Problem 1

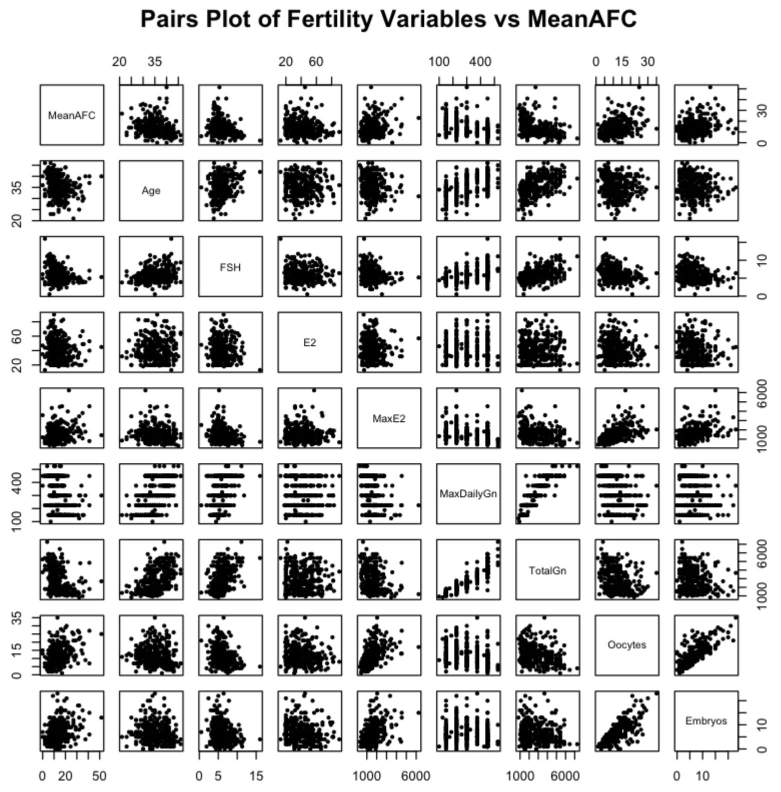
a. Create a correlation matrix using cor() to discover which variables - other than LowAFC - that appear to most strongly correlate to the variable MeanAFC. Create a pairs plot to acccompany the correlation matrix

In [21]:
```r
fertility <- read.csv("./data/Fertility.csv")

fertility_no_low <- fertility[, !names(fertility) %in% c("LowAFC")]
cor_matrix <- cor(fertility_no_low)
print(cor_matrix["MeanAFC", ])

pairs(fertility[, c("MeanAFC", "Age", "FSH", "E2", "MaxE2", "MaxDailyGn", "T
      main = "Pairs Plot of Fertility Variables vs MeanAFC",
      pch = 19,
      cex = 0.5)
```

```
       Age    MeanAFC        FSH         E2      MaxE2 MaxDailyGn    TotalGn
-0.2296947  1.0000000 -0.2963703 -0.1273285  0.2456819 -0.3966655 -0.3839206
  Oocytes    Embryos
0.4172390  0.3464034
```



Based on the data above, it looks like oocytes aand embryos have the strongest positive correlation, while MaxDailyGn and TotalGn show the strongest negative correlation.

b. Choose 4 to 6 variables (not LowAFC) and create a multi-variable linear model to predict the response variable MeanAFC. Use the backwards elimination technique to redice the model to just a set of significant variables (use a 0.05 level of significance) based on the p-values for the coefficients. Write the equatino for the model and beifly describe the meaning of the terms in the equation.

Using the results above, I'll use the mostly strongly correlated variables to create the initial model: Oocytes, MaxDailyGn, TotalGn, Embryos, and Age:

In [3]:
```
initial_model <- lm(MeanAFC ~ Oocytes + MaxDailyGn + TotalGn + Embryos + Age
summary(initial_model)
```

```
Call:
lm(formula = MeanAFC ~ Oocytes + MaxDailyGn + TotalGn + Embryos + 
    Age, data = fertility)

Residuals:
    Min      1Q  Median      3Q     Max 
-12.955  -3.787  -0.890   2.738  32.508 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) 15.3324597  2.9207096   5.250 2.75e-07 ***
Oocytes      0.3577948  0.0932478   3.837 0.000149 ***
MaxDailyGn  -0.0122019  0.0075659  -1.613 0.107763    
TotalGn     -0.0005977  0.0006121  -0.976 0.329571    
Embryos      0.1145409  0.1327383   0.863 0.388821    
Age         -0.0375001  0.0915929  -0.409 0.682498    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.421 on 327 degrees of freedom
Multiple R-squared:  0.2639,	Adjusted R-squared:  0.2527 
F-statistic: 23.45 on 5 and 327 DF,  p-value: < 2.2e-16
```

