**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)** |
| **def loop():**  **password\_check = "False"**  **while password\_check == "False":**  **user\_input = input("Please enter your password: ")**  **length = len(user\_input)**  **if length < 8:**  **print("Password is too Short")**  **elif length > 16:**  **print("Password is too long")**  **else:**  **password\_check = "True"**  **print("Password Accepted")**  **return user\_input**  **def main():**  **calling = loop()** |
| **Question 1 part (b)** |
|  |

|  |
| --- |
| **Question 2 part (b)** |
| Nothing happens  The algorithm has no stated what will happen when the user enters a negative value.  So the program continues and just prints out nothing as the rest of the calculations will all give out negative values. |

|  |
| --- |
| **Question 2 part (c)** |
| # define functions  def algorithm():  validGrid = False  while validGrid == False:  thisGridSize = int(input("Please enter the size of the grid(max 20): "))  #if thisGridSize < 0:  #print("You have entered an invalid number ({0})".format(thisGridSize))  if thisGridSize <= 20:  validGrid = True  #else:  #print("The value entered is too big({0})".format(thisGridSize))  return thisGridSize  def GetGridRow(rithm):  # draws a single row using |\_ for each square  thisRow = '|\_' \* (rithm)  # 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)  # main program  rithm = algorithm()  rowToDraw = GetGridRow(rithm)  DisplayGrid(rithm, rowToDraw) |
| **Question 2 part (d)** |
|  |
| **Question 2 part (e)** |
|  |
| **Question 2 part (f)** |
|  |

|  |
| --- |
| **Question 3 part (a)** |
| #Functions  def get\_input():  length\_of\_journey = float(input("Please enter your length of journey in miles: "))  miles\_per\_gallon = float(input("Please enter the miles per gallon of the vehicle: "))  price\_of\_fuel = float(input("Please enter the current price of fuel(in pence): "))  return length\_of\_journey, miles\_per\_gallon, price\_of\_fuel  def calculate\_cost(length\_of\_journey, miles\_per\_gallon, price\_of\_fuel):  total\_cost = (length\_of\_journey/miles\_per\_gallon)\*price\_of\_fuel  return total\_cost  def display\_cost(total\_cost):  print("The cost of fuel for the journey is £{0}".format(total\_cost))  #main  def main():  user\_input = get\_input()  calculate = calculate\_cost(length\_of\_journey, miles\_per\_gallon, price\_of\_fuel)  display = display\_cost(total\_cost) |
| **Question 3 part (b)** |
|  |