CSC1024 Programming Principles

Programming Project: A Master Mind Computer Game

Yohen Sheraun A/L Regu

18060285

3 December 2021

Presentation Video YouTube link:

https://youtu.be/dPArfubVVes

Display Data

Display data is used in this program to communicate with the player

It is used to:

1. Display game instructions to the player
2. Display the colours the user can choose from
3. Let the player know if he or she has won the game, or if the player must try again with certain hints to improve the players chance of wining

Data is displayed to the user via the print function

```
print()
print("Welcome, you are currently playing MASTERMIND!")
print("In order to play this game you are required to select four colours from the given choices "
      "in order to guess the colours selected by the computer")
print(f"Your choices are {colour_choices}")
print("Please enter your input in the form: cyan purple blue red")
print("You can enter the same colour multiple times as well")
print("Good luck and have fun!\n")
print("What are your guesses?")

print("Oops! You have guessed too little, please select four guesses")

print("Oops! You have guessed too much, please select four guesses")

print(f"Oops! You have made an invalid choice '{guess}' please choose valid colours from the choices "
      f"provided")

print(f"Congratulations! You have won the game in your first guess! ")
print("You have won the title master guesser! :P")

print(f"Congratulations! You took {number_of_guesses} guesses to win the game, well done!")

print("Aww looks like you did not win the game!")
print(f"Guesses that are of the correct colour correct place: {colours_in_correct_position}, "
      f"correct colour wrong place: {colours_in_wrong_position}")
print("Please try again!")
```

Input

- The input function prompts the player to enter his/her guess of the colours that was selected by the computer (opponent)
- The input is then run through the lower function to convert the players guess to lower case characters
- .split function splits the players input by whitespaces

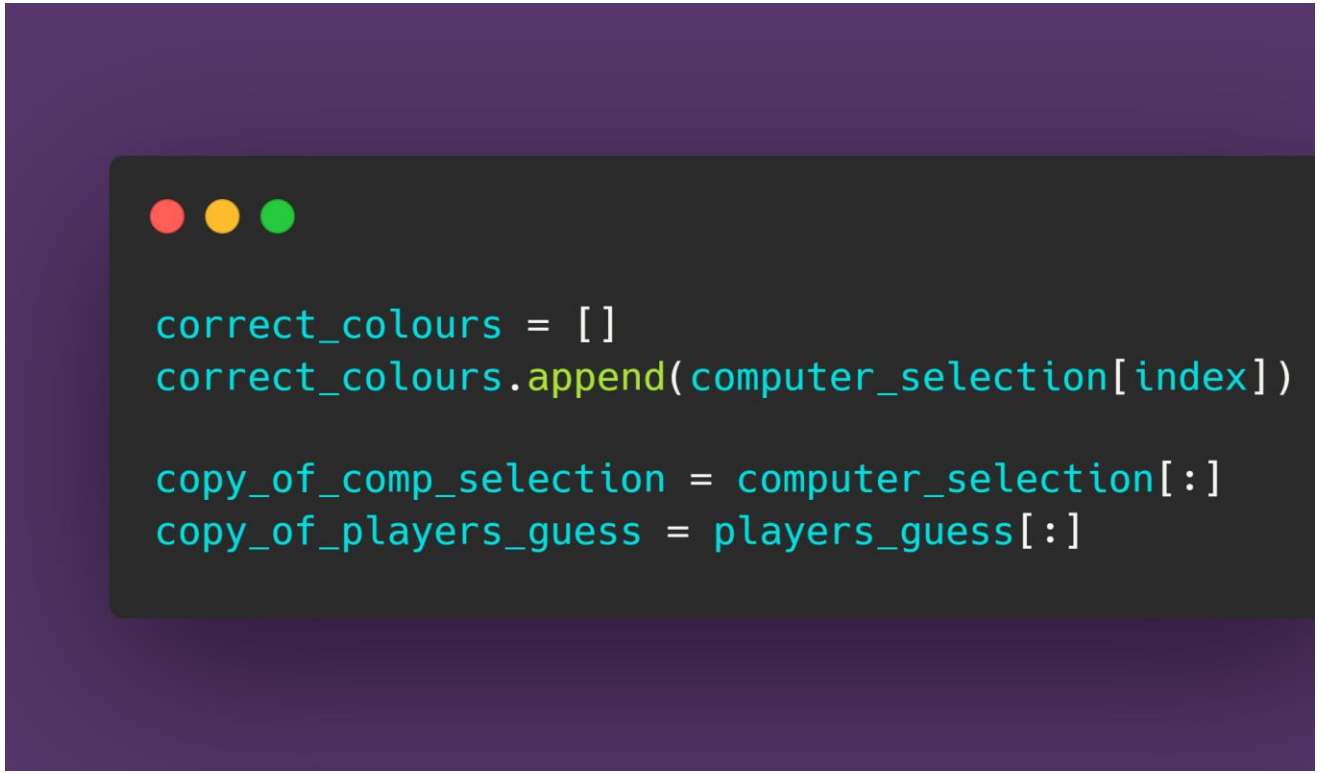
Lists

Data structures used to store colour choices, the computers selection of colours, the colours guessed by the player and colours that are in the correct spot

