Frogger SFML report:

Module name: C++ Programming

Module code: IMAT1212

Title of the Assignment: Frogger

Lecture tutor: Coupland S

Lab tutor: Kumar V

Name: Mike Cheung

Pnumber: p14133881

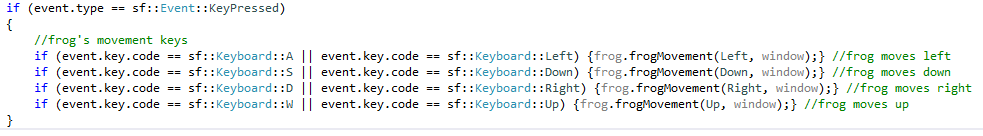
Introduction:

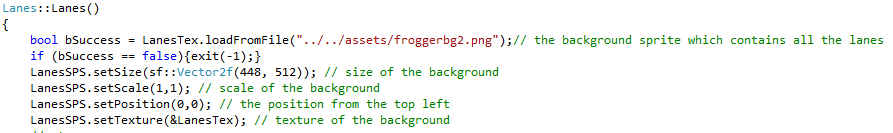
In assessment 2 we have to make the classic arcade game, Frogger. I had used most of the original sprites for the game that I have made and this gives the game the same type of feelings as the original game developed by Konami in 1981. I have implemented most of the game mechanics in this version of Frogger that I have made in this project. One of the most common Frogger’s mechanic such as movement of the frog is created as one of the first mechanics I had put into the game. The movement keys are made with a simple W, A, S, D or the arrow keys as the buttons used for the game when moving the frog. During this project, I have used software such as Adobe Photoshop to crop out the sprites into a .PNG image file so that the game sprites are to a high standard. My program is made to loop when the user has won or lost the game, I have made this to make this game of Frogger continues when the player has lost all 3 lives or when all the frogs is in the base.

What I have achieved:

I have created the game in SFML with sprites and implemented most of the mechanics such as life counters is decreased when frog loses a life by getting hit by a car, stinking in the water or walking off the viewable screen. The frog has movement and it can move left, right, up and down. These frog movements are linked to the arrow keys and W, A, S, D as well. I have made more than three areas on the screen, the pavement at the bottom and middle of the screen are the safe zones of the game. The road area where traffic moves horizontally left and right across the screen at different speeds and have different sprites with different spawn rate, the cars can kill the frog. The water area where logs moves horizontally left and right across the screen at different speeds and have different sprites with different spawn rate too, however, unlike the road area which are safe to walk on the logs are the ones that are safe to walk on and move along with the logs and the water area can kill the frog. The home area at the top of the screen which also contains frog bases are the goals of the frog to reach which can also kill the frog if the frog enters the base when another frog is inside the base. The frog has three lives, which are shown by life counters, the frog loses a life by being hit by a car, walking off screen, and going into a base with another frog or jumping into the water. When the frog loses all three lives the game ends and shows the game over screen and the game reset after a short amount of time. When all of the frogs reach the frog bases the game starts at a new level where all the frogs from before in the frog base are gone. I was also able to implement logs within my game which works differently from cars, although the logs and cars are spawned alike, however, unlike the cars which will kill the frog is they touch, the logs are actually the safe zone or area when going into the bases. I have made the water zone of this game a death zone and therefore would kill the frog if the frog isn’t on the logs. I have also made the bases to fill up with a frog when a frog has reached the base, this is the main goal of the game which is to try to fill up all of the bases. I have also implemented movement for the logs to have different spawn time and speed which are also randomly generated, therefore every time the game is played the patterns would be different. The game is working like it should, such as if the frog reaches the base, it’s safe and if all of the life counters is gone the player will get a game over screen and the game reset so the game can be played again.

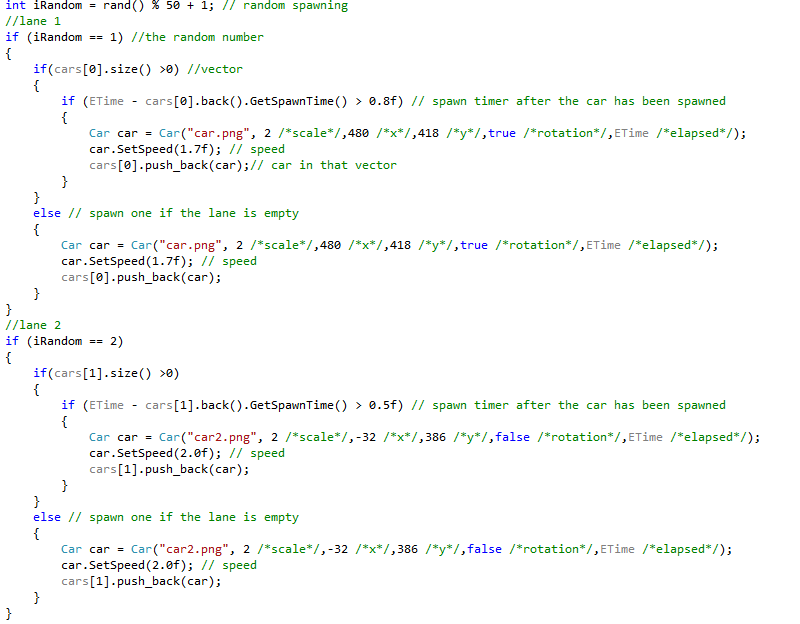
My program:

