Chester Descallar

Unit 16: Procedural Programming

Assignment 1 – Task 2

Contents

[Introduction 2](#_Toc513709360)

[Main Menu Specification 3](#_Toc513709361)

[Higher or Lower Specification 4](#_Toc513709362)

[Maths Challenge Specification 4](#_Toc513709363)

[Word Scramble Specification 5](#_Toc513709364)

[Rock Paper Scissors Specification 6](#_Toc513709365)

[Pontoon Specification 7](#_Toc513709366)

[Screen Designs 8](#_Toc513709367)

[Main Menu 8](#_Toc513709368)

[Higher or Lower 8](#_Toc513709369)

[Maths Challenge 9](#_Toc513709370)

[Word Scramble 9](#_Toc513709371)

[Rock Paper Scissors 10](#_Toc513709372)

[Pontoon 11](#_Toc513709373)

[Data Dictionary 0](#_Toc513709374)

[Game Application Pseudocode 0](#_Toc513709375)

[Higher or Lower Functions 3](#_Toc513709376)

[Maths Challenge Function 4](#_Toc513709377)

[Word Scramble Function 4](#_Toc513709378)

[Rock Paper Scissors Function 5](#_Toc513709379)

[Pontoon Function 7](#_Toc513709380)

# Introduction

As a Junior Developer, I have been asked to design and develop game concepts that could be translated for use on a website, mobile application or an e-book reader. Thus, I have decided that my concept would be a game application that allows the user to play various different games; these games are higher or lower, maths challenge, word scramble, rock-paper-scissors, and pontoon.

All of these games would follow the real game concept of the same name against a computer, and they will be placed all in on application where the user can choose which game they want.

These are the concepts of each game:

1. *Higher or Lower*

Game generates a random number from pre-defined range (e.g. 1 to 20). The player has to see how many turns it takes to guess the right answer.

1. *Maths Challenge*

The game asks the user ten maths questions and the goal is to get as many correct answers as possible. To do this:

* The game generates two random numbers from a pre-defined range (e.g. 1 to 10)
* The numbers are then multiplied together to produce an answer
* The player then has to enter what they think is the right answer
* This process is repeated 10 times

1. *Word Scramble*

A pre-defined word is scrambled and the user has to guess what the word is. Again the aim is to guess it in as few turns as possible. A two player version could allow the other to input a word for the other player to guess.

1. *Rock, paper, scissors*

Based on the game of the same name – the player has to choose rock, paper or scissors and the computer then guesses what the user will pick. Rock beats scissors. Paper beats rock. Scissors beats paper.

1. *Pontoon (also known as twenty one)*

Initially, the player and computer are both dealt two cards. Card values are generated are between 1 to 13 with picture cards Jack (11)/Queen (12).King (13) having an assigned a value of 10). An Ace can be given the value of either 1 or 11 by the player (if a card with the value of 1 is dealt). If the player hasn’t scored at least 15 he or she must keep choosing cards until this is the case (same for the computer). Any score greater than 21 equals a loss (known as bust). If the player and computer have the same score at the end then the computer wins.

This game will be targeted for all genders and anyone with gaming experience (casual, hard-core) allowing for a big demographic to play the game. It will be targeted to audience to audience aged 8 and older as the games have very simple concepts which are quite easy to understand.

