=============================================

= README for ex5: Game Project: “Aquanoid” =

=============================================

===================

= student details =

===================

Name: Yotam Tsorfi

ID: 034612317

Date: 25/06/2015

=========================

= Project description =

=========================

I implement a game in type of brick breaker.

The user has to “destroy” all the bricks by hitting it

With a ball that has a velocity and direction when it collide

Via paddle that located at the bottom of the screen and with the game

Frame.

Once the user “destroy” all the brick in the level he pass to the next one.

The user can earn points from different bricks, and can get different gifts (bad or good) during the game while hitting the gift brick.

======================

= File Description =

======================

Screen.h, WelcomScreen.h, HelpScreen.h, WinGameScreen.h, GameOver.h:

Each class holding different picture to the game, for example the help

Screen holding two buttons for start the game and help button.

All the screen classes inherit from Screen.h.

GameObject.h:

Holding pure virtual functions that will be used by other inherited classes, pure function like “draw”, “collide”, “isColiide”.

Ball.h:

This class represent the ball in the game, set the ball velocity and it direction.

This class holding the movement functions and getters & setters for the ball object created.

Paddle.h:

This class represent the paddle in the game, set the paddle first location, size etc. the class implement the virtual function from the gamObject class name “collide”, and set the ball direction according to the hitting spot on the paddle.

Brick.h + Brick.cpp, GiftBrick.h, Unbreakable.h, ExplodeBrick.h:

Brick.h has pure virtual function, and this class represent the “regular-brick”.

Brick.cpp holding the function named “ballUpdate” which update the ball direction according to the hit spot on the brick.

The other brick classes inherit from Brick.h and implement the required functions.

Gift.h + Gift.cpp, BadGift.h, GiftFlying.h, GiftBigger.h:

This class represent the Gifts in the game, it has virtual function such “collide” “move” and “draw” that implement part of those functions.

Board.h + Board.cpp:

This class represent the Board in the game, it holding a map contain pair of : <std>::pair<int, int> -> that represent the brick location, and the second is the “Brick” object .

This class has function such “setBrickRoDestroy”, “updateBoard”, which will be balled during the game in order to remove the bricks from screen when hitting them.

Board.cpp has the function “buildBoard” which get in it arguments the current level and the windows – this function insert the right brick object to specific place according to the character matrix that has been created while reading the text file.

Level.h + Level.cpp:

Level.h holding all the string constant path for the textures and song files location at the utilities folder in the game.

This class also holding functions like “addChar”, “addRow” which will be called at the time of reading the level.txt in order to build a character matrix that represent the different brick location.

Controller.h + Controller.cpp:

The Controller class receive the renderWindow of the game, it create the object “Ball”, “Paddle”, “Board” , Screens objects, all the tiny members such number of life amount, number of level, and so on. The Constructor of this class actually start the game, by calling the function which read from the file and then start the hierarchy build of the board. The Controller class responsible to change the enum members according to the game changes. This class has function which called during the “while window function”, and update the objects during the game.

MyException.h:

The Exception class, holding the function “show” which print the description of the error and the number of line it’s happen.

I use the try & catch when I load any file from the game folder “utilities”.

main.cpp:

The main create the sf::RenderWindow of the game, create the Controller object.

levels.txt:­

The file holding the character sign such: B, G, X, U, that represent different brick object and it location on the board game.

=======================================

= Main data structures =

=======================================

As I mention above, at the Board class has a \_brick2D

Member – which is map of location (int pair) and Brick object.

During the game I scan this map and check by Polymorphism each brick in term of collide, etc.

In level.h I create global function of “get Texture”, “getSound”, “getFont” which load each file just once when calling it from other classes. The sound/textures are static and identifying by enum members.

===========================

= algorithms =

===========================

I iterate through the maps object member, and on the gifts vector in order to apply other functions.

Thanks to the polymorphism, I used it to “mark” specific object that need to been deleted from the gameBoard/map , and each interaction between the games objects work automatically when each class implement the required virtual functions and do the changes on some other object via down casting.

================

= Known bugs =

================

================

= Enjoy =

================