- Colour choices is the options of colours that can be selected by the player and computer
- Picked colours is a list generated by randomly selecting colours from colour choices (represent computers selection)
- Player's guess would be split into a list by split function

The reason for using the list data structure was due to it being easier to compare the players guess and computers selection

Continuation of lists



```
correct_colours = []
correct_colours.append(computer_selection[index])

copy_of_comp_selection = computer_selection[:]
copy_of_players_guess = players_guess[:]
```

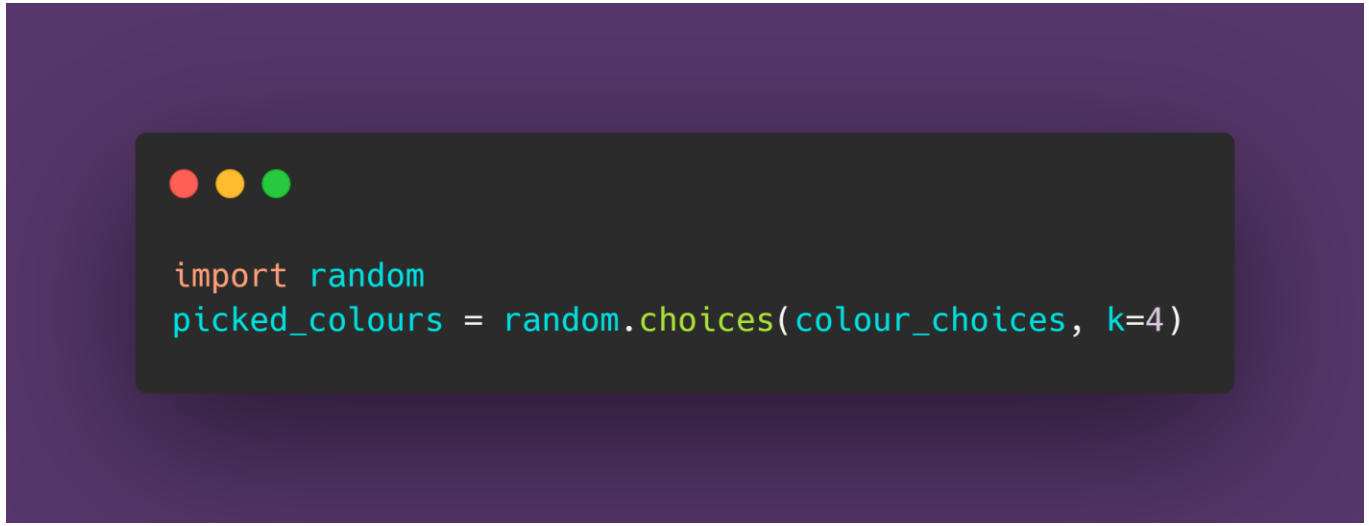
- Correct colours stores colours that were correctly guessed
- A copy of players guess, and computer selection was made in order to prevent modification of original list

Random choice from a list

In order to generate a unique list every time the program was run

The function .choices from the random library was used to select four colours from colour choices at random.

The function takes an additional argument "k" which defines the length of the list returned by the .choices function



If statement, and relational and logical operators

The first two if statements in the program checks if the length players guess is less than or greater than four with the help of a relational operator. If true, the player has a made an invalid guess and the player must the re-enter a valid guess

```
if len(players_guess) < 4:
        print("Oops! You have guessed too little, please select four guesses")
        return False

elif len(players_guess) > 4:
        print("Oops! You have guessed too much, please select four guesses")
        return False
```

```
colour_choices = ["green", "cyan", "purple", "blue", "red", "orange"]
```

The third if statement is used in conjuction with the logical operator not. This statement finds if the players guess has a colour not in the colour choices and returns a False value asking the player to try again

```
if guess not in colour_choices:
    print(f"Oops! You have made an invalid choice '{guess}' please choose valid colours from"
          f"the choices provided")
    return False
```

The fourth if statement calls the function input checker to check the validity of player guess. Breaks the loop if valid.

```
if input_checker(players_guess):
  break
```

The fifth and sixth if statement (the sixth being a nested if statement) both use the equality operator.