Using a 0.05 significance level, we can eliminate Age; while Embryos, TotalGn, and MaxDailyGn are also not significant, Age fails to meet the threshold more greatly than those:

In [4]:
```
model2 <- lm(MeanAFC ~ Oocytes + MaxDailyGn + TotalGn + Embryos, data = fert
summary(model2)
```

```
Call:
lm(formula = MeanAFC ~ Oocytes + MaxDailyGn + TotalGn + Embryos, 
    data = fertility)

Residuals:
    Min      1Q  Median      3Q     Max 
-13.171  -3.728  -1.030   2.683  32.345 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) 14.2958469  1.4541932   9.831  < 2e-16 ***
Oocytes      0.3543529  0.0927502   3.821 0.000159 ***
MaxDailyGn  -0.0130465  0.0072700  -1.795 0.073645 .  
TotalGn     -0.0006015  0.0006113  -0.984 0.325842    
Embryos      0.1183410  0.1322453   0.895 0.371518    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.413 on 328 degrees of freedom
Multiple R-squared:  0.2635,	Adjusted R-squared:  0.2545 
F-statistic: 29.34 on 4 and 328 DF,  p-value: < 2.2e-16
```

We'll reapply our process, eliminating Embryos:

In [5]:
```r
model3 <- lm(MeanAFC ~ Oocytes + MaxDailyGn + TotalGn, data = fertility)
summary(model3)
```

```
Call:
lm(formula = MeanAFC ~ Oocytes + MaxDailyGn + TotalGn, data = fertility)

Residuals:
    Min      1Q  Median      3Q     Max 
-13.452  -3.691  -0.984   2.864  32.274 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) 14.3720928  1.4512553   9.903  < 2e-16 ***
Oocytes      0.4160861  0.0619770   6.714 8.34e-11 ***
MaxDailyGn  -0.0130782  0.0072677  -1.799   0.0729 .  
TotalGn     -0.0006019  0.0006111  -0.985   0.3254    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.411 on 329 degrees of freedom
Multiple R-squared:  0.2617,	Adjusted R-squared:  0.255 
F-statistic: 38.88 on 3 and 329 DF,  p-value: < 2.2e-16
```

Finally, we'll remove TotalGn:

In [6]:
```r
model4 <- lm(MeanAFC ~ Oocytes + MaxDailyGn, data = fertility)
summary(model4)
```

```
Call:
lm(formula = MeanAFC ~ Oocytes + MaxDailyGn, data = fertility)

Residuals:
    Min      1Q  Median      3Q     Max 
-13.660  -3.789  -1.043   2.846  32.258 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 14.647697   1.423960  10.287  < 2e-16 ***
Oocytes      0.418061   0.061942   6.749 6.69e-11 ***
MaxDailyGn  -0.019523   0.003163  -6.172 1.98e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.411 on 330 degrees of freedom
Multiple R-squared:  0.2596,    Adjusted R-squared:  0.2551 
F-statistic: 57.84 on 2 and 330 DF,  p-value: < 2.2e-16
```

At this point, we have our final model, as both Oocytes and MaxDailyGn are statistically significant. Using the output, we can create our equation for our model using the estimates for our Intercept, Oocytes, and MaxDailyGn:

$$MeanAFC = 14.64 + 0.42(\text{Oocytes}) + -0.2(\text{MaxDailyGn})$$

c. Repeat part b, but use the `step()` command to automatically perform the backwards elimination. Discuss any differences that occur in this model from the one in part b.

