|  |  |  |
| --- | --- | --- |
| Input | Processing | Output |
| employeeNum (5 char)  employeeFirst (required, title case)  employeeLast (required, title case)  tripLocation  dateStart (required, format)  dateEnd (required, format, no more than 7 days after start)  carOwnership (required, O for Owned, R for Rented, uppercase)  carDistance (required, 2000km max, only if employee owns car)  claimType (S for Standard, E for Executive, upper) | #Define constants  DIEM\_RATE = 85.00  MILEAGE\_RATE = 0.17  CAR\_RENT\_RATE = 65.00  BONUS\_3\_DAY = 100.00  BONUS\_1000\_DISTANCE\_RATE = 0.04  BONUS\_EXECUTIVE\_RATE = 45.00  BONUS\_DEC\_RATE = 50.00  HST\_RATE = 0.15  #Gather inputs  while True:  while True:  employeeNum = input("Enter the employee's number (5 digits): ")  if employeeNum.isdigit() == False:  print("ERR: Employee Number must consist of numbers only.")  elif len(employeeNum) != 5:  print("ERR: Employee Number must be 5 digits long.")  else:  break    while True:  employeeFirst = input("Enter the employee's first name: ").title()  if employeeFirst == "":  print("ERR: Please enter the employee's first name.")  else:  break    while True:  employeeLast = input("Enter the employee's last name: ").title()  if employeeLast == "":  print("ERR: Please enter the employee's last name.")  else:  break    tripLocation = input("Enter the trip location: ")    while True:  try:  dateStart = input("Enter start date (MM-DD-YYYY): ")  dateStart = DateConvLib.strToDateConv(dateStart)  except:  print("ERR: Please enter the date in the format MM-DD-YYYY.")  else:  break    while True:  try:  dateEnd = input("Enter end date (MM-DD-YYYY): ")  dateEnd = DateConvLib.strToDateConv(dateEnd)  except:  print("ERR: Please enter the date in the format MM-DD-YYYY.")  else:  if dateEnd - dateStart < 1:  print("ERR: Please make sure the end date is later than the start date.")  elif dateEnd - dateStart > 7:  print("ERR: The end date must not be over 7 days later than the start date.")  else:  break    while True:  carOwnership = input("Enter whether the employee drove their own car or rented a car (O for Owned, R for Rented): ").upper()  if len(carOwnership) != 1:  print("ERR: Please enter either O for Owned or R for Rented.")  elif carOwnership != "O" and carOwnership != "R":  print("ERR: Please enter either O for Owned or R for Rented.")  else:  break    while True:  if carOwnership == "O":  try:  carDistance = int(input("Please enter employee mileage (km, must be under 2000km): "))  except:  print("ERR: Please input a number under 2000.")  else:  if carDistance > 2000 or carDistance < 0:  print("ERR: Please input a number between 0 and 2000.")  else:  break    while True:  claimType = input("Enter the employee's claim type (S for Standard, E for Executive): ").upper()  if len(claimType) != 1:  print("ERR: Please enter either S for Standard or E for Executive.")  elif carOwnership != "S" and carOwnership != "E":  print("ERR: Please enter either S for Standard or E for Executive.")  else:  break  #Perform calculations  dayDiff = int(dateEnd.days() – dateStart.days())  diemAmount = dayDiff \* DIEM\_RATE  if carOwnership == “O”:  mileageAmount = carDistance \* MILEAGE\_RATE  else:  mileageAmount = dayDiff \* CAR\_RENT\_RATE  bonusAmount = 0.00  if dayDiff > 3:  bonusAmount += BONUS\_3\_DAY  if carDistance > 1000 and carOwnership = “O”:  bonusAmount += (carDistance \* BONUS\_1000\_DISTANCE\_RATE)  if claimType = “E”:  bonusAmount += (dayDiff \* BONUS\_EXECUTIVE\_RATE)  if startDate.month() == 12 and (startDate.day() >= 15 or startDate.day() <= 22):  bonusAmount += (dayDiff \* BONUS\_DEC\_RATE)  claimAmount = diemAmount + mileageAmount + bonusAmount  taxes = claimAmount \* HST\_RATE  claimTotal = claimAmount + taxes  #Display outputs | Basic printout with headings  employeeNum  employeeFirst  employeeLast  tripLocation  dateStart  dateEnd  carOwnership  carDistance (if car is owned)  claimType  dayDiff  diemAmount  mileageAmount  bonusAmount  claimAmount  taxes  claimTotal |