The first checks if the player has won the game and enters the block to the following nested if statement which checks if the player has won the game on his first try displaying a unique congratulations message. If the player won but not on his/her first try, he/she would get a standard message.

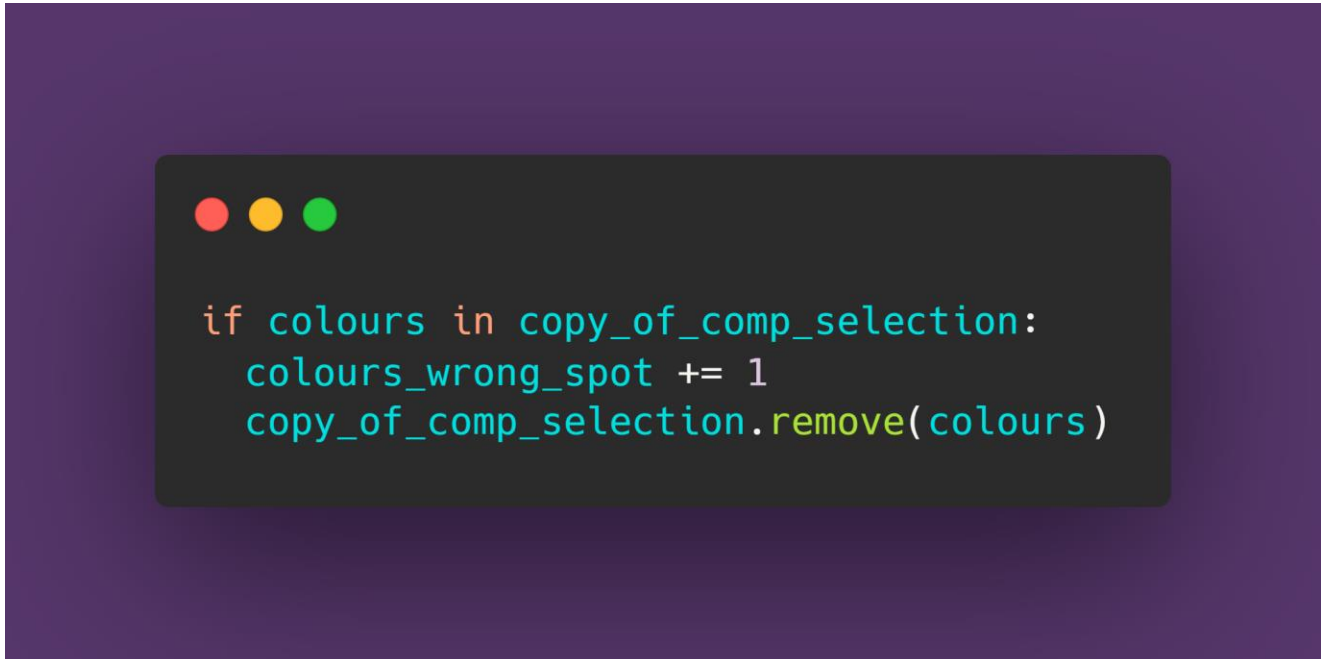
If the player had not won the game the else condition is executed, and the player is prompted to try again

```
if colours_in_correct_position == 4:

    if number_of_guesses == 1:
        print(f"Congratulations! You have won the game in your first guess! ")
        print("You have won the title master guesser! :P")
        player_won = True
    else:
        print(f"Congratulations! You took {number_of_guesses} guesses to win the game, well
              done!")
        player_won = True

else:
    print("Aww looks like you did not win the game!")
    print(f"Guesses that are of the correct colour correct place: "
          f"{colours_in_correct_position}, "
          f"correct colour wrong place: {colours_in_wrong_position}")
    print("Please try again!")
```

The if statement to the right is used to check if the colour guessed is in the correct spot by using an index that points to a colour at the same index in both lists.

```
if players_guess[index] == computer_selection[index]:
    count += 1
    correct_colours.append(computer_selection[index])
```

- The last if statement is used to check if the colour guessed by the user is in computer selection meaning the player guessed the correct colour but got its position wrong



Loops

```
    for guess in players_guess:
        if guess not in colour_choices:
            print(f"Oops! You have made an invalid choice '{guess}' please choose valid colours from "
                  "he choices provided")
            return False
```