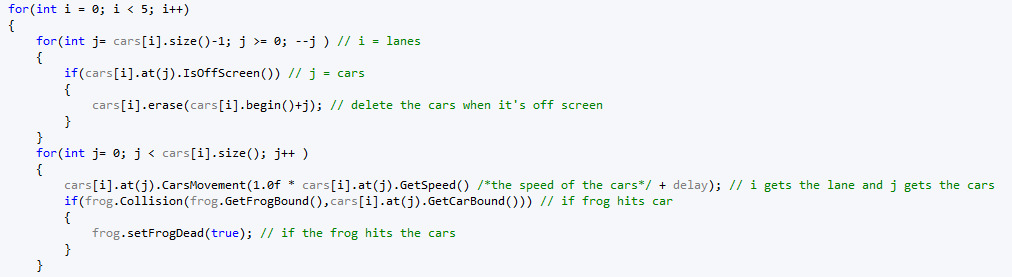
in my program I have used C++ in Visual Studio 2012 with the SFML libraries to create this Frogger game that I had made. I have made the frog movement first, this is because the frog is the most important part of the game and the game revolves around the frog. I map the movement key to the keyboard so I can move the frog and then I added the sprite for the frog after the movement has been detected.

I have then made the background suitable for the game and set it as a single sprite that goes on the back of the game and the background sprite is a rectangle shape. The background is important because it sets out the position of the frog and the other things to come such as the positions of the cars.

After the background and the frog has been implemented, the coding here became more complex. After drawing the frog and lanes which I made an update and render function in my main.cpp, I started making one car and experimenting with the size, scale and the position of the car. I have found a way to spawn multiple car at once by making a vector that contains more than one car.

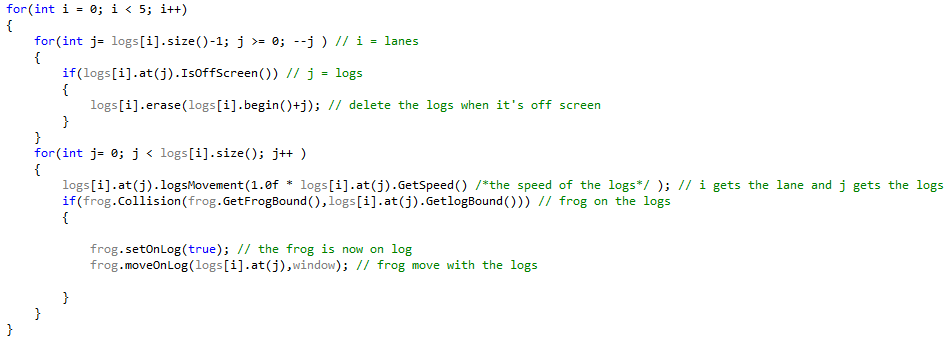
Making the cars spawn and move was the next after I had created the vector. I have made it so that the speed and spawning are all different, so the player will encounter different patterns in the cars. There is also a random number that is generated therefore the cars is also spawning randomly. The cars have also been put into groups, left and right. By putting the cars into group the game becomes much more enjoyable because the cars doesn’t just go in the same direction.

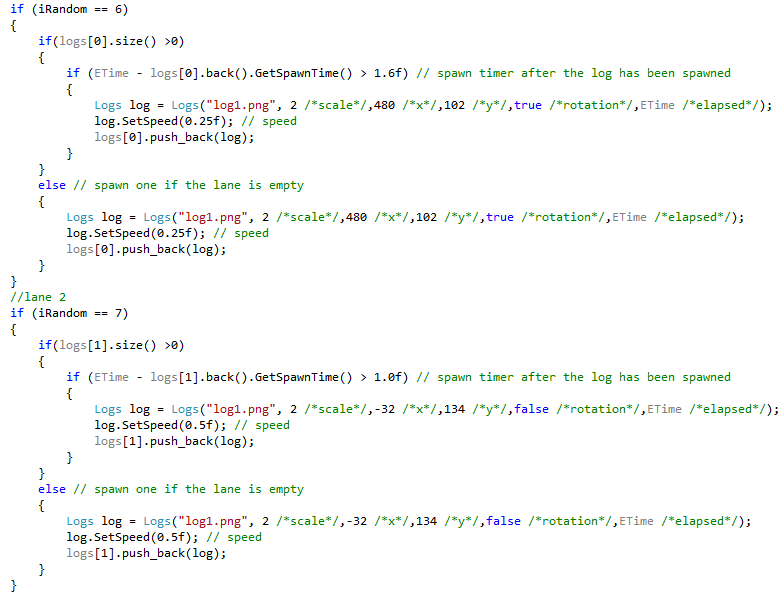
In my cars header file I have get and set speed and this let’s my set my speed in my main update, this also let’s my passed the function as well. Throughout my code I have use get and set as a way to pass the function from my header files to my cpp files, I feel like this is an easy way to passed a parameter and also set it in my main update section.

