GCSE Computer Science Non Exam Assessment 2019

Celebrity Dog Game

Veer Vohra - Dubai College

Contents

Introduction

AIMS AND OBJECTIVES

- The menu should be appealing and easy to use
- The entire program should be efficient
- The entire program should be user friendly and easy to use
- The game should work without any errors
- The game should be easy to understand and pick-up as a new player

GAME REQUIREMENTS

- User is prompted with menu in which they can play the game or quit
  - If “quit”, the program shuts down
- The user is asked how many cards they would like to play with
  - This is validated to be an even number within 4 and 30
- The deck of cards is split into 2 piles; one for the user and one for the computer
- The user’s first card is displayed on screen and they are asked for a category to play with
  - This is validated to ensure it is a valid option
- Their value is compared to the computer’s first card. The winner is decided.
- The cards are moved to the bottom of the decks

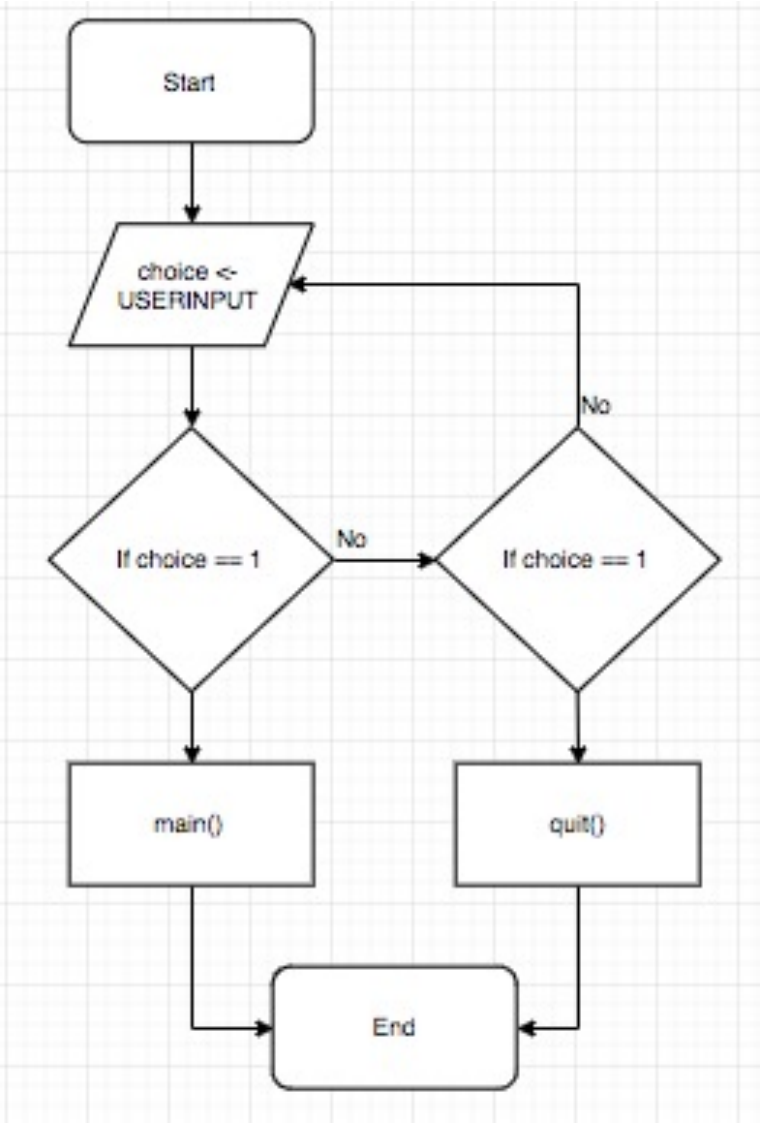
DESIGN

- The entire program will be split into subroutines to ensure that it is efficient and organized.
- The data types used are:
  - String
  - Integer
  - Array / List
- Data validation has been taken care of within each function

