CAB202 Assignment 1, Semester 2 2019

Assessment Weight: 30%

Context

You are tasked with bringing to life the classic characters of Tom and Jerry in a terminal based game. Tom is a cat with the simple agenda of catching the mouse, Jerry, running rampant in his house. Tom runs around the room Jerry is in, dropping mousetraps periodically, and constantly changing his direction to move toward Jerry. Jerry just wants cheese, and is faster than Tom, but after he has consumed enough cheese, and taken the chase to the second room of the day, he likes to have a bit of fun by introducing some weapons of his own to direct at Tom.

Level 1 Specifications

Task 1 - File IO, String Parsing and General Game Initialisation (6 Marks)

a) **File IO: Initialisation from Input Maps**

The game should expect a text file as a command line parameter. The general format of this text file is given in the appendix. You must be able to parse text files of any length, with any number of walls, traps and treasure encoded. You may assume the file will only ever contain one set of coordinates each for Tom and Jerry.

Please note: *The coordinates given in the text file are all relative to the screen size. You will need to convert these coordinates to screen coordinates when parsing the text file. Your game needs to adapt to the current terminal size, which could be any size greater than 80x24 (80 columns, 24 rows).*

b) **Other Game Setup: Status Bar**

A status bar should be drawn across the top 3 rows of the screen, the first row (screen row 0) should display **in order**;

i. Your student number
ii. The current game score (initially 0),
iii. The current lives of the active player (initially 5),
iv. The active player ('J' for Jerry, 'T' for Tom, Default: 'J'),
v. The time elapsed since game start in minutes and seconds format MM:SS (initially 00:00).

The third row (screen row 2), should display **in order**;

vi. The number of cheese currently on the screen (initially 0),
vii. The number of mousetraps currently on the screen (initially 0),
viii. The number of weapons currently on the screen (initially 0, see level 2 for details).

Task 2 – Tom and Jerry Movement (5 Marks)

a) **Jerry: User Controlled Movement**

By default, the user should be able to control Jerry using the 'w', 'a', 's', 'd' keys for up, left, down and right movement respectively. Jerry should move one pixel for each keypress, in the corresponding direction. Players should not be able to cross the status bar (the top 3 rows of the screen).

b) **Pause Mode**

The game should be able to be paused by pressing the 'p' key. In pause mode, all automatic game dynamics should be frozen including the timer, but the user should still be able to

move the player, collect treasure and lose lives. The game **should NOT** start in pause mode.

c) **Tom: Automatic Movement**

Tom aims to find and catch Jerry, while dropping mouse traps randomly throughout his journey. Tom changes speed and direction randomly whenever he collides with a wall. Marks will be also be awarded for this criterion if advanced movement is correctly implemented as specified in 5b(ii). Players should never cross the status bar (the top 3 rows of the screen).

d) **Basic Collision Detection**

Tom and Jerry can't walk through walls, nor can they leave the bounds of the room they're currently in (they must be confined within the borders of the screen and visible). You must implement collision detection such that Jerry will not move any further if the next step will cross a wall, screen border or the status bar. Tom should also not cross walls, screen borders, or the status bar, but he must automatically change speed and direction upon collision with any of these game elements. Collision detection must be present between all implemented elements of Tom, Jerry, Walls, Cheese, Traps, Weapons, Borders and the Status Bar. Behaviours exhibited upon collision are described in the relevant subsections of Task 3 and 4.

Task 3 – Traps and Treasure (5 Marks)

a) **Cheese Appearance**

After the start of gameplay, cheese should appear in random locations at 2 second intervals until a maximum of 5 cheese are displayed on the screen. Cheese should only appear in clear locations on the screen; there should be no overlap with existing game elements. Cheese should never appear in the status bar (the top 3 rows of the screen).

b) **Cheese Interactions**

Jerry is happiest when full of cheese. He will eat any cheese he crosses, without limit, but is happy to start exploring the next room once he has consumed 5 pieces of cheese in the current room. Each cheese he eats adds 1 to his score. The cheese will keep appearing at 2 second intervals after Jerry is full, but the door will unlock as soon as 5 cheese have been consumed (5 points accrued). Jerry does not enter the next room until he moves to the door, but the door must appear in the room and remain visible once 5 points have been accrued. Tom can collide with cheese and continue on his path without effect.