## Main Menu Specification

|  |  |  |
| --- | --- | --- |
| **Requirement No.** | **Requirement** | **Description** |
| 1 | Create an application that allows the user to play several games. | This game application must allow the user to play the games; higher or lower, maths challenge, word scramble, rock-paper-scissors, and pontoon. Just as they would in real life |
| 2 | Allow the user to input a username | The game application should ask the user what they want to be addressed by when playing the game. |
| 3 | Create a home page for the game | The home page is where the user will be first sent to upon opening the program and in here, they can choose the games that they want to play upon inputting the number that corresponds the game mode. |
| 4 | Allow the user to play higher or lower | Upon inputting the option to play higher or lower, the user should be allowed to play the game. |
| 5 | Allow the user to play the maths challenge | Upon inputting the option to play the maths challenge, the user should be allowed to play the game. |
| 6 | Allow the user to play word scramble | Upon inputting the option to play word scramble, the user should be allowed to play the game. |
| 7 | Allow the user to rock paper scissors | Upon inputting the option to play rock paper scissors, the user should be allowed to play the game. |
| 8 | Allow the user to play pontoon or 21 | Upon inputting the option to play pontoon, the user should be allowed to play the game. |
| 9 | Include a on screen help | The on screen help will be added in the main menu of the game and will be one of the options that the user can input. |
| 10 | Allow the user to exit the game | Upon inputting the option to exit the game, the game should be exited. |

## Higher or Lower Specification

|  |  |  |
| --- | --- | --- |
| **Requirement No.** | **Requirement** | **Description** |
| 4.1 | Create a higher or lower game | The game must allow the user to guess a number and the AI must tell the user if it is higher or lower until they get it correct. |
| 4.2 | Computer must generate a random number between 1 and 100 | The computer must generate a random number between 1 and 100, this will be the number that the user will guess. |
| 4.3 | Allow the user to guess the number | The game must allow the user to guess the number that the computer has generated. |
| 4.4 | Allow the user to guess several times | The game must allow the user to guess several times if they get the wrong answer |
| 4.5 | Game must tell the user if the number is higher or lower | The game must tell the user if their number is higher or lower depending on what they have inputted |
| 4.6 | Must tell the user how many guesses it took them | The game must display the amount of time the user has guessed to get the correct number. |
| 4.7 | Must ask the user if they want to play again | If the user wants to play again, the game will be restarted, however if they say No then they will redirected back to the main menu |

## Maths Challenge Specification

|  |  |  |
| --- | --- | --- |
| **Requirement No.** | **Requirement** | **Description** |
| 5.1 | Create a Maths challenge game | The game must output ten different maths question to the user with the goal of getting as many correct answers. |
| 5.2 | Game must have 10 questions | The game must output 10 different questions for the user to answer |
| 5.3 | Computer must generate two random numbers between 1 and 10 | The computer must be able to generate a random number between 1 and 10, these are the numbers used to multiply with each other |
| 5.4 | Computer must multiply both numbers | The game must multiply both numbers, to give the user a final answer. |
| 5.5 | Allow the user to input what they think is the correct answer to the equation | The game must allow the user to input an answer on what they think the answer to the math’s equation is |
| 5.6 | Must tell the user how many correct answers they have | The game must display to the user the amount of correct answers they have gotten right out of the ten questions |
| 5.7 | Must ask the user if they want to play again | If the user wants to play again, the game will be restarted, however if they say No then they will redirected back to the main menu |

## Word Scramble Specification

|  |  |  |
| --- | --- | --- |
| **Requirement No.** | **Requirement** | **Description** |
| 6.1 | Create a world scramble game | The game must generate a word from the list of words and scramble that word for the user to guess |
| 6.2 | Game must get a random word from the list provided. | The game must get a random word from the array that will be provided and this will be the word that the user will guess. |
| 6.3 | Must scramble the word | The game must scramble the selected word so that it doesn’t look like the word that the user will guess. |
| 6.4 | Allow the user to input what they think the word is. | The game must allow the user to input an answer on what they think the correct word is. |
| 6.5 | Must tell the user how many guesses it took them | The game must display the amount of time the user has guessed to get the correct word |
| 6.6 | Must ask the user if they want to play again | If the user wants to play again, the game will be restarted, however if they say No then they will redirected back to the main menu |