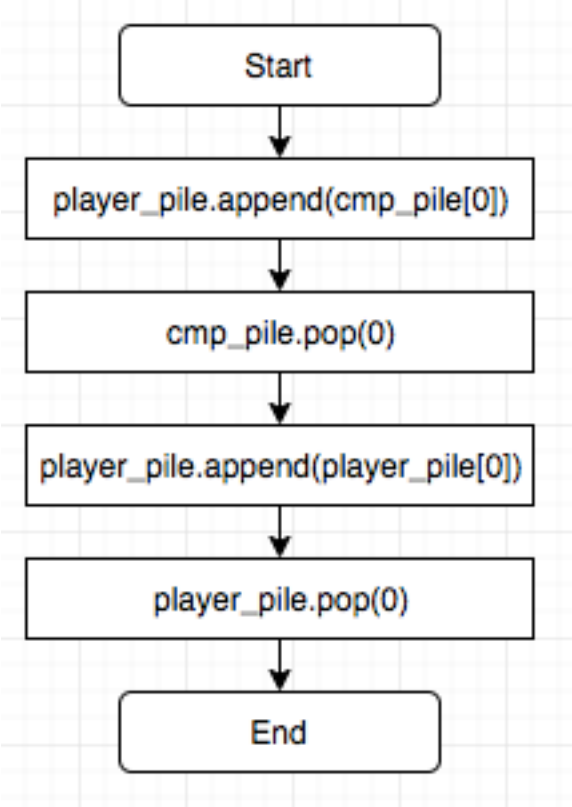
Design

——— MENU ———

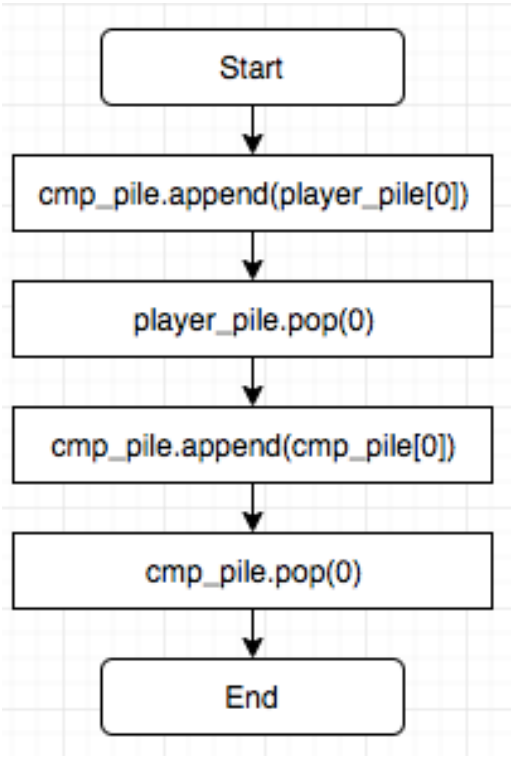
```
FUNCTION intro()
    choice <- USERINPUT
    WHILE choice != '1' AND choice != '2'
        choice <- USERINPUT
    ENDWHILE
    IF choice = ‘1' THEN
        main()
    ELSE
        quit()
    ENDIF
ENDFUNCTION
```



```
------ ROUND END (Player Win) ------
FUNCTION playerwin(player_pile ,
cmp_pile)
    player_pile.append(cmp_pile[0])
    cmp_pile.pop(0)
    player_pile.append(player_pile[0])
    player_pile.pop(0)
ENDFUNCTION
```

