Computing Science CMPT 361 | **Spring 2016**
Instructor: Tom Shermer | **Simon Fraser University**

Assignment #1 (30 marks)

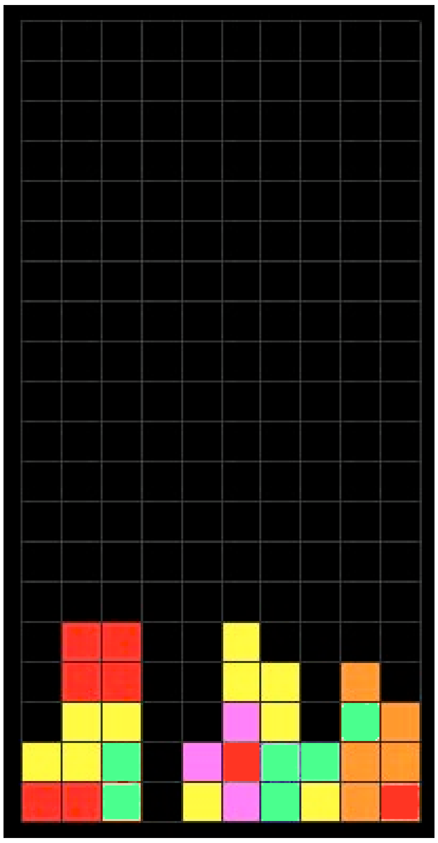
Written part due: February 17th, Wednesday, in class.
Programming part due: February 17th, Wednesday, at 11:59 pm.

Problem 1 (24 marks): Tetris meets Falling Fruits

You will implement a simple interactive game that combines some features from Tetris and Falling Fruits. Any visual flare that you wish to add to the appearance of your game will be judged by the grader and may be credited at his discretion.

The Falling Fruits game is similar to Tetris. (An example Falling Fruits game can be found at http://www.wordgames.com/search.html?query=falling+fruits).

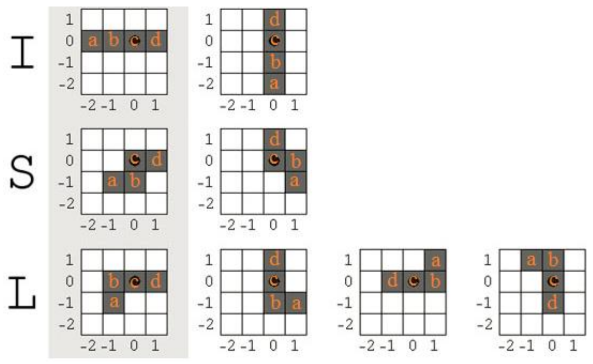
Your game window consists of a 20x10 square grid of appropriate size, e.g., so that the window will fit in the screen comfortably. Different fruits are represented by squares of different colors. You program should have five different fruits with color coding specified in figure (b). Each time, four fruits (chosen randomly with repetitions from the five types of fruit) fall from the top of the screen. (Note that the online Falling Fruits game is different and drops only three fruits at a time.) As indicated in the figure (c), these fruits are arranged in a I, S, L, or T shaped Tetris tile, which has a pivot of rotation indicated by a black dot. The T shape is not pictured but has squares (0,0), (-1, 0), (1, 0), and (0, -1) with a pivot center at (0,0). Also, the L and S shapes should come in left- and right- handed versions (left-hand versions are shown—flip horizontally to get the right-hand versions). We will not use the 2x2 square Tetris tile.



(a) The screen of the falling fruit game.

Grape	Apple	Banana	Pear	Orange
Purple	Red	Yellow	Green	Orange

(b) Use squares of different colors to represent different fruits. This table shows the color coding of fruits.



(c) The first column shows the "base" I, S, and L shapes; the other columns show the shapes when rotated.

You are advised to complete this problem in several steps:

(a) [6 marks] Tile and grid rendering and tile downward movement

Set up the game window with grid lines. At each time, randomly select a Tetris tile consisting of four fruits and drop it from the top of the game window. The starting position and orientation are chosen randomly. You can control the speed of its movement to suit your game playing. Movement of the tiles will be aligned with the grids and at uniform speed. For this step, the tiles can drop straight through the bottom. After one tile disappears, a new tile is dropped.

(b) [3 marks] Stack-up

In this step, the tiles will stack up on top of each other and the bottom of the game window will offer ground support. When any square of a falling tile is stopped, the entire tile stops (this is the Tetris rule for stacking, not the Falling Fruits one).

(c) [8 marks] Key stroke interaction and tile movements

The four arrow keys will be used to move the dropping fruit array. A pressing of the "up" key rotates the tile *counterclockwise* about its pivot, 90° at a time. The "left" and "right" key presses result in lateral movements of the array, one grid square at a time. The "down" key accelerates the downward movement. At no time should you allow a tile to collide with any existing tiles or the border of the game window.

(d) [7 marks] Additional game logic

1) When three same fruits are consecutive in a row, column, or diagonal, they will be removed and the tiles above them will be moved down. 2) When a row is completely filled, it is removed and the tiles above it will be moved one row down. Game terminates when a new tile piece cannot be fit within the game window. Press 'q' to quit and 'r' to restart. Pressing any of the arrow keys should not slow down the downward movement of a tile.

Note that the above steps build on top of each other, in order. You need not submit individual programs to correspond to these steps. If you can implement all the required parts, a single, complete program is sufficient. Skeleton code will be provided.

Although the marks have all been alloted for functionality above, the grader may also deduct marks for sloppy or confusing code, or for failure to follow laboratory procedure (e.g. not handing in the files specified).

<u>**Submission**</u>: All source code, a ***Makefile*** to make the executable called ***FruitTetris*** (the make command should be simply "make"), and a ***README*** file that documents any steps not completed, additional features, and any extra instructions for your TA.

Problem 2 (6 marks): Rigid-body transformations

Suppose you are given a 3D transformation Φ specified by a matrix M of the following form.

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

where the upper 3 by 3 submatrix R of M is orthonormal, i.e., $R^T = R^{-1}$.

1. [2 marks] What is the inverse of M? Note that you should not use brute force or a package such as Maple or Matlab to answer this question.
2. [1 mark] Let V = P_2 - P_1 be a vector in 3-dimensional real (Euclidean) space, where P_1 and P_2 are points in that space. Is it the case that Φ is linear in 3D? In other words, does $\Phi(P_2 - P_1) = \Phi(P_2) - \Phi(P_1)$?
3. [3 marks] Prove that the transformation Φ preserves lengths, angles, and the area of triangles in 3D.