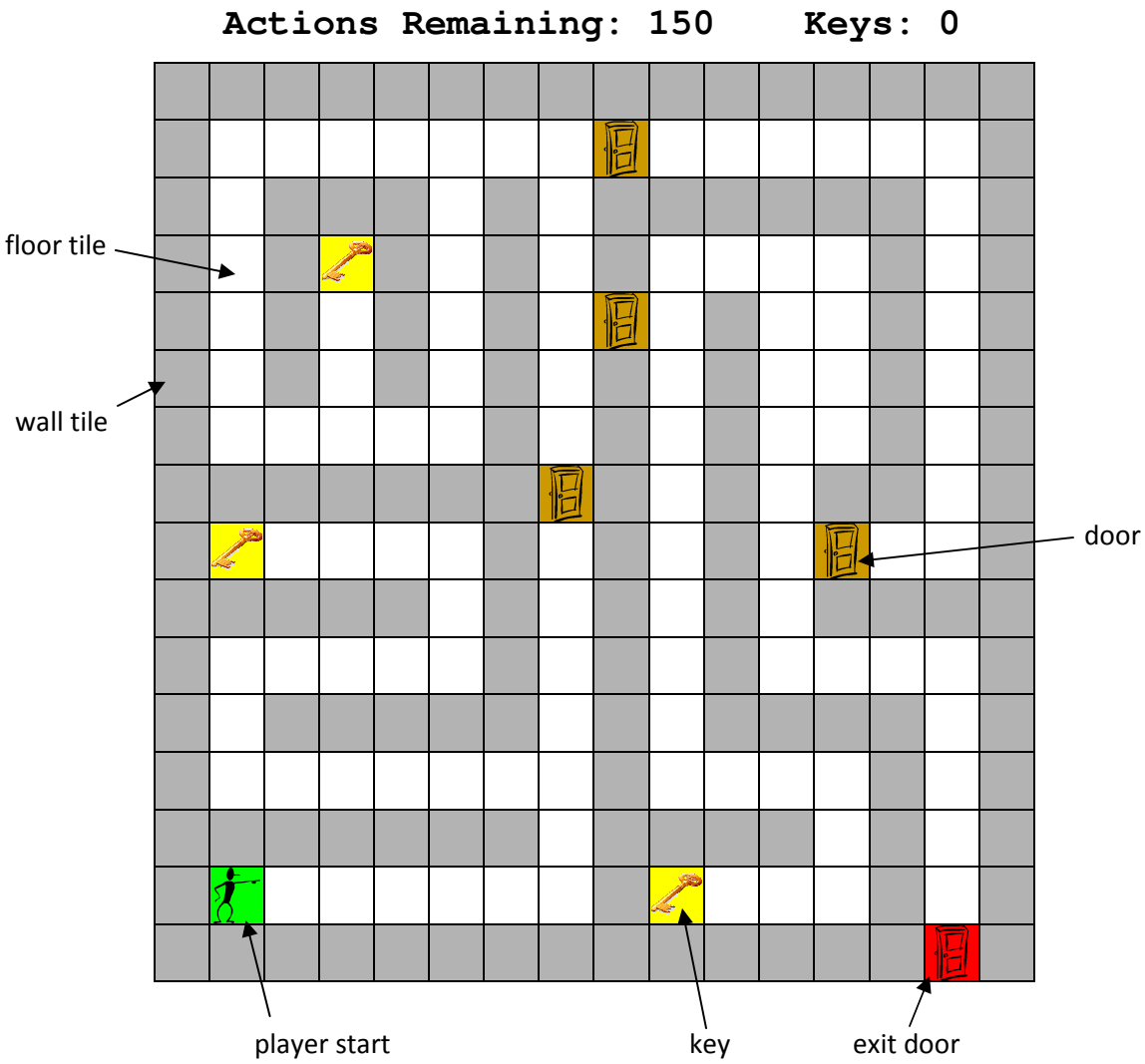
CPSC 359 – Winter 2014
Assignment 3
Raspberry Pi Video Game
55 points
Due March 31st @ 11:59pm (midnight)

Objective: Implement a video game that requires the player to travel through a maze with the goal of reaching the exit door within a specified number of moves. Solving the maze will require the player to move through the maze, collect keys, open doors that block the path, and open the exit door before the actions remaining counter runs out.



Game Logic

The *game environment* is a finite 2D *n* x *n* grid

- Each grid cell contains either a *wall tile* or a *floor tile*
    - Floor tiles may be marked as *player start* or *key*
    - Wall tiles may be marked as *door* or *exit door*
- A *game map* is an instance of the game environment (for a value of $n \geq 16$)
    - Specifies the maximum number of *player actions* for the map
    - Each cell of the grid is either a *wall* or *floor* tile
    - The cells on the edge of the map are filled with *wall* tiles
    - One floor tile is marked *player start* and one wall tile is marked *exit door*
    - There are *k* floors marked as *key* and *d* wall tiles marked as *door* ($1 < k \leq d$)
    - No floor tile is adjacent (in the cardinal directions) to more than one door tile
- A *game state* is a representation of the game as it is being played, and contains:
    - An instance of a *game map*
    - The *player's position* on the game map
        - Initialized to the position of the floor tile marked *player start*
    - The number of *actions remaining* that the player can perform
        - Initialized to the game map's maximum player actions
    - The number of *keys collected* by the player (initialized to zero)
    - A *win condition* flag (set if and only if the player opens the *exit door*)
    - A *lose condition* flag (set if and only if *actions remaining* equals zero)