c) **Mousetrap Appearance and Interactions**

Tom will randomly drop mousetraps as he moves around the screen, at 3 second intervals. He only has 5 traps, so cannot lay more than 5 in the room, but he does automatically regain his traps once Jerry has been caught by them. See life source management (3f) for Jerry collision details. Mousetraps should only appear in clear locations on the screen; there should be no overlap with existing game elements. Tom can collide with mousetraps and continue on his path without effect. Traps should never appear in the status bar (the top 3 rows of the screen).

d) **Door Appearance: Advance or Win**

Once Jerry has had sufficient cheese (5 pieces), a door must appear somewhere on the screen. This door should be denoted by an 'X' character (capital X), and should not appear anywhere that overlaps any current elements of the room (i.e. it must appear in a free space,

where there is no overlap with cheese, walls, characters, traps, status bar, etc.). When Jerry collides with this door, he should advance to the next level in the second room (if level 2 has been implemented), or he should win the game and the game over screen should appear. Door appearance is based purely on cheese consumption, not points accrued, in the current room.

e) **Life Source Management**

Jerry loses a life each time he collides with a mousetrap, and each time he is caught by Tom. If Tom catches Jerry, in addition to losing a life, Tom and Jerry both reset to their initial locations. All other elements of the game remain the same. If the player's lives reach 0 at any point during gameplay, the game over screen should appear.

f) **Game Over Screen**

If the player wins or loses the game, a game over screen should appear and the user should be offered the option to either Restart or Quit. Restart must reset all game play states to the initial state as read from the text file. Quit must return immediately to the terminal.

Level 2 Specifications

PRE-REQUISITES: *A decent implementation of level 1 must be demonstrated before marks can be achieved for level 2. You must have implemented the ability to complete level 1 and advance to level 2 through the door, as described in the level 1 specifications. You must also implement a shortcut where the user is able to cycle through levels by pressing the 'l' key to qualify for level 2 marks.*

Task 4 – Infinite Levels and Weaponisation (7 marks)

a) **Additional Text File Argument**

To implement level 2, you must pass at least one **additional** text file argument via command line. This second argument will be read to initialise level 2. All further task 4 and 5 behaviours should be demonstrated in level 2 (and further levels if implemented).

b) **Jerry's Turn: Weapons**

Give Jerry the ability to fire a weapon, such as a firework, by pressing the 'f' key. This firework should always shoot toward Tom, and for full marks, your firework should constantly update its angle toward Tom's current location as he moves around. If the weapon collides with a wall or goes off the screen before reaching Tom, the firework must disappear and the number of weapons currently on the screen should be updated accordingly. If the firework hits Tom, it should disappear and his location should reset to the initial position given in the text file for the current level. The firework can pass cheese and mousetraps without collision. Jerry gains 1 point for each firework that hits Tom. Weapons should never cross the status bar (top 3 rows of the screen).

c) **Infinite Levels**

The game should accept an arbitrary number of text files as command line arguments. Each text file provides the structure for a new level of the game. Each file should provide a different room structure, and should be accessible as with all levels; through a door which appears after 5 cheese have been consumed **in the current level**. The overall total points should still reflect points accumulated across all levels, including points accrued through weapons. 10 text files have been provided for you to use during testing, and you are encouraged to create more of your own by carefully following the same structure as the example files. To achieve full marks for this feature, you must also have implemented correct playability with arbitrary walls.

