|  |  |
| --- | --- |
| Candidate name: | Justin Leow |
| Centre number: | Dunman High School |
| Index number: |  |
| Programming language used: | Python 3 |

|  |
| --- |
| **Question 1**  **Evidence 1** |
| with open("WORDS1.txt","r") as words:  high = 0  highword = ""  word = words.readline()  n = words.readline()  while word != "":  if int(n) > high:  highword = word  high = int(n)  word = words.readline()  n = words.readline()  print("term with highest amount of occurences is: " + highword + " with " + str(high) + " occurences.") |
| **Evidence 2** |
| *Paste screenshot here* |
| **Evidence 3** |
| *Paste program code here*  with open("WORDS2.txt","r") as words:  high = 0  highwords = []  highwords.append("")  word = words.readline()  n = words.readline()  while word != "":  if int(n) > high:  highwords = []  highwords.append(word)  high = int(n)  elif int(n) == high:  highwords.append(word)  word = words.readline()  n = words.readline()    highwordstr = ""    for word in highwords:  highwordstr += str(word[:-1]) + " and "  print("terms with highest amount of occurences are: " + highwordstr[:-4] + "with " + str(high) + " occurences.") |
| **Evidence 4** |
| *Paste screenshot here* |
| **Question 2**  **Evidence 5** |
| *Paste program code here*  import re  EmployeeID = []  Surname = []  with open("EMPLOYEEDATA.txt",'r') as infile:  line = infile.readline()  while line != "":  pcode = line.split(" : ")  for i in range(len(pcode)):  if pcode[i][0] == "E":  foo = pcode[i].split('"')  EmployeeID.append(foo[1])  elif pcode[i][0] == "S":  foo = pcode[i].split('"')  Surname.append(foo[1])    line = infile.readline()  #print(EmployeeID)  #print(Surname)  arr = [EmployeeID,Surname]  pattern = re.compile("[SLN][0-9]{3}")  print("Search by EmployeeID or Surname:")  query = input()  if re.match(pattern, query):  #employee search  search = 0  result = 1  else:  #surname search  search = 1  result = 0  found = False  for i in range(len(arr[search])):  if arr[search][i] == query:  print(arr[result][i])  found = True  if not found:  if search == 1: #surname search  print("Surname not found in database")  else: #employee search  print("Employee ID not found in database") |
| **Evidence 6** |
| *Paste annotated screenshots here*  Shows that that if EmployeeID is searched the proper surname is returned  Shows that that if surname is searched the proper EmployeeID is returned  If invalid search, then a proper error message is returned  Code allows for the returning of multiple results, for example multiple employees with the same surname. |
| **Question 3**  **Evidence 7** |
| *Paste program code here*  class Node():  LeftP, RightP, Data = None, None, 0    def \_\_init\_\_(self):  self.LeftP = None  self.RightP = None  self.Data = ""  ThisTree = [Node() for x in range(20)]  Root = 0  NextFreePosition = 0 |
| **Evidence 8** |
| *Paste program code here* |
| **Evidence 9** |
| *Paste program code here* |
| **Evidence 10** |
| *Paste screenshot here* |
| **Evidence 11** |
| *Paste screenshot here* |
| **Evidence 12** |
| *Paste program code here* |
| **Evidence 13** |
| *Paste screenshot here* |
| **Question 4**  **Evidence 14** |
| *Paste program code here* |
| **Evidence 15** |
| *Paste screenshot here* |
| **Evidence 16** |
| *Paste program code here* |
| **Evidence 17** |
| *Paste screenshot here* |
| **Evidence 18** |
| *Paste program code here* |
| **Evidence 19** |
| *Paste screenshots here* |