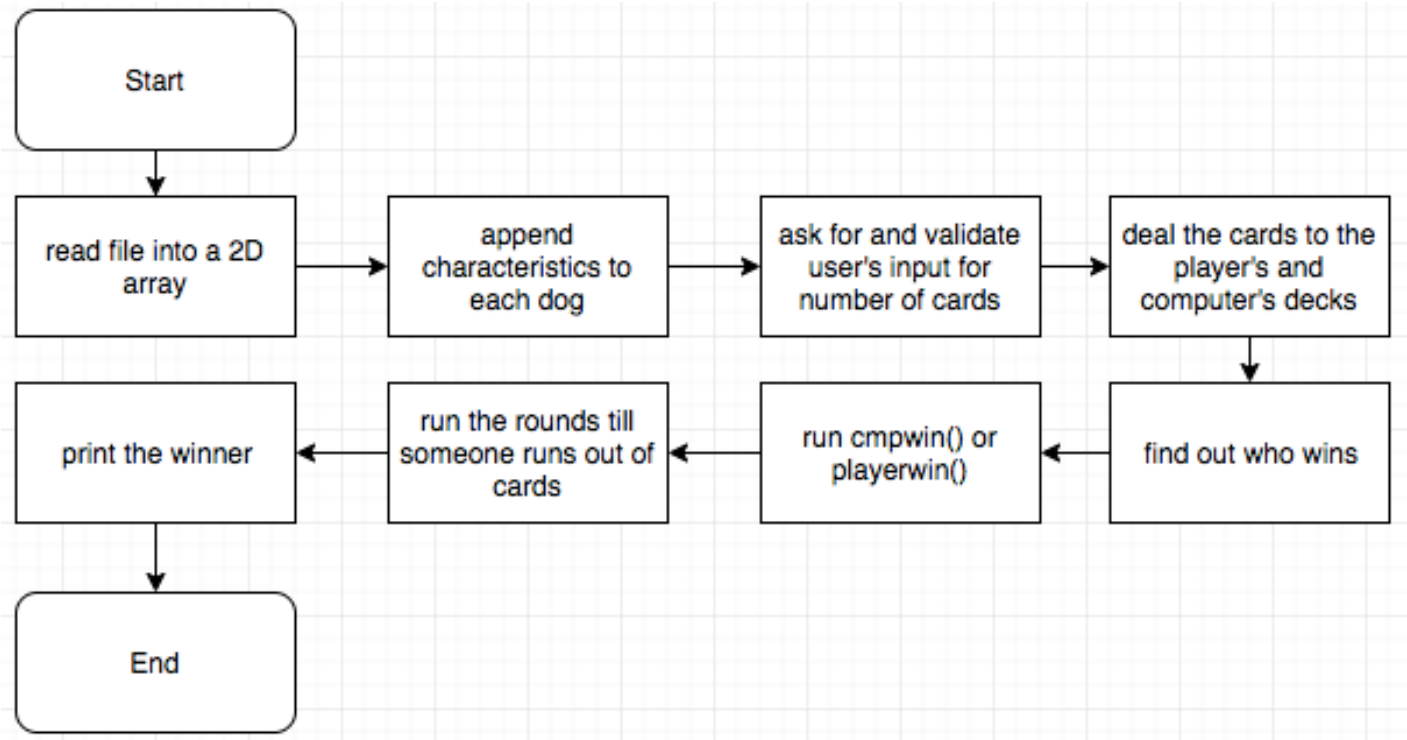



```
------ ROUND END (Computer Win) ------
FUNCTION cmpwin(player_pile , cmp_pile)
    cmp_pile.append(player_pile[0])
    player_pile.pop(0)
    cmp_pile.append(cmp_pile[0])
    cmp_pile.pop(0)
ENDFUNCTION
```



```
——— MAIN GAME ———
FUNCTION main()
    player_pile , cmp_pile , dog_cards , deck <- [] , [] , [] , []
    WITH open('dogs.txt') AS file
        file <- file.readlines()
    file <- [x.strip()  FOR x IN file]
    FOR dog IN file
        exercise , intelligence , friendliness , drool <-
random.randint(1,6) , random.randint(1,101) , random.randint(1,11) ,
random.randint(1,11)
        deck.append([dog, exercise , intelligence , friendliness ,
drool])
    END FOR
    cards <- USERINPUT
    WHILE cards.isalpha() = True
        cards <- USERINPUT
    ENDWHILE
    cards <- int(cards)
    WHILE cards < 4 OR cards > 30 OR ((cards % 2) != 0)
        cards <- USERINPUT
    ENDWHILE
    FOR i IN range(int(cards))
        dog_cards.append(random.choice(deck))
    END FOR
    half <- int((len(dog_cards) / 2))
    FOR i IN range(half)
        player_pile.append(dog_cards[i])
    END FOR
    FOR i IN range(half)
        cmp_pile.append(dog_cards[half])
        half += 1
    END FOR
    WHILE len(player_pile) != cards AND len(cmp_pile) != cards
        OUTPUT '_____________________\nYour Card  ',player_pile[0]
[0],'\n(1)Exercise   ',player_pile[0][1],' / 5\n(2)Intelligence
',player_pile[0][2],' / 100\n(3)Friendliness   ',player_pile[0][3],' /
10\n(4)Drool   ',player_pile[0][4],' / 10'
        category <- USERINPUT