|  |  |  |
| --- | --- | --- |
| **Requirement No.** | **Requirement** | **Description** |
| 7.1 | Create a rock paper and scissors game | The game should allow the user to play rock paper scissors just as they would in real life. |
| 7.2 | Allow the user to input their options | The user should be able to choose what option they want to use against the computer: rock, paper, scissors |
| 7.3 | Computer must output an option for the user to see | The game must randomly generate an option between the three to decide which wins in the game. |
| 7.4 | Winner of the match gets the point | When the user or computer chooses the winning option they must get a point for it. |
| 7.5 | First to 5 points wins the rock paper scissors game | In this game, the first one to hit 5 points will win the game |
| 7.6 | In this game, rock beats scissors | In this game, when someone outputs rock and someone outputs scissors, the one outputting rock would win the match up, that person would get a point. |
| 7.7 | In this game, paper beats rock | In this game, when someone outputs paper and someone outputs rock, the one outputting rock would win the match up, that person would get a point. |
| 7.8 | In this game, scissors beats paper | In this game, when someone outputs scissors and someone outputs paper, the one outputting rock would win the match up, that person would get a point. |
| 7.9 | In this game, if both output are the same it will be a draw | In this game, if both the output are the same, it will be a draw and neither would get a point. |
| 7.10 | Must ask the user if they want to play again | If the user wants to play again, the game will be restarted, however if they say No then they will redirected back to the main menu |

## Rock Paper Scissors Specification

## Pontoon Specification

|  |  |  |
| --- | --- | --- |
| **Requirement No.** | **Requirement** | **Description** |
| 8.1 | Create a pontoon game | The game should allow the user to play just as they would in real life. |
| 8.2 | The game must initially deal the computer and players two cards | The game should always deal two cards to the computer and user when the game starts |
| 8.3 | The game must have card values that are generated between 1 to 13 | The game must generate card values which are generated using numbers between 1 to 13 with 1 being Ace and 13 being King. |
| 8.4 | Ace must be given a value of 1 or 11 | In the game, the Ace can be given a value of 1 or 11 depending on the cards that the user have. |
| 8.5 | Picture cards must have an assigned value of 10 | In this game, pictures card such as Jack (11), Queen (12), and King (13) must have assigned value of 10, so if the user gets 2 Queens, their value would be 20. |
| 8.6 | Must keep choosing cards until the score is 15 | The game must ask the player and computer to keep choosing cards until they have scored at least 15 points |
| 8.7 | If the score is greater than 21, it is a lose | If the player or computer chooses a card and the total of those cards go over 21, then they will automatically lose. |
| 8.8 | Computer must win if both have the same score | If the player and computer both have the same score in the end e.g. they both get 18, the computer will always win |
| 8.9 | Closest to 21 or whoever gets 21 wins | The player who gets closest to 21 or gets to 21 will win the game. However, if it is over 21, e.g. 22 that player will lose as mentioned previously |
| 8.10 | Must ask the user if they want to play again | If the user wants to play again, the game will be restarted, however if they say No then they will redirected back to the main menu |

# Screen Designs

## Main Menu

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Press 1 for Higher or Lower \*

\*--------------------------------------------- \*

\* Press 2 to Maths Challenge \*

\*--------------------------------------------- \*

\* Press 3 for Word Scramble \*

\*--------------------------------------------- \*

\* Press 4 for Rock Paper Scissors \*

\*--------------------------------------------- \*

\* Press 5 for Pontoon \*

\*--------------------------------------------- \*

\* Press 6 for On Screen Help \*

\*--------------------------------------------- \*

\* Press 7 to Exit Program \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Input:

## Higher or Lower

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* HIGHER OR LOWER \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

---------------------------------------------------------------------------

I am thinking of a number between 1-100. Try and guess it.

--------------------------------

Guess:

--------------------------------

Higher.

Guess:

-------------------------------

Well done, you guessed it! You have guessed it x times!

-------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Play again? Y / N

User input:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## Maths Challenge

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Maths Challenge \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

---------------------------------------------------------------------------

First question: 8 x 6

Your answer:

32

Incorrect!

---------------------------------------------------------------------------

Second questions: 3 x 2

Your answer:

6

Correct!