In [7]:
```r
step_model <- step(initial_model, direction = "backward")
summary(step_model)
```

```
Start:  AIC=1244.43
MeanAFC ~ Oocytes + MaxDailyGn + TotalGn + Embryos + Age

             Df Sum of Sq   RSS    AIC
- Age         1      6.91 13489 1242.6
- Embryos     1     30.70 13513 1243.2
- TotalGn     1     39.31 13521 1243.4
<none>                    13482 1244.4
- MaxDailyGn  1    107.24 13589 1245.1
- Oocytes     1    607.02 14089 1257.1

Step:  AIC=1242.6
MeanAFC ~ Oocytes + MaxDailyGn + TotalGn + Embryos

             Df Sum of Sq   RSS    AIC
- Embryos     1     32.93 13522 1241.4
- TotalGn     1     39.82 13529 1241.6
<none>                    13489 1242.6
- MaxDailyGn  1    132.44 13622 1243.8
- Oocytes     1    600.27 14089 1255.1

Step:  AIC=1241.41
MeanAFC ~ Oocytes + MaxDailyGn + TotalGn

             Df Sum of Sq   RSS    AIC
- TotalGn     1     39.87 13562 1240.4
<none>                    13522 1241.4
- MaxDailyGn  1    133.09 13655 1242.7
- Oocytes     1   1852.46 15374 1282.2

Step:  AIC=1240.39
MeanAFC ~ Oocytes + MaxDailyGn

             Df Sum of Sq   RSS    AIC
<none>                    13562 1240.4
- MaxDailyGn  1    1565.3 15127 1274.8
- Oocytes     1    1872.0 15434 1281.5
Call:
lm(formula = MeanAFC ~ Oocytes + MaxDailyGn, data = fertility)

Residuals:
    Min      1Q  Median      3Q     Max 
-13.660  -3.789  -1.043   2.846  32.258 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 14.647697   1.423960  10.287  < 2e-16 ***
Oocytes      0.418061   0.061942   6.749 6.69e-11 ***
MaxDailyGn  -0.019523   0.003163  -6.172 1.98e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.411 on 330 degrees of freedom
Multiple R-squared:  0.2596,    Adjusted R-squared:  0.2551 
F-statistic: 57.84 on 2 and 330 DF,  p-value: < 2.2e-16
```

Using the `step()` function, we see that the results are nearly identical to our manual backwards elimination technique, with the model retaining Oocytes and MaxDailyGn as the final coefficients, with the same estimate and p-value.

d. Select 4 to 6 variables from your correlation matrix or pairs plot to create a multi-variable linear model to predict the response variable Embryos. Use either a by-hand backwards elimination or the `step()` command to reduce the model to a set of significant variables. Write the equation for the model and briefly describe the meaning of the terms in the equation.

In [8]:
```
print(cor_matrix["Embryos", ])
```

```
        Age     MeanAFC         FSH          E2       MaxE2  MaxDailyGn 
-0.12781624  0.34640339 -0.22317166 -0.08713926  0.43352785 -0.21837867 
    TotalGn     Oocytes     Embryos 
-0.20824976  0.75809878  1.00000000 
```

Looking at our output, we see that Oocytes, MaxE2, MeanAFC, MaxDailyGn, and TotalGn seem to have the highest correlation in either direction with Embryos, so let's use those for our backward elimination model:

In [9]:
```
initial_embryo_model <- lm(Embryos ~ Oocytes + MeanAFC + Age + MaxDailyGn +

stem_embryo_model <- step(initial_embryo_model, direction = "backward")
summary(stem_embryo_model)
```

```
Start:  AIC=660.52
Embryos ~ Oocytes + MeanAFC + Age + MaxDailyGn + TotalGn

             Df Sum of Sq    RSS    AIC
- TotalGn     1      0.02 2334.7 658.52
- MaxDailyGn  1      0.80 2335.5 658.63
- MeanAFC     1      5.32 2340.0 659.27
- Age         1     11.09 2345.8 660.09
<none>                    2334.7 660.52
- Oocytes     1   2488.55 4823.2 900.13

Step:  AIC=658.52
Embryos ~ Oocytes + MeanAFC + Age + MaxDailyGn

             Df Sum of Sq    RSS    AIC
- MaxDailyGn  1      3.78 2338.5 657.06
- MeanAFC     1      5.30 2340.0 657.27
- Age         1     11.08 2345.8 658.09
<none>                    2334.7 658.52
- Oocytes     1   2488.75 4823.5 898.14

Step:  AIC=657.06
Embryos ~ Oocytes + MeanAFC + Age

          Df Sum of Sq    RSS    AIC
- MeanAFC  1      3.48 2342.0 655.55
- Age      1      7.33 2345.8 656.10
<none>                 2338.5 657.06
- Oocytes  1   2514.46 4853.0 898.17

Step:  AIC=655.55
Embryos ~ Oocytes + Age

          Df Sum of Sq    RSS    AIC
- Age      1      9.92 2351.9 654.96
<none>                 2342.0 655.55
- Oocytes  1   3097.81 5439.8 934.19

Step:  AIC=654.96
Embryos ~ Oocytes

          Df Sum of Sq    RSS    AIC
<none>                 2351.9 654.96
- Oocytes  1    3178.2 5530.1 937.67
```

```
Call:
lm(formula = Embryos ~ Oocytes, data = fertility)

Residuals:
     Min       1Q   Median       3Q      Max 
-10.0914  -1.2416   0.1418   1.3283  10.4785 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.53176    0.32732   1.625    0.105    
Oocytes      0.52332    0.02474  21.149   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.666 on 331 degrees of freedom
Multiple R-squared:  0.5747,	Adjusted R-squared:  0.5734 
F-statistic: 447.3 on 1 and 331 DF,  p-value: < 2.2e-16
```

