Exercise 2:

(a) Fit the model, estimate coefficients β_l by MLE and their confidence intervals:
R-code:

```
#fitting the model:
model.InverseGaussianLog <- glm(Claim ~ CarType + DriverAge, data = ClaimData, family = inverse.gaussian("log"))
summary(model.InverseGaussianLog)

#coefficients estimated by MLE
model.InverseGaussianLog$coefficients

# 95% confidence interval for coefficients = solution to a)
confint(model.InverseGaussianLog)
```

R-output:

```
> #fitting the model:
> model.InverseGaussianLog <- glm(Claim ~ CarType + DriverAge, data = ClaimData, family = inverse.gaussian("log"))
> summary(model.InverseGaussianLog)

Call:
glm(formula = Claim ~ CarType + DriverAge, family = inverse.gaussian("log"),
    data = ClaimData)

Deviance Residuals:
       Min          1Q      Median          3Q         Max
-0.0020785  -0.0004329   0.0001098   0.0004375   0.0014463

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  7.69846    0.04348 177.060 2.19e-12 ***
CarType2    -0.06789    0.03904  -1.739 0.132677
CarType3     0.10321    0.04076   2.532 0.044535 *
DriverAge2  -0.21993    0.04966  -4.429 0.004427 **
DriverAge3  -0.37893    0.04800  -7.894 0.000219 ***
DriverAge4  -0.37477    0.04805  -7.800 0.000234 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for inverse.gaussian family taken to be 1.844004e-06)

    Null deviance: 2.0801e-04  on 11  degrees of freedom
Residual deviance: 1.1294e-05  on  6  degrees of freedom
AIC: 150.31

Number of Fisher Scoring iterations: 3

>
> #coefficients estimated by MLE
> model.InverseGaussianLog$coefficients
(Intercept)    CarType2    CarType3  DriverAge2  DriverAge3  DriverAge4
 7.69846480 -0.06788755  0.10321084 -0.21993411 -0.37892672 -0.37477120
>
> # 95% confidence interval for coefficients = solution to a)
> confint(model.InverseGaussianLog)
Waiting for profiling to be done...
                 2.5 %       97.5 %
(Intercept)  7.6132921  7.788693517
CarType2    -0.1447871  0.008787542
CarType3     0.0232250  0.183500495
DriverAge2  -0.3181684 -0.122798148
DriverAge3  -0.4741501 -0.285414616
DriverAge4  -0.4702144 -0.281032209
>
```

Comment: All parameters are significant, with one exception. Significance of CarType2 is questionable and we may discuss merging this class with class CarType1.

(b) Calculate the estimates and confidence intervals for multiplicative risk factors $\exp(\beta_l)$.
R-code:

```
#corresponding risk factors
exp(model.InverseGaussianLog$coefficients)

#95% confidence intervals for risk factors = solution to b)
exp(confint(model.InverseGaussianLog))
```

R-output:

```
> #corresponding risk factors
> exp(model.InverseGaussianLog$coefficients)
 (Intercept)     CarType2     CarType3   DriverAge2   DriverAge3   DriverAge4
2204.9603472    0.9343655    1.1087251    0.8025717    0.6845958    0.6874465
>
> #95% confidence intervals for risk factors = solution to b)
> exp(confint(model.InverseGaussianLog))
Waiting for profiling to be done...
                   2.5 %       97.5 %
(Intercept) 2024.9334472 2413.1627658
CarType2       0.8652064    1.0088263
CarType3       1.0234968    1.2014156
DriverAge2     0.7274803    0.8844422
DriverAge3     0.6224138    0.7517025
DriverAge4     0.6248683    0.7550040
```

(c) Calculate the predictions for expected claim amounts and their confidence intervals:

R-code:

```
#predictions on log link scale (for linear predictor) incl. standard error
logY <- predict(model.InverseGaussianLog, se.fit = TRUE)$fit
logY.stdErrors <- predict(model.InverseGaussianLog, se.fit = TRUE)$se.fit


#confidence intervals for claims in original scale = solution to c)
alpha <- 0.05
u <- qnorm(1-alpha/2, 0, 1)
lowerY <- exp(logY - u*logY.stdErrors)
upperY <- exp(logY + u*logY.stdErrors)

ClaimData["PredictClaim"] <-  exp(logY)
ClaimData["Claim_lowerBound"] <-  lowerY
ClaimData["Claim_upperBound"] <-  upperY
ClaimData
```

R-output:

```
> #predictions on log link scale (for linear predictor) incl. standard error
> logY <- predict(model.InverseGaussianLog, se.fit = TRUE)$fit
> logY.stdErrors <- predict(model.InverseGaussianLog, se.fit = TRUE)$se.fit
>
>
> #confidence intervals for claims in original scale = solution to c)
> alpha <- 0.05
> u <- qnorm(1-alpha/2, 0, 1)
> lowerY <- exp(logY - u*logY.stdErrors)
> upperY <- exp(logY + u*logY.stdErrors)
>
> ClaimData["PredictClaim"] <-  exp(logY)
> ClaimData["Claim_lowerBound"] <-  lowerY
> ClaimData["Claim_upperBound"] <-  upperY
> ClaimData
   CarType DriverAge Claim PredictClaim Claim_lowerBound Claim_upperBound
1        1         1  2000     2204.960         2024.842         2401.102
2        1         2  1800     1769.639         1635.380         1914.919
```

3	1	3	1500	1509.507	1400.621	1626.857
4	1	4	1600	1515.792	1406.309	1633.799
5	2	1	2200	2060.239	1894.153	2240.888
6	2	2	1600	1653.489	1529.974	1786.976
7	2	3	1400	1410.431	1310.436	1518.056
8	2	4	1400	1416.304	1315.756	1524.536
9	3	1	2500	2444.695	2240.705	2667.256
10	3	2	2000	1962.043	1809.454	2127.500
11	3	3	1700	1673.628	1549.538	1807.655
12	3	4	1600	1680.597	1555.836	1815.363

Comment: Confidence intervals for expected claim sizes (DO NOT confuse with the prediction interval for a single claim) are rather wide. This is due to small number of observation (we consider one claim in each risk class).