---------------------------------------------------------------------------

Well done, you guessed it! You have guessed it x times!

-------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Play again? Y / N

User input:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## Word Scramble

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Word Scramble \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

---------------------------------------------------------------------------

The word is: rophanilthpist

------------------------------------------------------------------------------

Guess the word: elephant

Incorrect!

---------------------------------------------------------------------------

Guess the word: philantrophist

Correct!

-----------------------------------------------------------------------------

Well done! You guessed the word in x amount of times.

---------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Play again? Y / N

User input:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## Rock Paper Scissors

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Rock Paper Scissors \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

---------------------------------------------------------------------------

Let’s play rock paper scissors!

Rock

Paper

Scissors

User’s option: Scissors

Scissors

Computer chooses paper.

User beats Computer

-------------------------------------------------------------------------------------------

User [1] – Computer [0]

--------------------------------------------------------------------------------------------

Let’s play rock paper scissors!

Rock

Paper

Scissors

User’s option: Rock

Rock

Computer chooses paper.

Computer beats User

--------------------------------------------------------------------------------------------

User [4] – Computer [5]

-------------------------------------------------------------------------------------------

The COMPUTER has WON the match!

-------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Play again? Y / N

User input:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## Pontoon

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Pontoon \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

---------------------------------------------------------------------------

User’s hand: 7, 8

Computer’s hand: 8, X

-----------------------------------------------------------------------------

1. Hit
2. Stay

User input: 1

Hit.

-----------------------------------------------------------------------------

User’s hand: 7, 8, 6

Computer’s hand: 8, X

-----------------------------------------------------------------------------

1. Hit
2. Stay

User input: 2

Stay.

-----------------------------------------------------------------------------

User’s hand: 7, 8, 6

Computer’s hand: 8, 9

------------------------------------------------------------------------------

User’s total : 21

Computer’s total : 17

-------------------------------------------------------------------------------

User has won the match!