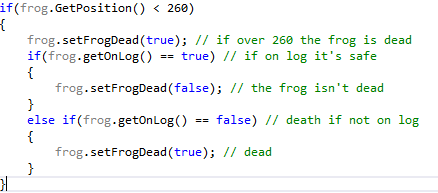
Now that the cars are moving I have made the bounding boxes for both my car and frog, and when they touch any part of this bounding box the frog should die. So first I made the cars have a bounding box and then my frog, and in my update function I have created for loops and if statements to see if they would touch each other and check for all of the cars in the vector as well. This also shows the cars are also getting deleted when it is off screen so that the game can ran smoother and it doesn’t overload the program.

After I had made the cars and my frog can die, I have made the logs. The log was simple to make this is because the mechanic of the logs are almost the same as the car, this means I would make my logs the same way as my car. I first created a vector for the logs just like for the cars. The logs, however have different effect in collision with the frog, cars would kill the frog if they have collision with each other but for logs, and the frog would just sit on top of the log and move along with the log but doesn’t kill the frog. The logs would carry the frog off screen which would then kill the frog that way however the water under the log would kill the frog if he isn’t on the log. I have made up some simple and effective code for the collision with the water.

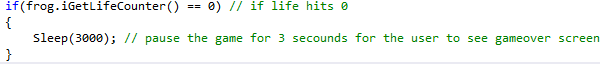
Vector for the logs.

Deleting logs when they are off screen, and create a Boolean and see if the frog is on the log and then the frog and can move with the log when it’s on the log.

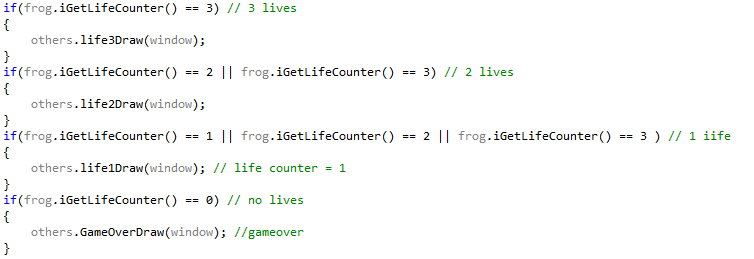
The main mechanic for the logs such as speed and spawning time.

Water collision with the frog will kill the frog, I have a simple code that would do that but I just have to change that when the frog is on the log the frog is safe.

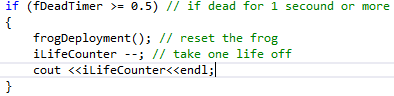
At this point the main game is done, however it still need a winning and losing condition. I have made a life counter that lets the player have 3 lives and this is a simple integer counter system that I have made, I have used a sprite as actual counters the frog’s lives. When the life counter reaches zero the game stops and the game over icon shows up to tell the player that they have lost, after 3 second the icon would go and the game will restart. When the frog reaches the base an icon would replace the base therefore this shows the player that there is a frog in the base already, if the player still chooses to go into the same base then the frog would die. When all the frogs are in the base, the game will start a new level without the frogs in the base.

Having a sleep function that pauses the game so that the player knows they have lost the game.

The life counter is just the same sprite however they disappear at different time to create player losing a life visual and when the play reaches 0 display the game over sprite.