Our final model shows that only Oocytes seems to be statistically significant, and we can use the output above to create our equation:

$$Embryos = .53 + .52(\text{Oocytes})$$

Problem 2

Split the original datafile to create a training sample (using 75% and 25% for the training set and testing set, respectively - I'll be using Professor Ahrens's suggestion of randomizing which cases go into which)

In [10]:
```
gpa_data <- read.csv("./data/FirstYearGPA.csv")

set.seed(42)

train_index <- sample(1:nrow(gpa_data), 0.75 * nrow(gpa_data))

train_data <- gpa_data[train_index, ]
test_data <- gpa_data[-train_index, ]
```

a. Use the training sample to fit a multiple regression to predict GPA using HSGPA, HU, and White. Give the prediction equation along with output to anaylze the effectiveness of each predictor, estimated standard deviation of the error term, and R^2 to assess the overall contribution to the model.

In [11]:
```
gpa_model <- lm(GPA ~ HSGPA + HU + White, data = train_data)
summary(gpa_model)
```

```
Call:
lm(formula = GPA ~ HSGPA + HU + White, data = train_data)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.78236 -0.28340  0.01448  0.21625  0.79553 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 1.047483   0.268090   3.907 0.000138 ***
HSGPA       0.450775   0.077632   5.807 3.34e-08 ***
HU          0.017403   0.003924   4.435 1.70e-05 ***
White       0.347254   0.072713   4.776 4.02e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3703 on 160 degrees of freedom
Multiple R-squared:  0.3759,	Adjusted R-squared:  0.3642 
F-statistic: 32.13 on 3 and 160 DF,  p-value: 2.614e-16
```

Using the output, we can create our prediction equation as

$$GPA = 1.05 + 0.45(\text{HSGPA}) + 0.02(\text{HU}) + 0.35(\text{White})$$

Looking at R^2, our model explains roughly 37.6% of the variance in college GPA, the F-statistic suggests the model is significantly better than no model. All predictors are highly significant, though it's notable that about 63% of variance remains unexplained.

b. Use the prediction equation in the previous part as a formula to generate predictinos of the GPA for each of the cases in the holdout sample. Also, compute the prediction errors by subtracting the prediction from the actual GPA for each case.

In [18]:
```r
test_predictions <- predict(gpa_model, newdata = test_data)

prediction_errors <- test_data$GPA - test_predictions

results <- data.frame(
    Actual = test_data$GPA,
    Predicted = test_predictions,
    Error = prediction_errors
)
print(prediction_errors)
```

```
          7           8          11          17          18          19 
 0.40981021  0.02569476  0.29802305 -0.37880706  0.26073670  0.11440899 
         21          22          23          30          39          44 
 0.46935651 -0.13919748 -0.48147050 -0.07811855 -1.12448202 -0.74918036 
         45          46          48          50          51          52 
-0.11438113  0.51701080 -0.76086182 -0.63332095 -0.56004602 -0.06805461 
         53          59          64          75          77          79 
-0.46349752 -0.17371917 -0.12414434  0.23508948 -0.67876812  0.81737963 
         87          88          93          94          98         105 
-0.24676884 -0.03225575 -0.36669428  0.30657697  0.02581924  0.72945873 
        107         117         123         124         135         142 
 0.16807326  0.12446679 -0.01981678  0.35387205  0.27037128 -0.24495993 
        145         147         151         155         159         167 
-0.88100025  0.31362606  0.13958322 -0.44330361  0.36592154 -0.01101960 
        172         173         175         178         179         180 
-0.82732818  0.21001453  0.04526092  0.19599076 -0.10940454  0.34028827 
        186         196         198         201         204         206 
-0.42905898 -0.59706758  0.21888625  0.31057834  0.69866540  0.08621241 
        209 
 0.45894174 
```

c. Compute the mean and standard deviation for the prediction errors. Is the mean reasonably close to zzero? Is the standard deviation reasonably close to the standard deviation of the error term from the fit to the training sample?

