CSC1024 PROGRAMMING PRINCIPLES
Programming Project: Mastermind Computer Game

Chiam Tow Wang
(Student ID: 19119023)
BSc (Hons) Software Engineering

15 July 2020

Presentation Video Weblink:
https://youtu.be/1n1XVuK4GIM

Mastermind game

What is Mastermind?

- Game to guess random sequence of colours
- Players can develop many skills spatial reasoning, logical reasoning, critical thinking and many more

Demonstration

Questions

How do you apply Lists in the Mastermind computer game?

```
#Initiate Variables
colourList = [ 'red' , 'blue' , 'green']    #List of colours
answerList = []                             #List of answer from user
```

List variables ‘colourList’, ‘answerList’ used to store data

What kind of computational problems where Lists are helpful for problem-solving?

- Store inputs
- Separate single input to multiple inputs in order using .split()
- Compare inputs whether it is in correct/same order
- Modify input by removing a specific element on the list or adding a new element into the list
- Screening of inputs

```
#Input validation and converting answer from str -> list
for i in answerList:
    if i.isdigit() == False  and i=='red' or i=='green' or i=='blue':
        strAnswerList.append(i)
    else:
        print("Input incorrect. Please check spelling and try again.")
        break
```

A section of the code used to screen user input using for loop

How do you generate randomise values in the Mastermind computer game?

- Import random library
- random.choice

```
import random
```

```
#Function to generate random colours and store into rColourList
def generateRandomColours(colourList):
    rColourList= []                         # Empty list of random colours
    for _i in range(puzzleLength):
        rColour = random.choice(colourList)
        rColourList.append(rColour)
    return rColourList
```

What kind of computational problems where randomisation is helpful for problem-solving?

- For variability of program so that the expected outcome is always different

How do you apply decision making in the Mastermind computer game?

- Using Boolean data type
- Using conditional statements like if, elif and else statements

Using Boolean:

```
while solved == False:
    userAnswer =getAnswer()                    #Function that gets user input
    guesses = countGuess(guesses)              #Function that keeps track the amount of guesses
    correctPosition = checkColours(answerList,userAnswer) #Function to analyse user input
    solved = isitSolved(correctPosition)       #Function to check whether the puzzle is solved
print(f"Congratulations! You have solved the puzzle in {guesses} steps.")
```

A Boolean variable ‘solved’ used to keep track whether the puzzle is solved or not to keep the iterative processes running

Using conditional statements

```
#function to check whether the user managed to guess the answer correctly
def isitSolved(correctPosition):
    if correctPosition == puzzleLength:
        return True
    else:
        return False
```

Conditional if, else used for decision making

Why do you need decision making in the Mastermind computer game?

- To filter inputs, so that user does not input anything invalid like:
  - Numbers
  - Misspelled colours e.g. gren
  - Too many inputs
- To compare the inputs to check whether the puzzle is solved or not
- To allow the program to re-execute the commands again if the puzzle is not yet solved

What kind of computational problems where decision making is helpful for problem-solving?

- Problems that can be broken down in to logical steps
- To enable separate different lines of code to be executed, for different scenarios
- Enables loops to be iterative until the desired outcome is produced

How do you apply the iterative process in the Mastermind computer game?

```
#Input validation and converting answer from str -> list
for i in answerList:
    if i.isdigit() == False  and i=='red' or i=='green' or i=='blue':
        validAnswerList.append(i)
    else:
        print("Input incorrect. Please check spelling and try again.")
        break
```

For loop used here as an iterative process to screen user's input

Why do you need to create an iterative process in the Mastermind computer game?

- To screen through lists
- To re-execute commands until the puzzle is solved using a while loop

What kind of computational problems where an iterative process is helpful for problem-solving?

- Processing a list
- Mathematical operation like multiply 2 to all elements of a list
- Re-prompt the user to input if inputs are invalid

How do you apply the user-defined function in the Mastermind computer game?

```
#Print instructions to teach users how to play
def setup():
    print("Welcome to Mastermind!")
    print(f"To win, guess the random generated sequence of {puzzleLength} colours in correct order")
    print("Give your answer comma-seperated, like the following:")
    print("red,yellow,green,blue,...")
```

```
#main function
def main():

    #Initiate Variables
    colourList = [ 'red' , 'blue' , 'green']    #List of colours
    answerList = []                             #List of answer from user
    guesses = 0                                 #Counter for guesses
    solved = False                              #Default state of the game
    answerList = generateRandomColours(colourList)    #Generate the answers

    #Sequences of functions for the game
    setup()                                     #Give instructions to teach users how to play
    while solved == False:
        userAnswer =getAnswer()                 #Function that gets user input
        guesses = countGuess(guesses)           #Function that keeps track the amount of guesses
        correctPosition = checkColours(answerList,userAnswer) #Function to analyse user input
        solved = isitSolved(correctPosition)    #Function to check whether the puzzle is solved
    print(f"Congratulations! You have solved the puzzle in {guesses} steps.")
```

What are the advantages of using subprogram in the Mastermind computer game?

- Efficient, by breaking up complex code to small chunks of problems, which can be solved by a team parallelly
- Modular coding which is easy to read, debug, and maintain
- Reusability of functions which shortens the code and minimize redundancy of code

Conclusion and Future Improvement:
What have you learned from the programming project?

- Apply Lists, decision making and iterative process into making a functional and interactive code
- Problem solving using conditional statements and Boolean data type
- Write user-defined functions to make the code modular so that it is easy to maintain and modify
- Debugging code to make sure the code runs and functions as expected as the specifications

What is the future improvement that you would like to carry out in the next level of designing the Mastermind computer game?

1. Interactive user interface with shapes and sounds
2. A main screen before the game to input names of players to keep track of high score
3. Include more colours to make the puzzle more complex

END