        WHILE category != '1' AND category != '2' AND category != '3'
AND category != '4'
            category <- USERINPUT
```

```
        ENDWHILE
        OUTPUT '_____________________\nMy Card  ',cmp_pile[0]
[0],'\n(1)Exercise   ',cmp_pile[0][1],' / 5\n(2)Intelligence
',cmp_pile[0][2],' / 100\n(3)Friendliness   ',cmp_pile[0][3],' /
10\n(4)Drool   ',cmp_pile[0][4],' / 10'
        IF category = '1' THEN
            IF player_pile[0][1] >= cmp_pile[0][1] THEN
                playerwin(player_pile , cmp_pile)
            ELSEIF player_pile[0][1] < cmp_pile[0][1]
                cmpwin(player_pile , cmp_pile)
            ENDIF
        ELSEIF category = '2'
            IF player_pile[0][2] >= cmp_pile[0][2]
                playerwin(player_pile , cmp_pile)
            ELSEIF player_pile[0][2] < cmp_pile[0][2]
                cmpwin(player_pile , cmp_pile)
            ENDIF
        ELSEIF category = '3'
            IF player_pile[0][3] >= cmp_pile[0][3]
                playerwin(player_pile , cmp_pile)
            ELSEIF player_pile[0][3] < cmp_pile[0][3]
                cmpwin(player_pile , cmp_pile)
            ENDIF
        ELSEIF category = '4'
            IF player_pile[0][4] <= cmp_pile[0][4] THEN
                playerwin(player_pile , cmp_pile)
            ELSEIF player_pile[0][4] > cmp_pile[0][4]
                cmpwin(player_pile , cmp_pile)
            ENDIF
        ENDIF
        time.sleep(2)
    ENDWHILE
    IF len(player_pile) = cards THEN
        OUTPUT "YOU WIN\nWELL DONE"
    ELSE
        OUTPUT "I WIN\nUNLUCKY"
    ENDIF
    intro()
ENDFUNCTION
```


