Jolie Khaled - 900211816

Janna Elfeky - 900215017

Mark Garas - 900215330

**Project Report**

Today marks the 8th of November, and we started working on the project. The first thing we did was reading all the instructions carefully for the milestone and brainstorm all the possible ideas we could do. After well understanding the tasks, we decided to start doing the board. Janna and Jolie started to draw it, and Mark, at the same time, started to search for images that we would need. The board will be, of course, 2d arrays, so we have drawn on paper ten rows and ten columns, and we have put the four bullets at the corner of the board, Franklin in the middle of the board, and finally, the two enemies in the previous last rows. We thought about how to arrange the board in many ways, but we decided that the board's borders would contain some obstacles and grass. We imagined that the obstacles could be buildings, cars, trees, street lamps, and of course, in the middle of the board, there would be the street and also some obstacles. After we had drawn the board map on paper, we started to copy it into a file text. Mark has found 2d images of Franklin, Franklin's home, the two enemies, all these obstacles, grass, lamps, bullets, and the street.

On the 10th of November, we started to write the code. We have set the size of the board as 600, and we have chosen that the background color will be black. For every obstacle, we have set the width, and the height will be 50. For the board, we have characters, and every character stands for a certain obstacle, grass, or road. For example, T stands for tree, C stands for car, and so on. Then we started to read from this file that contains the board. We did a double for loop that checks each number and inserts its image. After that, we ran to check that everything was going well so far. Unfortunately, nothing from the board had appeared; it was just a black background. We became very disappointed and tried to know where the problem came from. We thought that maybe the problem was that our board's body was characters instead of integers, so we decided to change all the characters to integer numbers. So we started to change the board and the code too. But nothing changed; a black screen appeared too. We tried for many hours to find out what the problem was, but we didn't find any solution.

On the 13th of November, we again tried to fix this problem; we were very confused about what we had done wrong. We also looked at the code of the Tweety game as a reference, and we found that we almost did the same thing, which increased our confusion. After spending a lot of time on this, we decided to split the tasks. Janna continued to try to fix the error, and Mark and Jolie started to think about the classes. We decided that we would have 4 classes: Franklin, Enemies, Bullet, and Power. We started with franklin, we set many private data members such as lives, rows, and columns, and for the functions, we did one for the press event. The same thing happened to Enemies. While writing the classes, Janna decided to close the old project and open a new one, and then she copied the same code, and actually, it worked. The images started to appear. So it was not totally a problem with our code, but QT sometimes cracks. However, the board was not complete, and there was a problem with the patterns. There were some obstacles that had not appeared, and some of them were placed in the wrong place. For example, we set in the board that Franklin's house was in the middle of the board, but in the output, it was placed on the right down.

On the 15th of November, we started to think about the last issue we faced. We thought that it might be because it could not read from the file. So Janna has suggested that we can change all the boards to zeros and ones, and then we can cout what is inside the board in the double for loop where it reads from. When we did this, the output was all zeros; it did not read the ones. So at least, we knew where the problem was. We concluded that it does not read the images appropriately, which is why they were not inserted. The images were already in the form of ".png", and we already inserted it with its path, not the name. So we thought maybe we needed to convert them into ".txt" to solve the problem. We did this, and it actually worked; the images were all inserted in the right place.

On the 16th of November, after the board was done successfully, we started to work on Franklin and the enemies. In Franklin's constructor, we have set its image, giving him a life of 3 and a row and column of 5. In the function of the keypress that will allow Franklin to move, we made him move it if it's only 2, which stands for road. So he can't move if the next row or column is anything other than the road. In the case of a colliding item, we said that if he collides with the bullet, the bullet will be removed, and Franklin will be transformed into another image where he will hold like a gun in his hand. We started to make a timer for one second after he got the bullet, but we didn't finish it. Also, in case he collides with the power pellet. We said that if he did not have a power pellet and collided with the enemy, his life would be decreased by 1. For the enemy's class, we thought at first that we would do 2 classes for the enemies, as we would have 2 enemies, but then we decided to do just one and, in the main, create 2 objects of type enemy. In the enemy's constructor, we have inserted an image and set his life to 2. We also did a function that would permit them to move. In this function, we have assigned that their rows and columns are randomly assigned. However, we thought that we must put a condition so that the enemy continues to move through the game round, so we decided that if Franklin's life is not equal to 0, which means that he did not die and the game is working, he must continue to move randomly. We, of course, did a class for bullets and power pellets and inserted their images.

On the 17th of November, after making sure that all that we did would make Franklin and the enemies appear and move, we went to the main function and started to create objects of the classes. We started by creating an object of class Franklin and 4 objects of type Bullet as we will have 4 bullets. The bullets must be in the corners of the board, so while creating the objects, we have set that their rows and columns must equal the right and left top and right and left bottom rows and columns. Also, we made sure that the bullets must be on the road because we already set in Franklin's move function that he only moves if it's a road, so that means that the bullets must be on the road in order to be able to take it. We have created 2 objects of type enemy; they will randomly move, but we have set their starting rows and columns to be in the middle bottom, so we passed in the constructor their rows and column numbers. Enemy's objects have then called the function that made the enemy move, and we have passed to it Franklin's life to fulfill the condition we have put. Finally, we decided to run, expecting that at least Franklin and the enemies would appear and move. After we ran, there were no errors, but it was written that there was a problem with the linker. We did not know what to do, so we closed and opened the app, then it worked. Franklin and the enemies have appeared in the right places, but none of them have moved. We started to think about what could be the problem; we thought that if Franklin did not move, that meant that the function responsible for his movement did not work. So we began to trace it and compare it to the tweety function. However, we have noticed that we already did not call the function in the main, so how will it work? When we asked our TA, Bassiouny, he said that we must search for how to call a slot function that will be in the constructor.