The game transitions into a new state by the player performing an action

- Only transition to a new state if the player performs a *valid action*
    - Decrement by one the *actions remaining* counter in the new game state
- Action: Move by one cell in one of the four cardinal directions (up, down, left, right)
    - Move action is *valid* only if the destination cell contains a floor tile
    - Results in the *player's position* being set to the position of the destination cell
    - Moving into a floor tile marked as *key* will result in:
        - The number of *keys collected* being incremented by one
        - The removal of the *key* marking on the destination floor tile
- Action: Open a door or the exit door with a key
    - An open door action is *valid* only if:
        - Player is adjacent to a wall tile marked *door* or *exit door*
        - Number of *keys collected* is greater than zero
    - Opening a door results in:
        - Adjacent wall tile marked as *door* or *exit door* is replaced by a floor tile
        - Number of *keys collected* is decremented by one

The game is over when either the *win* or *lose condition* flags is set in the game state

Game Interface

Main Menu Screen

- The Main Menu interface is drawn to the screen
    - Game title is drawn somewhere on the screen
    - Creator name(s) drawn somewhere on the screen
    - Menu options labeled “Start Game” and “Quit Game”
    - A visual indicator of which option is currently selected
- The player uses the SNES controller to interact with the menu
    - Select between options using Up and Down on the D-Pad
    - Activate a menu item using the A button
    - Activating Start Game will transition to the Game Screen
    - Activating Quit Game will clear the screen and exit the game

Game Screen

- The current game state is drawn to the screen
    - Represented as a 2D grid of cells
        - All cells in the current game state are drawn to the screen
        - Each cell is square (width = height), and at least 16x16 pixels
        - 2D grid should be (roughly) in the center of the screen
    - Each different tile type is drawn with a different visual representation
        - ie: wall, floor, door, key, exit door, player start and player
        - Minimally, cells are filled with different colours based on tile type
    - Actions Remaining and Keys Collected are drawn on the screen
        - A label followed by the decimal value for each field
    - If the Win Condition flag is set, display a “Game Won” message
        - If the Lose Condition flag is set, display a “Game Lost” message
        - Both messages should be prominent (ie: large, middle of screen)
- The player uses the SNES controller to interact with the game
    - Pressing up, down, left or right on the D-Pad will attempt a move action
    - Pressing the A button will attempt to perform an door open action
    - All buttons need to be released before a new action can be performed
    - Performing a valid action will require the game state to be redrawn
    - Pressing the Start button will open a Game Menu
        - Two menu items: Restart Game and Quit
        - Visually display menu option labels and a selector
        - Menu drawn on a filled box with a border in the center of the screen
        - Normal game controls not processed when Game Menu is open
        - Pressing Start button will close the Game Menu
        - Press up and down on D-Pad to select between menu options
        - Pressing the A button activates a menu option
        - Activating Restart Game will reset the game to it’s original state
        - Activating Quit will transition to the Main Menu screen
    - If the win condition or lose condition flags are set
        - Pressing any button will return to the Main Menu

Grading:

1.	Main Menu Screen	
	a. Draw game title and creator names	2
	b. Draw menu options and option selector	2
	c. Select between menu options using up / down on D-Pad	1
	d. Press A button with Start Game selected to start game	1
	e. Press A button with Quit Game selected to exit game	1
2.	Game Screen	
	a. Draw current game state	
	i. All cells drawn according to interface specifications	7
	ii. Actions Remaining and Keys Collected drawn	3
	iii. Game Won message drawn on win condition	1
	iv. Game Lost message drawn on lose condition	1
	b. Draw game menu	
	i. Filled box with border in center of screen	1
	ii. Draw menu options and option selector	2
	iii. Erase game menu from screen when closed	2
	c. Interact with game	
	i. Use D-Pad to move player (if move action is valid)	4
	ii. Press A button to open doors (if open action is valid)	3
	iii. Press Start button to open game menu	1
	iv. Press any button to return to main menu (game over)	1
	d. Interact with game menu	
	i. Use up / down on D-Pad to change menu selection	1
	ii. Press A button on Restart Game; resets the game	1
	iii. Press A button on Quit; returns to main menu	1
	iv. Press Start button to close game menu	1
3.	APCS compliant functions	3
4.	Well structured code	
	a. Use of functions to generalize repeated procedures	5
	b. Use of data structures to represent game state, etc.	5
5.	Well documented code	5
	Total	55 points

Programs that do not compile cannot receive more than **5 points**. Programs that compile, but do not implement any of the functionality described above can receive a maximum of **7 points**.

Teams: You may work in teams of up to three students in order to complete the assignment, but you are not required to do so. Peer evaluation in teams may be conducted.

Demonstration & Submission: Submit a .tar.gz file of your entire project directory (including source code, make file, build objects, kernel.img, etc) to your TA via the appropriate dropbox on Desire2Learn. If you are working with group members in different tutorial sections, choose just one TA to submit to. You will also need to be available to demonstrate your assignment during the tutorial of the TA you submitted the assignment to (April 7th or 8th, depending on the tutorial section).

Late Submission Policy: Late submissions will be penalized as follows:
-12.5% for each late day or portion of a day for the first two days
-25% for each additional day or portion of a day after the first two days
Hence, no submissions will be accepted after 5 days (including weekend days) of the announced deadline

Academic Misconduct: Any similarities between assignments will be further investigated for academic misconduct. While you are encouraged to discuss the assignment with your colleagues, your final submission must be your own original work. Any re-used code of excess of 20 lines (20 assembly language instructions) must be cited and have its source acknowledged. Failure to credit the source will also result in a misconduct investigation.