Start
read file into a 2D array
append characteristics to each dog
ask for and validate user's input for number of cards
deal the cards to the player's and computer's decks
find out who wins
run cmpwin() or playerwin()
run the rounds till someone runs out of cards
print the winner
End

Code

```
import random , time

def intro():
    ##### Main Menu #####
    choice = input("------------------------------\nWelcome to
Celebrity Dogs Game\n1 - Play Game\n2 - Quit\n>>> ")
    while choice != '1' and choice != '2': # validation
        choice = input("ERROR\nInvalid Option\nPlease Try Again\n>>>")
    if choice == '1': # if the user chooses to play
        main()
    else:
        print("------------------------------ Bye
------------------------------")
        quit() # exiting the program

def playerwin(player_pile , cmp_pile):
    # If the player wins, their card and the computer's card is moved
to the back of the player's deck
    player_pile.append(cmp_pile[0]) # append computer card
    cmp_pile.pop(0) # remove computer card
    player_pile.append(player_pile[0]) # append player card
    player_pile.pop(0) # remove player card

def cmpwin(player_pile , cmp_pile):
    # If the computer wins, their card and the player card is moved
      to the back of the computer's deck
    cmp_pile.append(player_pile[0]) # append player card
    player_pile.pop(0) # remove player card
    cmp_pile.append(cmp_pile[0]) # append computer card
    cmp_pile.pop(0) # remove computer card

def main():
    ##### Variables #####
    player_pile , cmp_pile , dog_cards , deck = [] , [] , [] , []

    ##### File Handling #####
    with open('dogs.txt') as file:
        file = file.readlines()
    file = [x.strip() for x in file] # removing "\n" from dog names
    # this has to be done because the names are on separate lines

    ##### Card Creation #####
    for dog in file: # iterating through all dog cards
        exercise , intelligence , friendliness , drool =
random.randint(1,5) , random.randint(1,100) , random.randint(1,10) ,
random.randint(1,10)
```

```
        deck.append([dog, exercise , intelligence , friendliness ,
drool]) # appending random values for the dog's characteristics

    ##### Number of Cards #####
    cards = input("Number of Cards : ")
    while True:
        try:
            cards = int(cards)
            if cards < 4 or cards > 30 or (cards % 2) != 0:
                raise Exception # raising an Exception
        except Exception:
            print("Card count must be a NUMBER inbetween 4 and 30
(inclusive)\n_____________________\n")
            intro()
        else:
            cards = int(cards)
            for i in range(int(cards)): # selecting random cards to be
played
                dog_cards.append(random.choice(deck))
            break

    ##### Dealing #####
    half = int((len(dog_cards) / 2))
    for i in range(half):#appending half the cards to the players deck
        player_pile.append(dog_cards[i])
    for i in range(half):#appending half the cards to the computer's
deck
        cmp_pile.append(dog_cards[half])
        half += 1

    ##### Game #####
    # running the game until someone has won (below)
    while len(player_pile) != cards and len(cmp_pile) != cards:
        print('_____________________\nYour Card: ',player_pile[0]
[0],'\n(1)Exercise : ',player_pile[0][1],' / 5\n(2)Intelligence :
',player_pile[0][2],' / 100\n(3)Friendliness : ',player_pile[0][3],' /
10\n(4)Drool : ',player_pile[0][4],' / 10')
        category = input("Please select a characteristic\n>>> ")
        while category != '1' and category != '2' and category != '3'
and category != '4': # validation
            category = input("ERROR\nPlease enter one of the
categories\n>>> ")
        print('_____________________\nMy Card: ',cmp_pile[0]
[0],'\n(1)Exercise : ',cmp_pile[0][1],' / 5\n(2)Intelligence :
',cmp_pile[0][2],' / 100\n(3)Friendliness : ',cmp_pile[0][3],' /
10\n(4)Drool : ',cmp_pile[0][4],' / 10')
        if category == '1': # if the user picks exercise
            if player_pile[0][1] >= cmp_pile[0][1]:
```

```
                print("_____________________\nCATEGORY : Exercise\nYOU
WIN\n_____________________")
                playerwin(player_pile , cmp_pile)
            elif player_pile[0][1] < cmp_pile[0][1]:
                print("_____________________\nCATEGORY : Exercise\nYOU
LOSE\n_____________________")
                cmpwin(player_pile , cmp_pile)

        elif category == '2': # if the user picks intelligence
            if player_pile[0][2] >= cmp_pile[0][2]:
                print("_____________________\nCATEGORY :
Intelligence\nYOU WIN\n_____________________")
                playerwin(player_pile , cmp_pile)
            elif player_pile[0][2] < cmp_pile[0][2]:
                print("_____________________\nCATEGORY :
Intelligence\nYOU LOSE\n_____________________")
                cmpwin(player_pile , cmp_pile)

        elif category == '3': # if the user picks friendliness
            if player_pile[0][3] >= cmp_pile[0][3]:
                print("_____________________\nCATEGORY :
Friendliness\nYOU WIN\n_____________________")
                playerwin(player_pile , cmp_pile)
            elif player_pile[0][3] < cmp_pile[0][3]:
                print("_____________________\nCATEGORY :
Friendliness\nYOU LOSE\n_____________________")
                cmpwin(player_pile , cmp_pile)

        elif category == '4': # if the user picks drool
            if player_pile[0][4] <= cmp_pile[0][4]:
                print("_____________________\nCATEGORY : Drool\nYOU
WIN\n_____________________")
                playerwin(player_pile , cmp_pile)
            elif player_pile[0][4] > cmp_pile[0][4]:
                print("_____________________\nCATEGORY : Drool\nYOU
LOSE\n_____________________")
                cmpwin(player_pile , cmp_pile)

        time.sleep(2) # rest for user to read output

    if len(player_pile) == cards: # checking if player won
        print("YOU WIN\nWELL DONE")
    else:
        print("I WIN\nUNLUCKY")
    intro()

intro()
```

