Candidate name: Justin Leow

Centre number:

Index number:

Programming language used: Python 3

|  |
| --- |
| **EVIDENCE 1**  def DecimalToBinary(DecimalNumber):  ans = ""  while DecimalNumber > 0: #repeated floor division  rem = DecimalNumber % 2 #modulus to get remainder  DecimalNumber = DecimalNumber // 2 #get quotient  ans += str(rem)    numzeros = 8 - len(ans)    realans = ""    for i in range(numzeros):  realans += '0' #add extra zeros to front of string    for i in range(len(ans)):  realans += ans[len(ans) - i - 1] #reverse string      print(realans)  DecimalToBinary(18)  DecimalToBinary(0)  DecimalToBinary(255)  DecimalToBinary(128)  DecimalToBinary(64) |
| **EVIDENCE 2** |
| **EVIDENCE 3**  def bitshift(string):  shiftedbit = string[0]  newstring = ""  for i in range(1,8): #shifts all bits forward by 1, except the eighth bit  newstring += string[i]  newstring += shiftedbit  return newstring  inputAccepted = False  while not inputAccepted:  string = input("Input bits to shift: ")  #validate input  if string == "":  print("Empty input") #presence check  elif len(string) != 8:  print("Input must be 8-bit") #length check  else:    for i in range(len(string)):  if string[i] != '0' and string[i] != '1': #value check  print("Input can only utilise the digits 0 and 1 for bits")  break  else:  inputAccepted = True  string = bitshift(string)  print(string) |
| **EVIDENCE 4**    Presence check    Length check    Boundary Data check (9 characters)      Input value check |
| **EVIDENCE 5**  def DecimalToBinary(DecimalNumber):  ans = ""  while DecimalNumber > 0:  rem = DecimalNumber % 2  DecimalNumber = DecimalNumber // 2  ans += str(rem)    numzeros = 8 - len(ans)    realans = ""    for i in range(numzeros):  realans += '0'    for i in range(len(ans)):  realans += ans[len(ans) - i - 1]      return realans  def bitshift(string):  shiftedbit = string[0]  newstring = ""  for i in range(1,8): #shifts all bits forward by 1, except the eighth bit  newstring += string[i]  newstring += shiftedbit  return newstring  ##main  word = input("Input word to be encrypted: ")  ans = ""  for i in range(len(word)):  char = word[i] #current character to encrypt  charAscii = ord(char) + 1  charBin = DecimalToBinary(charAscii)  charShifted = bitshift(charBin)  ans += charShifted + " "  print(ans) |
| **EVIDENCE 6** |
| **EVIDENCE 7**  def Factorial(n):  if n == 0 or n == 1:  return 1  else:  return n \* Factorial(n-1)  print("0!: " + str(Factorial(0)))  print("50!: " + str(Factorial(50)))  print("100!:" + str(Factorial(100))) |
| **EVIDENCE 8** |
| **EVIDENCE 9**  def Factorial(n):  ans = n  while n > 1:  ans \*= (n-1)  n = n-1  return ans |
| **EVIDENCE 10** |
| **EVIDENCE 11**  with open("MARKS.txt",'r') as infile:  data = []  line = infile.readline()  while line != "": #add info from read file into a 2D array  info = line.split(',')  info[2] = int(info[2]) #convert the last element, which includes \n, into purely an integer  #info[2] = info[2][0:2] will also work    data.append(info)    line = infile.readline()  ##main  highest = 0  highestStudent = []  total = 0  numStudents = 0  for info in data:  mark = info[2]  if mark > highest: #update highest mark and student  highest = mark  highestStudent = []  highestStudent.append(info[1])  elif mark == highest:  highestStudent.append(info[1]) #add student who got equally highest mark into an array  total += mark  numStudents += 1  highestStudentsStr = ""  for studentName in highestStudent: #print all students who got the same highest mark  highestStudentsStr += studentName + " and "  highestStudentsStr = highestStudentsStr[:-5]  #calculate average mark and format it to 2dp  avg = "{0:.2f}".format(total / numStudents)  print("The highest mark was " + str(highest) + " scored by " + highestStudentsStr)  print("The average mark was " + avg) |