On the 18th of November, we decided to do 3 main things: search how to call a slot function, know how to do a timer so when Franklin takes a power pellet for 10 seconds, he can't be killed by the enemy even if he hit him, and timer for the bullet that after he takes it, it changes the image of him holding a gun for 1 second, and finally search for how to create widgets that show Franklin and enemies lives and the timers. We started by searching for the function, and we found some good information, but Franklin still did not move. We spent many hours trying many things we found, but nothing worked. We decided to divide the 3 tasks among us for the sake of time. For the widgets, we first thought we could create on the ui window all the widgets we need, so Jana started to implement this idea. Jolie started to search for the timer idea, and Mark continued to see the problem with the slot function. When we saw the timer's code, there were many problems with it; for example, the connect function had a lot of errors, so we fixed it. For the widgets ideas, we did not know how to connect the ui window with the board. We thought about how the labels could appear on the board, but we did not arrive at an answer, so we decided to leave it at the end. For Franklin's movement, we did not notice any errors again, and we really came disappointed because we were sure that everything we wrote was right.

On the 19th of November, we started to work on how the closest enemy should lose a life when Franklin takes a power pellet. So we started to calculate the distance between enemy1 and Franklin, then enemy2 and Franklin. Then we compared these 2 distances; the smaller distance means that we should decrease the life of its enemy. We also change the image of Franklin when he takes the power pellet to Hulk, which represents power. We felt that we added many things that need to be tested, but in order to test them, Franklin at least should move. We decided to create a new project and implement a code that only makes Franklin move, regardless of any obstacle or anything. So we started from the beginning, we wrote line by line the code again, and actually, Franklin moved. We were happy but, at the same time, very confused because we were the same people who wrote the first one and literally the same code. The next step we did was that we compared the old to the new code, and we found a silly small mistake in the keypress function, which was responsible for Franklin's movement. We already named our board "border" which has 10 rows and columns; in the if condition of this function, we misplaced the word border with data that is already declared in the main. So it didn't give me any syntax error; it was an unknown logical error that we never thought of because we forgot that our board's name is border, not data.

On the 20th of November, we got enthusiastic that Franklin had finally moved, and it was time to move the enemies. It took us so much time to move them. We first assigned that enemy's rows, and columns randomly moved. However, when we ran, it started to move, but it moved not step by step, but it kind of jumped many steps. Then, we decided to put them in a function similar to the function keypress of class Franklin. So we decrease or increase the row and column by 1 depending on what we want them to move either to the right, left, up, or down. Then, we started to work on how the game could end. We decided that when Franklin's image is equal to 0 because he passed through the enemy 3 times and didn't have a pullet, we will show an image saying" Game Over" and the tab will close. If Franklin's life is not equal to 0 and had taken all the 4 bullets, this means that he won. In this case, we made that Franklin's home in the middle of the board will be replaced by a door where Franklin must go at the end, and finally, an image appears saying, "You Win". So simply, when bullets are equal to 0, this image will appear. When we decided to run, we faced a problem: when Franklin takes a power pellet and comes across the enemy 3 times, he doesn't die. We tried to search for the error; then, we found that we had put nested loops, but we didn't close or put braces for the first one, so the loop didn't work correctly.

Today, the 21st of November marks the last day to work on milestone 1. We started to work on some logical errors we had and then thought about the widget that must appear to indicate the character's life and the timer. We faced a problem when Franklin took the power pellet and then came across the enemies; he died, although he should not have. Also, we have set that when he takes them, his image must be changed to hulk for 10 seconds, but it just lasts for approximately one second. For the timer problem, we used something called a single shot that has solved this problem, and we have sent to its parameters 10000, which means 10 seconds, and a function where we insert hulk's image. For the power pellet problem, we declared a data member with type boolean called powerful in the class, and we initialized it with false in the constructor but true in hulk's function. So if powerful is equal to false and then he comes across the enemy, the hulk's function will not be called and automatically will lose life; otherwise, powerful will be equal to true when he takes the pellet and will not lose any lives for 10 seconds. When we made these changes, it worked successfully. Franklin took the power pellet, transformed into hulk, and was not affected by any enemies, and after 10 seconds, its images returned to the normal Franklin, and he started to lose lives in case of any colliding with the enemies. Regarding the lives and timer, we knew that we did not need to create any real widgets on the ui window; we could just write what we what in coding lines. Mark stated to create these widgets, he used something called signal, slots, and connect function that helped to make the labels appear. Jana and Jolie worked on win and lose functions because the images we inserted did not appear when Franklin won or lost. Also they did a class for the door, set that the door’s image should appear in the place of Franklin’s home, and made that when enemies’s lives are equal to 0, a door image should appear, and then when Franklin goes to the door, an image saying “You win”appears”.

Finally, we were very proud of what we all did in the milestone; we thought that it was really considered a big achievement. Although overcoming the many errors we faced was very challenging, arriving at the right output made us forget all the rough times. Also, as we can see that we all did every task together; we only preferred to divide the tasks if we thought that we were running out of time. We felt that we would produce a more significant result if it was a collaboration more than individual work. We hope that milestone 2 goes as well as milestone 1.