Testing

——— VARIABLES ———

Variable Name	Data Type	Validation
choice	string	WHILE loop used to ensure it is 1 or 2
player_pile	2D array of strings and integers	N/A
cmp_pile	2D array of strings and integers	N/A
dog_cards	2D array of strings and integers	N/A
deck	2D array of strings and integers	N/A
file	string AND array	N/A
exercise	randomly generated integer	N/A
intelligence	randomly generated integer	N/A
friendliness	randomly generated integer	N/A
drool	randomly generated integer	N/A
cards	integer	WHILE loop + try/except is used to ensure it is a numeric value in-between 4 and 30.
category	integer	WHILE loop is used to ensure it is 1,2,3 or 4

——— TESTING TABLE ———

Test No	Test Description	Input Data	Test Output	How error was fixed
1	Testing the main menu	3 2	No ERROR Correctly exited	
2	Testing the main menu	1	No ERROR Correctly started the game	
3	Testing the main menu		ERROR (syntax) Missing “:”	Added “:”
4	Testing the exit function	2	No ERROR Correctly exited	
5	Testing the game	1(main menu) 4(number of cards) 1(exercise category)	No ERROR Game played as intended	
6	Testing the game	1(main menu) 400(number of cards) 41(number of cards) 31(number of cards) 26(number of cards)	No ERROR Data is validated correctly	
7	Testing post-game menu	(all inputs have not been included as they are simply in-game inputs) 4(drool category) 5(menu) 4(menu) 3(menu) 2(menu)	No ERROR Data is validated correctly until a valid input is entered	
8	Testing the game	1(main menu) 1(number of cards) h(number of cards)	ERROR “h” is not a numeric value and cannot be converted into an integer. This is due to the structure of the while loops	Re-writing and combining the 2 WHILE loops into a try/except function
9	Testing to ensure ERROR.8 was fixed	1(main menu) 1(number of cards) 1(main menu) h(number of cards) 1(main menu) 20(number of cards)	No ERROR Game runs as intended	

——— SCREENSHOTS ———

1.

```
------------------------------
Welcome to Celebrity Dogs Game
1 - Play Game
2 - Quit
>>> 3
ERROR
Invalid Option
Please Try Again
>>>2
------------------------------ Bye ------------------------------
```

2.

```
------------------------------
Welcome to Celebrity Dogs Game
1 - Play Game
2 - Quit
>>> 1
Number of Cards :
```

3.

```
------------------------------
Welcome to Celebrity Dogs Game
1 - Play Game
2 - Quit
>>> 1
Number of Cards : 4
--------------------
Your Card:  George the Great Dane
(1)Exercise :  3  / 5
(2)Intelligence :  50  / 100
(3)Friendliness :  2  / 10
(4)Drool :  8  / 10
Please select a characteristic
>>> 1
--------------------
My Card:  Frank the French Bulldog
(1)Exercise :  1  / 5
(2)Intelligence :  68  / 100
(3)Friendliness :  10  / 10
(4)Drool :  8  / 10
--------------------
CATEGORY : Exercise
YOU WIN
--------------------
```

4.

```
------------------------------
Welcome to Celebrity Dogs Game
1 - Play Game
2 - Quit
>>> 2
------------------------------ Bye ------------------------------
```

5.

```
  File "mainVEER.py", line 10
    while choice != '1' and choice != '2'
                                       ^
SyntaxError: invalid syntax
```

