Authors: Linh Do, Na Nguyen and Tam Nguyen

Project name: Click-Drag-Drop

Date: Dec 14 2022

**Final Project Report**

1. **User’s Manual**

Ngon is an interactive, multi-level game that invites users to play and learn more about Vietnamese cuisine at the same time. Using Tkinter to create its Graphic User Interface program, Ngon offers users a tailored experience by letting them create their own user name and choose their own level. A unique Vietnamese dish unraveled to users at every level, Ngon gives users information about the dish (its ingredients as well as its historical background) before taking them to the game screen, in which they must click on the correct icons of ingredients among many flying across the screen to win.

By opening the “” file and running it on Python, you will start the game. Every instruction on how to continue (“Next” button), choose (“Easy",”Medium”,”Hard” buttons), and quit (“Quit” button) the game is written on the screen, allowing for easy user access.

Once the game starts, you would be asked to enter your name. By clicking the “Next” button one canvas after another, you will be instructed on the main objectives of the game. After hitting “Next”, if you choose the “Easy” level, the background information of “bánh cuốn" will appear, introducing its origins as well as how it is eaten. Then, the Next button will bring you to the next phase of the game in which you can familiarize yourself with the main ingredients of bánh cuốn which you will have to remember and choose correctly afterwards. Finally, after you're ready to play and hit “Next", you will play the game by clicking on the correct flying ingredients to have it automatically added to the basket. If you click on the wrong ingredient, you will automatically lose and be carried to another screen where you can either choose to “Play again" or “Quit”. If your basket successfully has all the correct ingredients, the game will end with a congratulations message. You can also choose to “Play again" or “Quit” the game. Hitting the “Play again” button will bring you back to the screen with the Level menu, while quitting will close the game completely.

1. **Program’s Contents**

*2.a. Function summary*

Our program consists of 15 functions in total, of which:

* 1 is the init function that sets up the basics.
* 9 are functions that create canvases holding information like images, lists, dictionaries, greetings, food introduction, gameplay instructions, update message when user clicks “medium” or ‘hard’ button, and win/lose canvases to notify user whether they have won the game or not.
* 3 are helper functions:
  + moveBall: to move balls in random directions and make them bounces off the edges
  + chooseBall: to program to recognize which ball has been selected by the user and add the selected ball into a basket
  + basketCheck: to check for the correct list of ingredients. Once the user clicks on an incorrect ingredient, it returns False. On the other hand, if the user manages to select only the correct ingredients and the length of the basket list (from the chooseBall function) equals the fixed length of the correct list (defined resetEasy3), then it returns True (meaning user has selected all the right ingredients)
* Go function: to update the program based on the player’s choice
* Stop function: to stop the program when the user wants to end the program.

Our program also consists of multiple interface designs including background canvas, “cow”, food, ingredients, texts which are created, selected, refined and designed by ourselves. These designs are stored in a folder, on the same level with the code file, so they can be easily called when needed.

*2.b. Implementation*

* We started off by creating canvas creating functions and played around with them to ensure they were called in our desired order
* Then we added the moveBall and chooseBall functions by modifying them slightly to fit the intentions of our project. For example, in the chooseBall function, we changed it so that when a free-flying object is clicked, it automatically moves to the center of the basket image we created prior.
* After that we embarked on building the basketCheck function to compare the real time userBasket object with a pre-determined correct ingredient list.
* As a next and final step, we added the basketCheck to the go function and changed up the go function to display the win canvas if the basket is correct or lose canvas if the basket is wrong.

2.c. *Testing*

* Buttons: we made sure when we clicked the “next button” it would lead us to the next canvas exactly as intended. When it didn’t, we checked back at the ‘command’ attribute of tkinter buttons and ensured the right function was called. Therefore, the main testing part was just us trying the program over and over again.
* For a more complicated group of functions, such as the gameplay (moveBalls, chooseBalls, basketCheck,..) we tested it on a different file first before incorporating it into our main program. We will attach a separate changingCanvases file and A basketCheck function file to demonstrate our point.

1. **Conclusions**

We learn that we can use different functions to create multiple scenarios depending on users’ choices. Specifically, we learned to how to empty our lists and dictionaries when switching levels to avoid errors. The most interesting part was creating the basketCheck function and how to let it know what level we are playing.

The flow of canvas (or the flow of gameplay) works as we expected, that is every different choice the user makes, it results as predicted in our plan. The surprising thing is we have to try a lot of different methods to compare the user list with the correct list before we get the result we want. Right now, because of the placement of the user/correct list, we have been only able to do one level that includes everything we intended a level to have. Even though we have successfully created the basic gameplay for the ‘Medium’ level but did not have enough time to create its introduction or think about how harder this level should be compared to the easy one. We, sadly, were not able to complete the contents for the ‘Hard’ level at all and so we added a “Sorry” canvas in place of the intended contents and a ‘Next’ button that takes the user back to the ‘levels’ canvas.

As future extensions, there is a couple of things we would like to accomplish. First, we want to really think about how to differentiate components of the game, like the speed at which different balls move, or the location at which each ball is selected to make the game more interesting as the user moves up one level. Second, we would have loved to finish the ‘Medium’ and ‘Hard’ levels thoroughly, add countdown clock to establish a time stress for the gameplay as well as background music (e.g during the gameplay, when the user wins or loses) so as to improve the user’s overall experience with our program.

**Bibliography**

Ernst, E. (2022). *The tkinter module for Graphical User Interfaces (GUIs)*. Https://Docs.google.com/Document/D/1S6xWzzMEF30FI0WBbxxZTI4m335kiA\_wEVvuLztkNfE/Edit#Heading=H.el7lsoxdwyyd. https://docs.google.com/document/d/1S6xWzzMEF30FI0WBbxxZTI4m335kiA\_wEVvuLztkNfE/edit#heading=h.el7lsoxdwyyd

Miller, B., Ranum, D., Elkner, J., Wentworth, P., Downey, A., Meyers, C., & Mitchell, D. (2013). *How to Think like a Computer Scientist: Interactive Edition*. Runestone.academy. https://runestone.academy/ns/books/published/thinkcspy/index.html