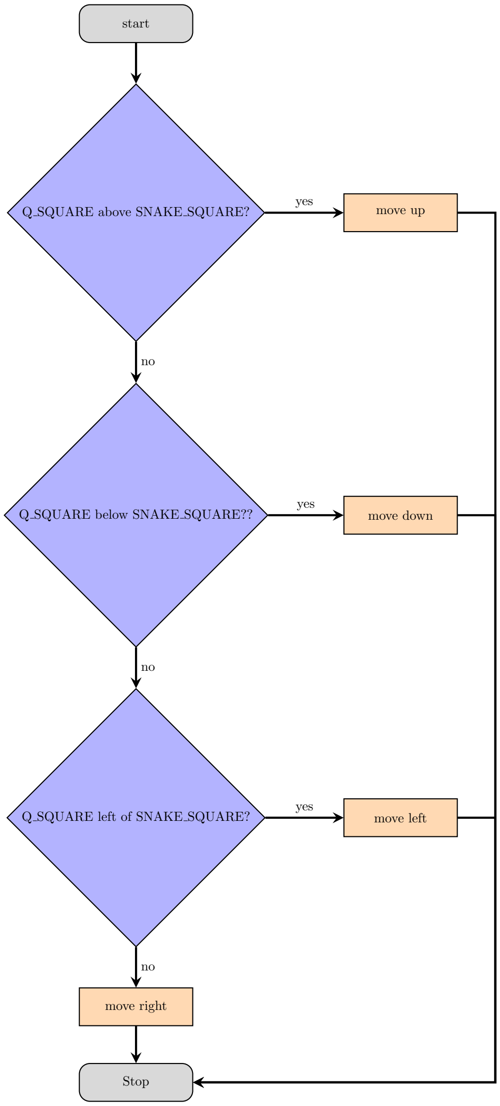


2.15 moveSnake Routine

Moves the snake in the direction of Q'bert on the pyramid. Only one snake will be spawned at a time.

Usage
If the current Q_SQUARE is above the SNAKE_SQUARE move the snake up. If the current Q_SQUARE is below the SNAKE_SQUARE move the snake down. If the current Q_SQUARE is left of the SNAKE_SQUARE move the snake left. If the current Q_SQUARE is right of the SNAKE_SQUARE move the snake right.

2.15.1 moveSnake Flow Chart

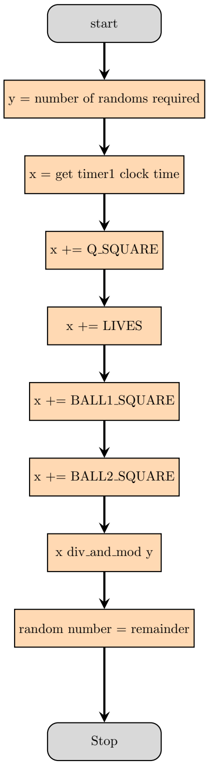

start
Q_SQUARE above SNAKE_SQUARE?
yes
move up
no
Q_SQUARE below SNAKE_SQUARE??
yes
move down
no
Q_SQUARE left of SNAKE_SQUARE?
yes
move left
no
move right
Stop

2.16 randomNum Routine

This routine take a number and generates a random number from zero through the given number minus one.

Usage
This routine gets the time from timer1 clocc register. This number is added to the number of lives the player has left, the squares that ball1 and ball2 are currently on. This total is then divided by the given number to generate a remaineder of zero up to the number given minus one(i.e. if given the number 4 a number between 0-3 will be returned).

2.16.1 randomNum Flow Chart


start
y = number of randoms required
x = get timer1 clock time
x += Q_SQUARE
x += LIVES
x += BALL1_SQUARE
x += BALL2_SQUARE
x div_and_mod y
random number = remainder
Stop