Task 5 – Demonstrate Mastery (7 marks)

a) Arbitrary Walls

Once Tom and Jerry enter the second room of the game, we find that the walls are less orderly. Many of them are diagonal, but both Tom and Jerry will continue to exhibit the same behaviours with these new walls. This will involve writing an updated, creative solution to collision detection that will work for walls of **any** diagonal or straight configuration.

b) Player Enhancement and Fluidity

The user should be able to press the 'z' key to switch character, and players should have enhanced abilities. The character which is not currently in play should demonstrate an appropriate movement. This movement should be in keeping with the theme that Tom seeks Jerry, and Jerry attempts to evade Tom while collecting cheese. This will involve;

i. Implementing an intelligent 'evasion' movement mode for automated Jerry to avoid user-controlled Tom while 'seeking' cheese,
ii. Implementing a 'seeking' movement mode for automated Tom to seek user-controlled Jerry,
iii. Allowing manual placement of cheese to lure automated Jerry closer to user-controlled Tom by pressing the 'c' key (up to a maximum of 5 on screen at once),
iv. Allowing manual placement of mousetraps at the current location of user-controlled Tom by pressing the 'm' key (up to a maximum of 5 on screen at once),
v. A lives system for Tom (initially 5 lives, decrements by 1 per firework hit),
vi. Automation of Jerry's weapons (1 firework should fire per 5 seconds),
vii. A points system for Tom, such that he gains;
    - 1 point each time Jerry gets caught by a mousetrap,
    - 5 points if he catches Jerry.
viii. The ability to advance to the next room after Tom has accrued 5 points in the current room. This should still adhere to the door specifications given in specification 3d.

c) Other Approved Extension

Through consultation with your tutor, you may also demonstrate mastery through an approved equivalent alternative approach.

General Expectations – Penalties Up To 5 Marks Apply If Violated

➔ A statement of completeness is required. A penalty of 2 marks will be applied if you do not submit a statement of completeness, or include misleading or falsated statements.

➔ Penalties up to 3 marks will be applied if the code is not implemented with an acceptable level of comprehension and maintainability, or exhibits any general defects or undesirable behaviour not otherwise covered in this document. This includes but is not limited to such things as:

- Errors in object motion, such as objects jumping more than one character position per frame;
- Objects moving outside their permitted area;
- Failure to clear the screen appropriately between updates;
- Collisions involving invisible or non-existent objects;
- Disturbing or flashing display;
- Sluggish response times, excessively fast response times, or other performance defects which render the simulation difficult to operate according to specification.

- Gratuitous errors in program structure and organisation, including but not limited to: inappropriate use of recursion; use of inappropriate data structures such as linked lists or binary search trees; using #include to insert the contents of source files rather than header files; uploading of multiple versions of the same source file.
- General readability and maintainability errors, including but not limited to: functions exceeding 25 executable C language instructions, where comma separated expressions with side effects will be classified as separate executable instructions; functions which are substantially identical to any other function; limited or no attempt to implement suitable documentation comments; failing to adhere to professional and consistent formatting; excessive use of global variables.

Appendix – Text File Example and Explanation

You will be provided with some example text files, but will need to be able to parse any text file of the same general structure. There are three different characters which will be used as identifiers at the start of each line, these being ‘W’, ‘T’, and ‘J’. These characters are indicators as to what the remaining digits on the line are representative of, where;

- ‘W’ indicates the line will include the relative end point coordinates (x1, y1) and (x2,y1) for a wall, formatted as follows;
  - W x1 y1 x2 y2
- ‘T’ indicates the line will include the relative starting coordinates (x, y) for Tom, formatted as follows;
  - T x y
- ‘J’ indicates the line will include the relative starting coordinates (x, y) for Jerry, formatted as follows;
  - J x y

All coordinates will be presented relative to the screen size, meaning they will all range from 0 to 1. You must interpret these coordinates by multiplying by (screen_width()-1) or (screen_height()-1) accordingly.

W 0.2 0.1 0.2 0.2
W 0.2 0.2 0.8 0.2
W 0.8 0.2 0.8 0.1
W 0.25 0.25 0.35 0.25
W 0.45 0.25 0.55 0.25
W 0.65 0.25 0.75 0.25
J 0 0
T 1 1
W 0.1 0.4 0.3 0.4
W 0.1 0.4 0.1 0.9
W 0.1 0.9 0.7 0.9
W 0.7 0.9 0.7 0.7
W 0.7 0.7 0.4 0.7
W 0.95 0.65 0.85 0.65
W 0.85 0.65 0.85 0.45
W 0.85 0.45 0.95 0.45