-------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Play again? Y / N

User input:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Data Dictionary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Identifier Name** | **Variable of Constant** | **Scope** | **Data Type** | **Value** | **Justification** |
| menuAnswer | Variable | Local | string | User input | The variable is where the user’s option of the main menu will be assigned. The variable will be used in an IF statement control structures to send the user to start the game, go to an onscreen help or exit the game. |
| menuLoop | Variable | Loop | Integer | 0 | This variable is used to loop the main menu, so that the user gets asked to input something if they input the incorrect option. |
| playerName | Variable | Global | String | User input | This variable will be used to refer to the user whenever an action is done such as in cases where they win. |
| gamesPlayed | Variable | Global | Integer | 0 | This variable is the amount of games the user has played as they are using the application. Every time the user launches a game, +1 will be added to this counter. |
| playerInput | Variable | Global | String | User input | This is the variable used to store any input that the user will be inputting when they are playing the games. This will be used throughout the program every time the user plays the game that they want. |
| playerAgainInput | Variable | Local | String | User input | This is where the user will input if they want to play again, this will be compared to an if statement where if they want to play again, the game restarts but if not they are sent to the main menu |
| wholeGameLoop | Variable | Local | integer | 0 or 1 | This is a variable to loop each game if the user wants to play again. It will be assigned with 0 to allow the player to play the game and 1 to end the loop |
| helpInput | Variable | Local | String | User input | This is the variable used for the user input, it will be used with an IF statement where if the answer is Y the user will be sent back to the main menu to be able to select the games. |
| guessesTaken | Variable | Local | Integer | 0 | This variable is used to store the amount of guesses the user will be guessing throughout the higher or lower game. |
| compNumber | Variable | Local | Integer | Random number | This variable will be used to store the random number generated by the random number generator between 1 and 100 |
| gameLoop | Variable | Local | Boolean | True | This is a variable to allow the user to keep guessing as many times as they want. It will have a value of TRUE until the user has guessed the correct number. |
| Score | Variable | Local | Integer | 0 | This is a variable used in the maths challenge, it will be used to show the user the amount of correct answers that they got out of the 10 random questions. For every question the user answers correctly, this variable will be added by 1 |
| mathsLoop | Variable | Local | Integer | 0 | This is a variable that will be used in a while loop in the Maths challenge. A for loop is used in this variable so that there are 10 questions generated to the user for them to answer. |
| compNumber1 | Variable | Local | Integer | Random number (1,10) | These variables are used to store that two random numbers that will be generated to be multiplied to give the user an equation to answer. |
| compNumber2 | Variable | Local | Integer | Random number (1,10) |
| Words | Array | Local | String | Python, programming, testing, games, design | This array will be used in the World Scramble game to store the words that will be scrambled for the user to guess. |
| correctWord | Variable | Local |  | Random.choice(Words) | This variable is used to store the randomly generated word that is taken from the array “Words”. The word in this variable will be used to scramble |
| Scramble | Variable | Local |  |  | Scramble is a variable used to store the scramble words that is the variable correctWord which will be displayed for the user to see. |
| Guesses | Variable | Local | Integer | 0 | The variable ‘guesses’ is a loop to allow the user to guess the scrambled word as many times as she wants until they guess it correctly |
| playerScore | Variable | Local | Integer | 0 | Both of these variables are used to store the score that a player will get if they win a game of rock paper and scissors. It will be used in a loop to keep the game going until the one of the score is 5. In control structures, one of these variables gets added by ‘1’, whoever wins the game. |
| compScore | Variable | Local | Integer | 0 |
| rpsLoop | Variable | Local | Integer | 0 | This is variable used in a loop to keep the rock paper and scissors game running |
| compChoiceInput | Variable | Local | string | Rock or Paper or Scissors | This variable is where the computer’s choice will be randomly generated using a random integer. In simple terms, it is the computer’s option between rock, paper and scissors. |
| compChoices | Array | Local | String | Rock, Paper, Scissors | This array is where the choices are stored and will be used as part to generate a random option for the computer. |
| deckOfCards | Array | Local | Integer | 1-10 with 16 tens in the array | This array is where all of the value of the cards is stored, it will be used to call a ‘random card’ from the deck |
| userHand | Array | Local | Integer |  | This array is where the user’s random generated cards will be stored both when the cards are initially being dealt, and when they hit. |
| dealerHand | Array | Local | Integer |  | This array is where the computer’s random generated cards will be stored both when the cards are initially being dealt, and when they hit. |
| Match | Variable | Local | Boolean | True | The variable is used to keep the pontoon game in a loop until the user decides to quit the game. |
| stick | Variable | Local | Boolean | False | The variable is used in a loop to stop the current match if the player decides to stick to their card |
| cardDeal | Variable | Local | Integer | 0 | The variable ‘cardDeal’ is used to loop the whole game so that the game keeps going on until the user wants to quit. |
| usersTotal | Variable | Local | Integer | Sum of User’s hand | This variable is used to find the sum of the user’s hand. It will be used in control structures to determine whether it is above, below or equal to 21 to check of the user has won or not |
| dealersTotal | Variable | Local | Integer | Sum of computer’s hand | This variable is used to find the sum of the computer’s hand. It will be used in control structures to determine whether it is above, below or equal to 21 to check of the user has won or not |

# Game Application Pseudocode

START

Import random, time, sys

Global playerName, playerInput, gamesPlayed

playerName 🡨 “ ”

playerInput 🡨 “ “

gamesPlayed 🡨 0

wholeGameLoop 🡨 0

Function MainMenu():

Execute Input player name function

Output (‘ Press 1 for Higher or Lower’

Output (‘Press 2 for Maths Challenge’)

Output (‘Press 3 for Word Scramble’)

Output (‘Press 4 for Rock, Paper, and Scissors’)

Output (‘Press 5 for Pontoon’)

Output (‘Press 6 for On Screen Help’)