6.

```
----------------------------
Welcome to Celebrity Dogs Game
1 - Play Game
2 - Quit
>>> 1
Number of Cards : 400
ERROR
Card count must be an even number inbetween 4 and 30 (inclusive)
Number of Cards : 41
ERROR
Card count must be an even number inbetween 4 and 30 (inclusive)
Number of Cards : 31
ERROR
Card count must be an even number inbetween 4 and 30 (inclusive)
Number of Cards : 26
--------------------
Your Card:  William the Whippet
(1)Exercise :  3  / 5
(2)Intelligence :  70  / 100
(3)Friendliness :  7  / 10
(4)Drool :  7  / 10
Please select a characteristic
>>>
```

7.

```
--------------------
Your Card:  Annie the Afgan Hound
(1)Exercise :  1  / 5
(2)Intelligence :  37  / 100
(3)Friendliness :  1  / 10
(4)Drool :  1  / 10
Please select a characteristic
>>> 4
--------------------
My Card:  Gertie the Greyhound
(1)Exercise :  2  / 5
(2)Intelligence :  45  / 100
(3)Friendliness :  4  / 10
(4)Drool :  7  / 10
--------------------
CATEGORY : Drool
YOU WIN
--------------------
YOU WIN
WELL DONE
------------------------------
Welcome to Celebrity Dogs Game
1 - Play Game
2 - Quit
>>> 5
ERROR
Invalid Option
Please Try Again
>>>4
ERROR
Invalid Option
Please Try Again
>>>3
ERROR
Invalid Option
Please Try Again
>>>2
------------------------------ Bye ------------------------------
```

8.

```
----------------------------
Welcome to Celebrity Dogs Game
1 - Play Game
2 - Quit
>>> 1
Number of Cards : 1
ERROR
Card count must be an even number inbetween 4 and 30 (inclusive)
Number of Cards : h
Traceback (most recent call last):
  File "mainVEER.py", line 111, in <module>
    intro()
  File "mainVEER.py", line 13, in intro
    main()
  File "mainVEER.py", line 52, in main
    cards = int(input("ERROR\nCard count must be an even number inbetween 4 and 30 (inclusive)\nNumber of Cards : "))
ValueError: invalid literal for int() with base 10: 'h'
Veers-MacBook-Air:code veervohra$
```

9.

```
------------------------------
Welcome to Celebrity Dogs Game
1 - Play Game
2 - Quit
>>> 1
Number of Cards : 1
Card count must be a NUMBER inbetween 4 and 30 (inclusive)
--------------------

------------------------------
Welcome to Celebrity Dogs Game
1 - Play Game
2 - Quit
>>> 1
Number of Cards : h
Card count must be a NUMBER inbetween 4 and 30 (inclusive)
--------------------

------------------------------
Welcome to Celebrity Dogs Game
1 - Play Game
2 - Quit
>>> 1
Number of Cards : 20
--------------------
Your Card:  Harry the Harrier
(1)Exercise :  1  / 5
(2)Intelligence :  4  / 100
(3)Friendliness :  5  / 10
(4)Drool :  1  / 10
Please select a characteristic
>>>
```

Evaluation

- My code has met all my objectives because it is easy to use, user friendly and robust. In addition, my program has met all the requirements of the specification.

  - *"The menu should be appealing and easy to use"*
  - I met this objective by making the menu clean and understandable.

  - *"The entire program should be efficient"*
  - I met this objective by making my code as efficient as possible, using subroutines and reducing the number of lines.

  - *"The entire program should be user friendly and easy to use"*
  - I met this objective by keeping the outputs neat and clear.

  - *"The game should work without any errors"*
  - I met this objective by following the instructions, testing the code and ensuring that all the data was validated.

  - *"The game should be easy to understand and pick-up as a new player"*
  - I met this objective by explaining what the user has to do at each step.

- If I were to rewrite this game, I would incorporate a Graphical User Interface (GUI) using TKinter or Pygame. Furthermore, I would incorporate it into a website using HTML, CSS and Javascript. This would make it more enjoyable, user-friendly and appealing.