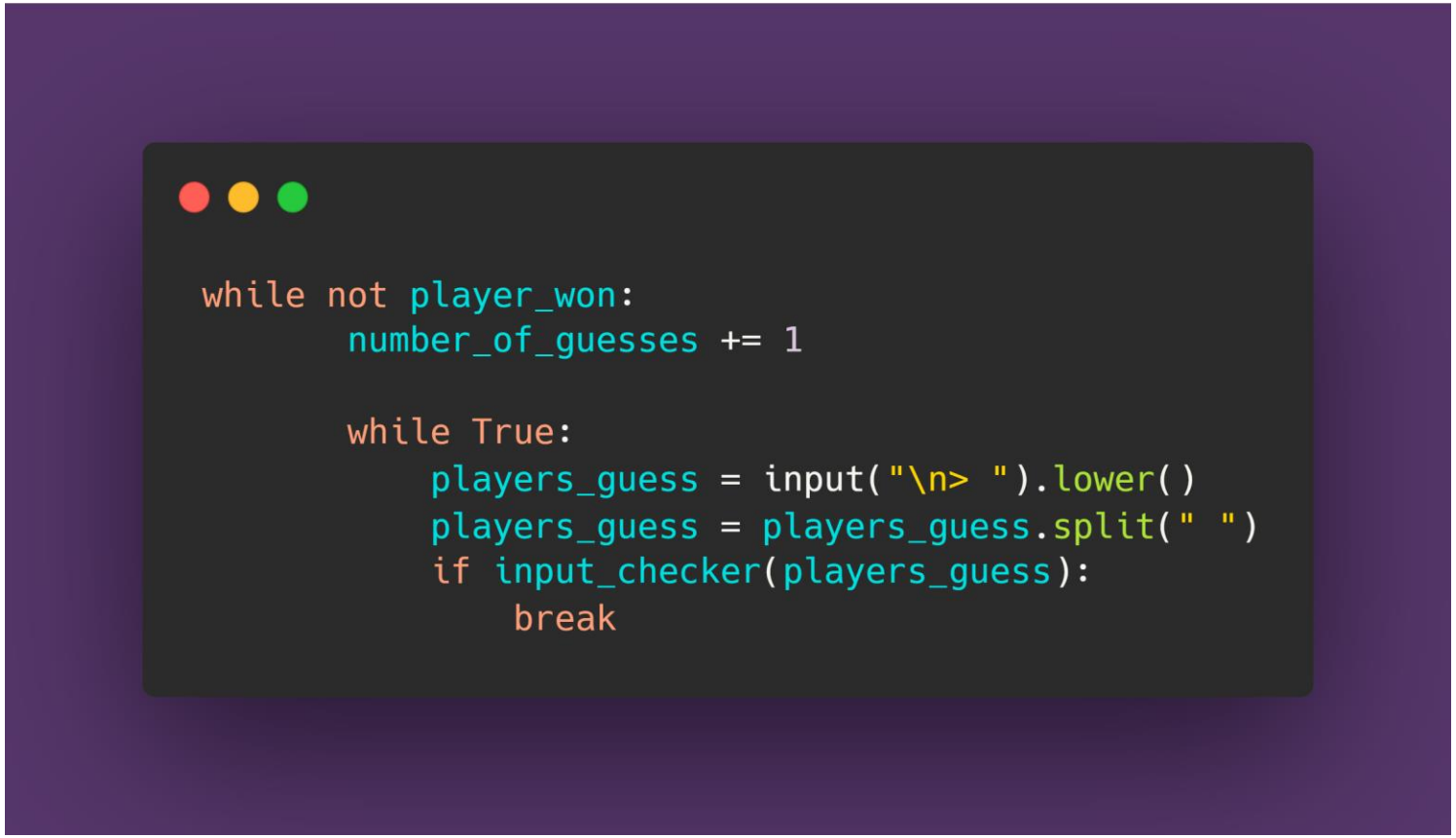
- The for loop above is used for checking the validity of player input
- It loops through the list of players guess and checks if each colour is in the colour choices provided

Continuation of Loops

The for loop below is used to check if player has made a correct guess at the correct spot

The for loop iterates over the length of players guess and compares it with the computer's selection of same index

```
for index in range(len(players_guess)):
    if players_guess[index] == computer_selection[index]:
        count += 1
        correct_colours.append(computer_selection[index])
```



```
while not player_won:
        number_of_guesses += 1

        while True:
            players_guess = input("\n> ").lower()
            players_guess = players_guess.split(" ")
            if input_checker(players_guess):
                break
```

- The function guess checker uses nested while loops, the first of which causes the game to run till the player wins

- The nested while loop is used to make sure the player enters a valid input, if the player does not enter a valid input the loop is repeated indefinitely till the player enters said valid input.

User defined functions

- Functions that are not already defined by python and is created by the user to carry out specific tasks
- This program uses six user defined functions which are
  1. Colour generator
  2. Start menu
  3. Input checker
  4. Guess checker
  5. Correct spot
  6. Wrong spot

Thank you