Output (‘Press 7 to Exit’)

While menuLoop = 0

menuAnswer 🡨 user Input (‘Make your choice:’)

if menuAnswer == ‘1’ Then;

while wholegameLoop == 0

Execute function HigherorLower()

Execute function genNumber()

Execute function HoLNumber()

Execute function playAgain()

Else if menuAnswer == “2” Then;

Execute function MathsChallenge()

Execute Function MathsGenNumber()

Execute MathsUserGuess()

Execute function playAgain()

Else if menuAnswer == “3” Then;

Execute function WordScramble()

Execute function WordScrambleGame()

Execute function WordUserGuess()

Execute function playAgain()

Else if menuAnswer == “4” Then;

Execute function RPS()

Execute function RPSGame()

Execute function playAgain()

Else if menuAnswer == “5” Then;

Execute function Pontoon()

Execute function playAgain()

Else if menuAnswer == “6” Then;

Execute function OnScreenHelp()

Else if menuAnswer == “7” Then;

menuLoop 🡨 menuLoop + 1

Exit()

Else:

Output (‘Invalid option’)

Clear display.

return MainMenu()

Function playerName()

Output (‘Enter player name:’)

playerName 🡨 User input

if playerName == ‘ ‘ Then;

playerName 🡨 “Player 1”

end if

return playerName

function playAgain()

Output (‘Do you want to play again? [Y / N]’)

playAgainInput 🡨 user input

if playAgainInput == Y or y

wholeGameLoop 🡨 0

else if == N or n

wholeGameLoop 🡨 1

return MainMenu()

else

Output (“Invalid option”)

Function onScreenHelp()

Output (“ON-SCREEN HELP”)

Output (on screen help here)

Output (“Do you wish to go back?” Y/N)

helpInput🡨 user input

If helpInput 🡨 Y or y

Return MainMenu()

Else if helpInput 🡨 user input

Break

Else

Output (“Invalid option”)

## Higher or Lower Functions

Function HigherOrLower()

Output (‘HIGHER OR LOWER’)

gamesPlayed 🡨 gamesPlayed + 1

guessesTaken 🡨 0

function genNumber()

compNumber 🡨 random.randint(1,100)

Output (‘I’m thinking of a number between 1 and 100, try and guess what it is.)

Function HoLGuess(playerInput, guessesTaken)

While gameLoop = True

playerInput 🡨 User input (‘Take a guess’)

playerInput 🡨 int(playerInput)

guessesTaken 🡨 guessesTaken + 1

If playerInput == compNumber:

guessesTaken = str(guessesTaken)

Output(‘You guessed my number! You did it in’ + guessesTaken ‘amount of times’)

else if playerInput > compNumber:

Output(‘Number is too high’)

Else if playerInput < number :

Output (‘Number is too low’)

Else:

Output (‘That is not a number’)

## Maths Challenge Function

Function MathsChallenge()

Output (‘MATHS CHALLENGE’)

gamesPlayed 🡨 gamesPlayed + 1

Score 🡨 0

Function MathsGenNumber()

For mathsLoop in range( 10)

compNumber1 🡨 random.randint (1,10)

compNumber2 🡨 random.randint (1,10)

compAnswer 🡨 compNumber1 \* compNumber2

Output (compNumber1 + ‘X’ + compNumber2)

playerInput 🡨 Userinput (‘Answer: ‘)

playerInput 🡨 int(playerInput)

if playerInput == compAnswer:

Output(‘That is correct!’)

Score 🡨 score + 1

Else:

Output (‘that is incorrect’)

Output (‘Well done! You have completed the challenge.’)

