1)

```r
# yi: times to death
yi <- c(65, 156, 100, 134, 16, 108, 121, 4, 39, 143, 56, 26, 22, 1, 1, 5, 65)
yi
# xi: log10(initial white blood cell count)
xi <- c(3.36, 2.88, 3.63, 3.41, 3.78, 4.02, 4.4, 23, 3.73, 3.85, 3.97, 4.51, 4.54, 5.5, 4.72, 5)
xi
# Initial values b0
beta <- matrix(c(11, -2))
beta
#Design matrix X
X <- matrix(c(rep(1, 17), xi), nrow = 17, ncol = 2, byrow = FALSE)
X
#Transpose of design matrix of X
Xt <- t(X)
Xt
#The matrix of working weights W = Diagonal matrix
W <- diag(c(rep(1, 17)), nrow = 17, ncol = 17)
#The information matrix Tau = Xt * W * X
tau <- Xt %*% W %*% X
tau
#The score statistics U1 and U2
U1 <- sum(-1 + (yi / exp(beta[1] + beta[2] * xi)))
U2 <- sum(-xi + (yi * xi / exp(beta[1] + beta[2] * xi)))
#The vector of scores U
U <- matrix(c(U1, U2))
U
# Iterative equation
b <- beta + (solve(tau) %*% U)
b
```

Repeat the steps but change the initial value to the last value of beta you obtained. Stop the iteration process if you get the same value of beta in two successive steps.

2) OR By using Loop

```r
# yi: times to death
yi <- c(65, 156, 100, 134, 16, 108, 121, 4, 39, 143, 56, 26, 22, 1, 1, 5, 65)

# xi: log10(initial white blood cell count)
xi <- c(3.36, 2.88, 3.63, 3.41, 3.78, 4.02, 4.04, 4.23, 3.73, 3.85, 3.97, 4.51, 4.54, 5.00, 5.00, 4.72, 5.0
0)

# Initial values
beta <- matrix(c(11, -2))
epsilon <- 1e-6
max_iter <- 100

# Iterative process
for (iter in 1:max_iter) {
    # Design matrix X
    X <- matrix(c(rep(1, 17), xi), nrow = 17, ncol = 2, byrow = FALSE)
    # Transpose of design matrix of X
    Xt <- t(X)
    # Working weights matrix W = Diagonal matrix
    W <- diag(c(rep(1, 17)), nrow = 17, ncol = 17)
    # Information matrix Tau = Xt * W * X
    Tau <- Xt %*% W %*% X
    Tau_inver <- solve(Tau)
    # Score statistics
    U1 <- sum(-1 + (yi / exp(beta[1] + beta[2] * xi)))
    U2 <- sum(-xi + (yi * xi / exp(beta[1] + beta[2] * xi)))
    U <- matrix(c(U1, U2))
    # Iterative equation
    b <- beta + (Tau_inver %*% U)
    # Check for convergence
    if (max(abs(b - beta)) < epsilon) {
        break
    }
    # Update beta for the next iteration
    beta <- b
    cat("Iteration", iter, ": beta =", t(b), "\n")
}
```

3) using statistical software

```r
model <- glm(yi ~ xi, family = Gamma(link = "log"))
summary(model, dispersion = 1)

#Find the 95% confidence interval for BETA1 and BETA2
CI_of_beta <- confint.default(model, level = 0.95)
CI_of_beta

#Find approximately the variance-covariance matrix of the MLE b
 Tauinver <- vcov(model, dispersion = 1)
Tauinver

#Find approximate of the information matrix evaluated at b
Tau_at_b <- solve(vcov(model, dispersion = 1))
Tau_at_b

#The fitted values
yhat <- exp(8.4775 - 1.1093 * xi)   # or yhat <- fitted.values(model, dispersion = 1)
yhat
y <- yi
ri <- (-y - yhat) / yhat
ri
test_statistic <- sum(ri^2)
test_statistic
chi_table <- qchisq(1 - 0.05, 17 - 2)
chi_table
```