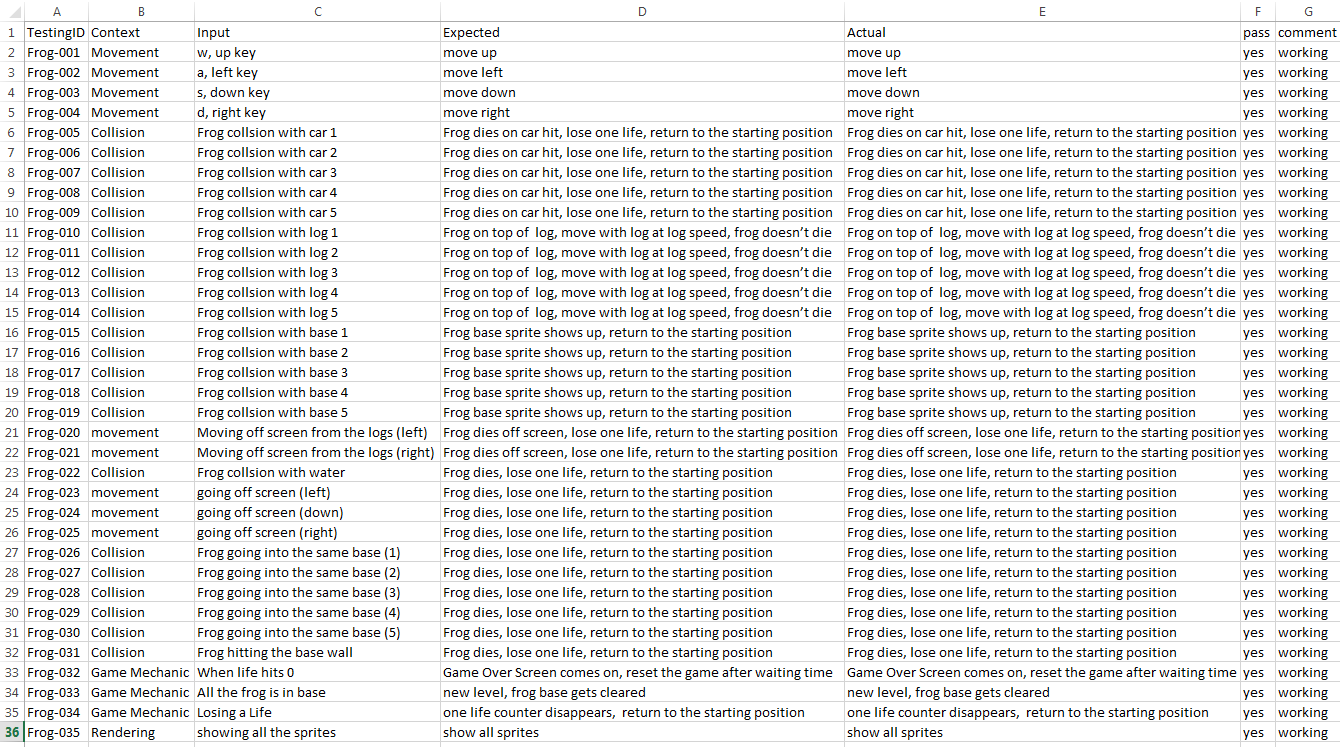
When the game is won reset the frog in the base.

Testing:

Testing is one of the most important stages of making game, I have done testing throughout the project. Every important code I put in the program I would run the program and test out the part I have just put in, this would reduce the error on that part later on in the programming stage and save me some time later. I have tested my game a lot more than just once, for example the frog movement is one of the easiest to test and would need that to work in order to test out other things should as log collision. Another thing I have used in the past is using ‘cout’ to see if my code is going into if statements or if they are doing what they should, such as the life counter.

I have also made latest version of a testing excel document. I have tested most of my important codes to test if there are any errors throughout my work. The test that I have made is black box testing, black box testing will tell me if the code works or not. For easy testing sometimes I have disable some of my code to get to a point to test the code that is being tested faster and easier to test it.

Here is my testing table:



The test was a success and my game is working as its intended, however I may have not tested some of the minor issues. I fix most of the issues I have come across however I have fixed some of them when I was doing the testing document. I have now fixed all of the issue with the game. Now that I have tested my game I would say it’s all working and there isn’t any errors.

Concussion:

Reflecting on my own work:

I would like to believe that my work is up to a high standard, however I could have done a lot more with the game mechanics such as timer and score. I had to pay attention to details such as having life counters instead of displaying text for it. I think that the game is still fun to play the game and I would play it in the future. During this assessment I have learnt valuable skill and increased my knowledge in the SFML libraries, I have learnt that managing my time is very important, this is a skill often overlooked, but if I would have managed my time better I would have had a better game. My coding skill have improved throughout this project because I would have to self-taught by using the internet and I have ask peers for better ways to improved my code, which will make me do my future projects better and complete the task sooner. I have made my code neat and well commented throughout this is because coding a game of this size can make you forget what you were working on last time. I have tested my game throughout this project very frequently, this is to avoid disaster in the future when the game wasn’t tested to a high standard.

If I could do this work again? :

If I can do this work again I would have included time and score. This is the main thing I would add if I can do this again, this is because the game wasn’t as competitive other games and I liked it to be. If I could do this again I would also add an animation to the frog, this couldn’t be added in my game due to the time constrain of the project. If I do this in the future I would be able to finish this faster with more mechanics can be added to the game. I feel like the work I had made this time around isn’t bad at all and I would build my game like this but with better content. Sound effect is another element I would implement into the game to make this game a lot more interested.