Output (‘You scored’ , score , ‘out of 10’)

## Word Scramble Function

Function WordScramble()

Output (‘Word Scramble’)

gamesPlayed 🡨 gamesPlayed + 1

Words = [‘python’, ‘programming’, ‘testing’, ‘games’, ‘design’]

correctWord = random.choice(Words)

scramble = ‘ ‘

function WordScrambleGame()

scramble = list(theWord)

random.shuffle(scramble)

scramble = ''.join(random.sample(theWord, len(theWord)))

Output(‘The scrambled word is: ‘ , scramble)

function WordUserGuess(guesses)

guesses 🡨 0

playerInput 🡨 user input (‘What is the word: ‘)

while playerInput != correctWord:

Output(‘That word is incorrect!’)

Guesses = guesses + 1

playerInput 🡨 user input (‘Have another guess: ’ )

if playerInput == correctWord:

Output (‘That is the correct word!’)

Output (‘You have guessed it in’, guesses ‘amount of times’)

## Rock Paper Scissors Function

function RPS()

Output (“Rock Paper Scissors!”)

Output (“First to Five”)

gamesPlayed 🡨 gamesPlayed + 1

Function RPSGame()

playersScore 🡨 0

compScore 🡨 0

While rpsLoop = 0:

Output (“(“Rock, Paper, Scissors”)

While playerScore < 5 OR compScore <5

playerInput 🡨 user input

compChoices 🡨 [“Rock”, “Paper”, “Scissors”]

compChoiceInput 🡨 compChoices[randint(0,2)]

if userChoiceInput == compChoiceInput Then;

Output (‘It is a tie’)

Else If playerInput == “rock”:

If compChoiceInput == “paper”

Output ( playerName + ‘Lose! Paper beats Rock’)

CompScore + 1

Else:

Output ( playerName + ‘Wins! Rock beats Scissors”)

PlayerScore + 1

Else if playerInput == “paper”:

If compChoiceInput == “scissors”

CompScore + 1

Output (playerName + ‘Loses! Scissors beat Paper’)

Else:

PlayerScore + 1

Output (playerName + ‘Wins! Paper beats Rock’)

Else if playerInput == “scissors”:

If compChoiceInput == “rock”

CompScore + 1

Output ( playerName + ‘Lose! Rock beats Scissors’)

Else:

PlayerScore + 1

Output (playerName + ‘Wins!’ Scissors beats paper’)

Else:

Output (“It is an invalid input, please try again!”)

Return RPSPlayerChoice

rpsLoop 🡨 rpsLoop + 1;

## Pontoon Function

Function Pontoon()

Output (“Pontoon or 21”)

gamesPlayed 🡨 gamesPlayed + 1

deckOfCards 🡨 Store all cards in this array as integers between 1-10 with 16 tens in this array to represent the picture cards.

userHand = [ ]

dealerHand = [ ]

match 🡨 True

stick 🡨 false

cardDeal 🡨 0

Output (“PONTOON!”)

While match 🡨 true

While cardDeal =< 2

userHand 🡨 deckofCards[randint(0,52)]

dealerHand 🡨 deckofCards[randint(0,52)]

cardDeal 🡨 cardDeal + 1

while stick == false

Output (playerName + “this is your current hand:” + (userHand))

Output (Do you wish to [1] Hit and play another card or [2] stay with the current hand?

playerInput 🡨 userInput (“Your choice : “)

if playerInput == 1 then

userHand 🡨 deckofCards[randint(0,52)]

else if playerInput == 2 then

stick 🡨 true

else

Output (“Invalid option”)

Return playerInput

if dealersTotal < 15

dealerHand 🡨 deckofCards[randint(0,52)]

Output userHand

Output dealerHand

usersTotal 🡨 sum of userHand

dealersTotal 🡨 sum of dealerHand

if usersTotal > dealersTotal

if usersTotal <= 21

Output (“ You win! Well done”, playerName)

Else if usersTotal > 21

Output (“It’s a bust! Sorry”)

Else if usersTotal < dealersTotal

If dealersTotal > 21

Output (“The dealer went bust!” playerName “Wins!”)

Else

If usersTotal == dealersTotal

Output (“Although it’s a tie, the dealer wins!”)

match 🡨 false

#Main

Execute Main Menu function