**Functions Spot Check Electronic Answer Document (EAD)**

Use the following document to record your answers to the assignment spot check. You should then submit the completed EAD to the link provided on Moodle by your teacher.

|  |
| --- |
| **Question 1 part (a)** |
| **#09-12-2014**  **#spot check q1**  **def set\_up():**  **password = ""**  **return password**  **def length(password):**  **pw\_length = len(password)**  **return pw\_length**  **def validation(pw\_length):**  **password = "a"**  **while pw\_length < 8 or pw\_length >16:**  **password = input("Please enter your password: ")**  **pw\_length = length(password)**  **if pw\_length<8:**  **print("Too short")**  **elif pw\_length >16:**  **print("Too long")**    **print("Password Accepted")**  **def main():**  **password = set\_up()**  **pw\_length = length(password)**  **validation(pw\_length)**  **main()** |
| **Question 1 part (b)** |
|  |

|  |
| --- |
| **Question 2 part (b)** |
| It displays nothing because a negative number is less than 20 so it passes the validation check however when the computer tries to display a negative number of grids it can’t and therefore displays nothing. |

|  |
| --- |
| **Question 2 part (c)** |
| def GetGridSize():  thisGridSize = 8  validGrid = False  while validGrid == False:  thisGridSize= int(input("Please enter the size of the grid(max 20): "))  if thisGridSize <= 20 and thisGridSize > 0:  validGrid = True  return thisGridSize  def GetGridRow(aRowLength):  # draws a single row using |\_ for each square  thisRow = '|\_' \* (aRowLength)  # add closing | to row  thisRow = thisRow + '|'  return thisRow  def DisplayGrid(aGridSize, aRow):  # display top of grid using \_ as top of each square  print(' \_' \* aGridSize)  # display rows of |\_| for each row  for rowCount in range(aGridSize):  print(aRow)  thisGridSize = GetGridSize()  rowToDraw = GetGridRow(thisGridSize)  DisplayGrid(thisGridSize, rowToDraw) |
| **Question 2 part (d)** |
|  |
| **Question 2 part (e)** |
|  |
| **Question 2 part (f)** |
|  |

|  |
| --- |
| **Question 3 part (a)** |
| def GetInput():  distance = float(input("Please input the distance: "))  mpg = float(input("Please input the Miles per gallon of the vehicle: "))  price = float(input("Please input the current price of fuel: "))  return distance,mpg,price  def CalculateCost(distance,mpg,price):  price\_per\_gallon = price\*4.55  gallons\_needed = distance/mpg  journey\_fuel\_cost = gallons\_needed\*price\_per\_gallon /100  return journey\_fuel\_cost  def DisplayCost(journey\_fuel\_cost):  print("The total fuel cost for the journey is £{0:.2f}".format(journey\_fuel\_cost))  def main():  distance,mpg,price = GetInput()  journey\_fuel\_cost = CalculateCost(distance,mpg,price)  DisplayCost(journey\_fuel\_cost)  main() |
| **Question 3 part (b)** |
|  |