| **EVIDENCE 12** |
| **EVIDENCE 13**  with open("MARKS.txt",'r') as infile:  data = []  line = infile.readline()  while line != "": #add info from read file into a 2D array  info = line.split(',')  info[2] = int(info[2]) #convert the last element, which includes \n, into purely an integer  #info[2] = info[2][0:2] will also work    data.append(info)    line = infile.readline()  ##3.1  highest = 0  total = 0  numStudents = 0  for info in data:  mark = info[2]  if mark > highest: #update highest mark and student  highest = mark  total += mark  numStudents += 1  #calculate average mark and format it to 2dp  avg = "{0:.2f}".format(total / numStudents)  ##main  with open("GRADES.txt",'w') as outfile:  for line in data: #loop for each line read    line.append(avg) #add average into next element to be output    #calculate grade  mark = line[2]  if mark == highest: #current student is highest scoring student  grade = 'M'  elif mark < float(avg) - 10: #score 10 marks below module average  grade = 'F'  else:  grade = 'P'    line.append(grade) #add grade as next element    line[2] = str(line[2]) #convert integer into string, so the array can work with .join  finalString = ",".join(line) #compiles all elements into one final string to be output  finalString += "\n"    outfile.write(finalString) #writes line into GRADES.txt |
| **EVIDENCE 14**  S0101,ALFRED ARUMUGAM,33,71.64,F  S0102,BILL BAXLEY,76,71.64,P  S0103,CATHERINE CHAN,69,71.64,P  S0104,DANIEL DAMIEN,39,71.64,F  S0106,FRED FLINT,89,71.64,P  S0108,HELEN HO,95,71.64,M  S0109,INDRA INUIT,94,71.64,P  S0110,JAMES JAY,95,71.64,M  S0111,JOSEPH TAN,65,71.64,P  S0112,Marion MATTHEW,72,71.64,P  S0115,Susan Costrua,61,71.64,F |
| **EVIDENCE 15**  def numSyllables(line):  syllableCount = 0  words = line.split(' ')  for word in words:  if word[-1] == ',' or word[-1] == '.':  word = word[:-1] #trim away the , and . in each word    vowelCount = 0  vowels = ['a','e','i','o','u','y','A','E','I','O','U','Y']  for i in range(len(word)): #count number of vowels in each word  if word[i] in vowels:  if word[i-1] in vowels:  #is part of a group vowel  pass  else:  #is an individual vowel  vowelCount += 1    if word[i] == 'e': #check if last character is e  vowelCount -= 1    if vowelCount == 0:  vowelCount = 1 #if no vowels, in word, it is treated as 1 syllable    syllableCount += vowelCount    return syllableCount  ##main  with open("PAS.txt",'r') as infile:  '''to read the middle ten lines, the total number of lines in the file will be checked  followed by getting the average - 5th line and reading ten from that number'''  for i, line in enumerate(infile):  pass  lastLine = i  midPos = lastLine // 2 - 5  lastPos = lastLine - 10    firstLines, midLines, lastLines = [],[],[]    infile.seek(0)    for i, line in enumerate(infile):  line = line[:-1] #trim away the \n from each line    if 0 <= i <= 9:  firstLines.append(line)  elif midPos <= i <= midPos + 9:  midLines.append(line)  elif i > lastPos:  lastLines.append(line)  testLines = firstLines + midLines + lastLines  numPSW = 0  for line in testLines:  if numSyllables(line) >= 3:  numPSW += 1  sqrtPSW = numPSW \*\* 0.5  SMOG = sqrtPSW + 3  print("Sample text name: PAS.txt")  print("No. of PSW: " + str(numPSW))  print("Square Root of PSW: {0:.2f}".format(sqrtPSW))  print("SMOG grade: {0:.2f}".format(SMOG)) #format SMOG to 2dp |
| **EVIDENCE 16** |