1/2/2019

By: Dikshya Kafle, Elyse Isimbi and Christine Giwar

Teacher: Eric zhu

**MIRA-MIRA**

Project Report and User Manual ICS

Group Members: **Dikshya Kafle** (2018380039), **Elyse Isimbi** (2018380090) and **Christine Makera Giwar** (2018380042)

Title: Mira-Mira

Description:

This is a program created by Raptor Programming that allows the user to match the same images appearing in the window until all images have been matched. By the click of a button, two images will disappear as they are matched with the corresponding image in the window. We have chosen to include flags as our images to signify the abundance of cultures we are immersed in and the inclusivity that we portray amongst ourselves. The title Mira-Mira simply means Look in Spanish, further proving the openness to foreign identities.

Solution:

Our earliest hurdle was establishing the number of images to use in the game itself and working out the dimensions of the screen and what size each image had to be in order the fit the graph window. After strategic thinking and mathematical analysis was applied onto this crucial part of the game, we simply had to decide what our game needed to really look like, what message it should convey to our audience. Soon afterward was the logic behind the game. We asked questions like: *Since there are only 18 pairs of images, should we only have 18 possible scenarios/ decision statements? In what order will we place the boundaries of each image? And how exactly will an image disappear after it has been matched?* We had also encountered a problem as we desired to have background music installed in the program itself as the game was in session.

Results and Analysis:

As aforementioned, it was a matter of mathematical analysis in the end. It was agreed that 18 images would be utilised meaning 36 photos will be displayed in a 300x300 graph window. This meant that the grid would be made up of 6x6 images. Consequently, the dimensions would automatically mean 50x50 for 36 images to fit the desired window size (300x300).

Upon thoroughly assessing these questions we came to an agreement to have all the images’ boundaries listed in logical order starting from the origin (0,0) climbing up the y-axis. Now that all images would be listed, it was only a matter of linking the matching pair’s boundaries as the second set of conditions needed to eliminate both images. The elimination process was simply drawing a white-filled box to replace the recently matched flags. As hard as it was to assess a logic like that, we were surprised how easy to follow the actual logic is.

Soon after it was initialized that there simply could not be 18 decision statements or conditions for there were two ways to go about eliminating a pair of flags. Therefore, we had to have at least 36 conditions. Just like it was explained in the logic section above, the idea to have 36 conditions arose from thinking like the audience. Not every person would click the rightmost flag – or follow our hidden logical format- so 36 options are placed to accommodate the user’s freedom of selection.

As for the background music situation, we only had to change the name of the file to .wav but the file was not converted into a format that was compatible for Raptor to play it. Upon seeking advice from a colleague the music file was converted into the .wav format via an online converting tool. From there the Call command to play background music was added and the feature worked from then on. (Refer *to Special Features: Background Music below)*

Conclusion:

In conclusion this program opened our programming eyes a lot more as well as our capacity of strategic and logical thinking. We have enhanced and deepened our knowledge of Raptor as well as our groupmates’ understanding of Raptor and it is very satisfying knowing that we had managed to create a game this early in our university life. It has been established among all three of us that passion is one trait that will allow you do something and to do it very well. If you haven’t found a solution it doesn’t mean it isn’t there, you simply have to approach the problem differently. Somethings are not as hard as you think, you just have to think outside the box. Finally, this programming journey has taught us to have a great deal of patience.

Resources:

Textbook (Introduction to Computer Science, Xiuwei Zhang, Yanning Zhang), online music format converter, images on Google, colleagues

*Title: Mira-Mira*

*User Manual:*

* **Click “Start” to open the “Procedure” page. Read the procedure.**
* **Click “Next” when you have read the instructions of the game.**
* **The objective of the game is to click two of the same pictures until you have clicked all pairs of the images. Only then will the game end.**
* **You will see a congratulatory message once you have accomplished the objective.**

From the programmers, enjoy Mira-Mira!

***Home page:***

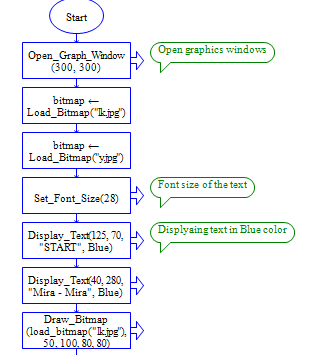
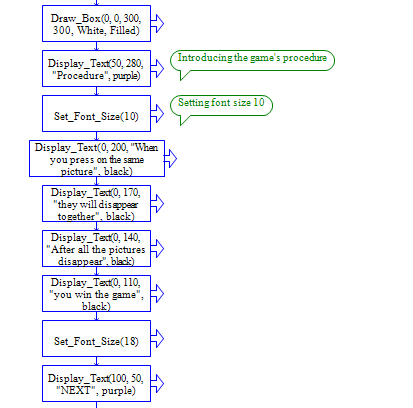
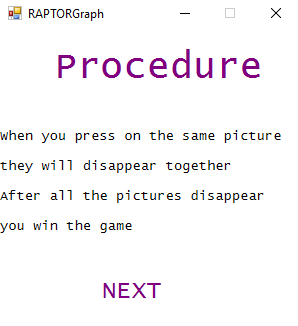


Figure 1: The Home page or the first screen displaying the game's name: Mira-Mira and the start option

Figure 2: Programmed commands of the Home page.

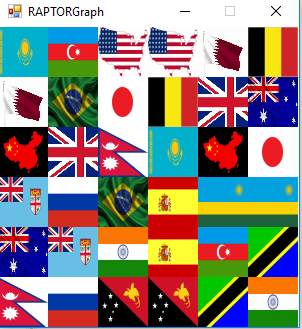
*Procedure:*

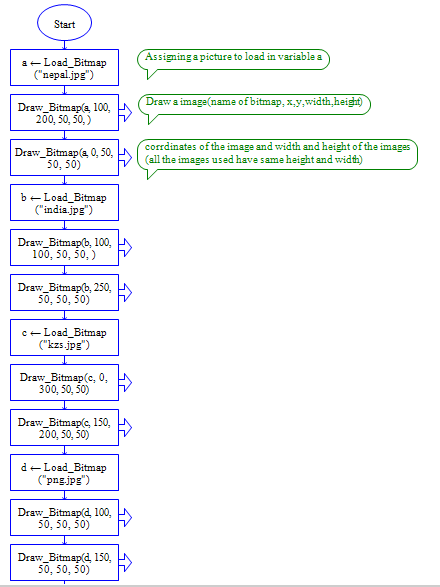




**Figure 3: The call commands in Raptor that will display the "Procedure Screen" that is seen on the right**

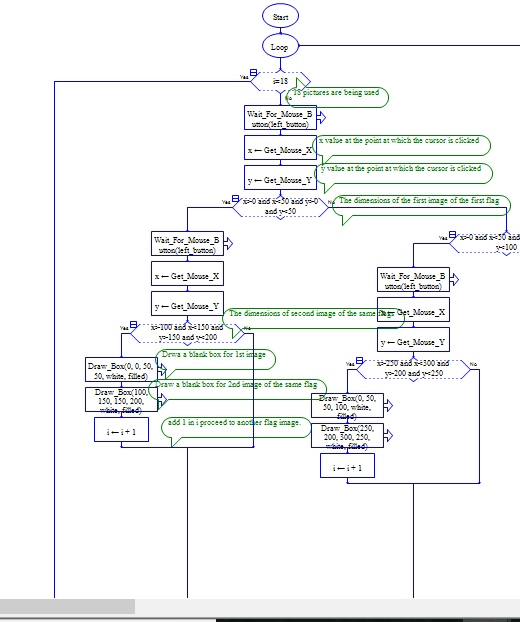
*Game screen:*

**



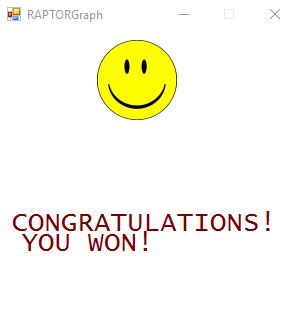
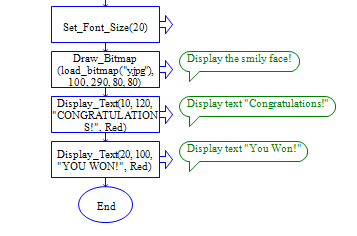
**Figure 4 (left): The subchart of all the images that is the " Game screen".**

Above: The game screen when the program is executed.

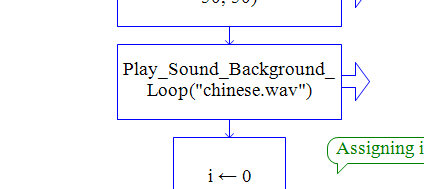
*LOGIC BEHIND THE GAME:*

The logic behind selecting the images and having them “disappear” as they go is as follows:

18 flags have been used in total so upon duplication there are 36 images in the game screen window, each image belonging to its own set of coordinates and boundaries. Since there are 18 pairs of images ( ***i***=18), the foremost decision statement is exactly that. Only after 18 pairs of images have been matched, will the game end. To get to that stage, the user will have to click on an image, more specifically, within the boundaries that the specific image is bounded to. For our own understanding we have placed the boundaries for each image in the topmost decision statements in logical order from the origin of the graph (0,0) up the y-axis. Since we ourselves knew the location of every matching pair, we simply made the second decision statement match the boundaries of that matching pair. This action was repeated to ensure all possibilities were considered for the selecting process relied solely on the user and not on one assumption. Therefore, the same action of matching the second pair of the flag is present but swapped further down the line of decision statements. It is important to note that there are at least 36 different possibilities present in the program but upon execution only 18 will be analysed depending on which matching pair the user selects first, hence the ***i*** variable being only 18.

*Last page:*

**Figure 5: The screen displayed once the aim of the game is reached.**

*Special Features:*

*Background Music*

**Figure 6: The special feature includes background music**