In [19]:
```r
summary(prediction_errors)
rmse <- sqrt(mean(prediction_errors^2))
print(paste("Root Mean Square Error:", round(rmse, 4)))
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
-1.12448 -0.37275  0.02569 -0.04048  0.28420  0.81738 
[1] "Root Mean Square Error: 0.4358"
```

Our mean is -0.04, which is reasonably close to zero, indicating a slight tendency to overpredict GPA. Our RMSE is .4358, indicating a typical prediction error of about ± 0.44 GPA points. On a 4.0 scale, this is about an 11% error rate. I would say that this aligns with the R^2 value earlier of .376, indicating moderate predictive power.

d. Compute the cross-validation correlation between the actual and predicted GPA values for the cases in the holdout sample.

In [20]:
```r
cv_correlation <- cor(test_data$GPA, test_predictions)
print(paste("Cross-validation correlation:", round(cv_correlation, 4)))
```

```
[1] "Cross-validation correlation: 0.3974"
```

The correlation of .3974 indicates a moderate positive relationship between predicted and actual GPA - the model's predictions tend to move in the same directino as actual GPAs; again this roughly aligns with our assessment of the R^2 value of .376.

e. Square the cross-validation correlation and subtract from R^2 for the training sample to compute the shrinkage. Does it look like the training model works reasonably well for the holdout sample or has there been a considerable drop in the amount of variability explained?

Squaring the cross-validation correlation gives us a value of .1579, which results in a shrinkage of .3759 - .1579 = .218, which represents a drop from explaining 37.6% of variance in the training data to around 15.8% in the test data. Our model's predictive power has notably weakened. We might want to use a simpler model to reduce overfitting, or collect more training data to create more accurate coefficients.

Problem 3

a. What model would you use to predict the typical Time of a hike using any combination of the other variables as predictors? Justify your choice.

All columns except for Peak (Length, Ascent, Difficulty, and Elevation) would seem to be key predictors. Length is obviously a strong predictor - any activity that involves walking, hiking or running is going take more time for a person the longer the distance. Ascent as an indicator of altitude change would also correlate well with time, since the greater the elevation change the steeper the hike becomes. Difficulty is another strong indicator; this seems self-evident. Lastly, elevation could be a moderate predictor - higher elevation means thinner air and a slower pace, although I suspect that a large percentage of people tackling one of the peaks listed are relatively fit and might also be altitude adjusted, which would lessen the impact of elevation on performance.

Given all of that, I'm going to use a simple model that uses only Length and Difficulty to predict Time. I suspect that Difficulty is already factoring both Elevation and Ascent, and thus including the latter two columns would only add marginal predictive power to our model.

In [21]:
```
peaks_data <- read.csv("./data/HighPeaks.csv")
time_model <- lm(Time ~ Length + Difficulty, data = peaks_data)
summary(time_model)

# I want to check the correlation between the selected predictors and the ot
cor_matrix <- cor(peaks_data[, c("Time", "Length", "Difficulty", "Ascent", "
print("Correlation matrix:")
print(cor_matrix)
```

```
Call:
lm(formula = Time ~ Length + Difficulty, data = peaks_data)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.5341 -0.7174 -0.1508  0.5402  3.3743 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.15677    0.89369   0.175 0.861577    
Length       0.45784    0.08436   5.427 2.47e-06 ***
Difficulty   0.88969    0.25176   3.534 0.000994 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.291 on 43 degrees of freedom
Multiple R-squared:  0.7962,	Adjusted R-squared:  0.7867 
F-statistic: 84.01 on 2 and 43 DF,  p-value: 1.403e-15
[1] "Correlation matrix:"
                 Time    Length Difficulty    Ascent  Elevation
Time        1.0000000 0.8585079  0.8103239 0.4688753 -0.0162768
Length      0.8585079 1.0000000  0.7595616 0.4849692  0.1895775
Difficulty  0.8103239 0.7595616  1.0000000 0.3737203  0.1272571
Ascent      0.4688753 0.4849692  0.3737203 1.0000000  0.3395255
Elevation  -0.0162768 0.1895775  0.1272571 0.3395255  1.0000000
```

Interestingly, difficulty and ascent are actually not as strongly correlated with Difficulty as I expected; the moderate correlation is there, but lower than I would've thought. That said, when it comes to correlation with Time, both Length and Difficulty are much more strongly correlated than either Ascent or Elevation. In order to be fully sure, I'll use the backwards elimination technique to produce my final model.

In [22]:
```r
peaks_data <- read.csv("./data/HighPeaks.csv")
full_model <- lm(Time ~ Length + Difficulty + Ascent + Elevation, data = pea
final_model <- step(full_model, direction = "backward")
summary(final_model)
```

```
Start:  AIC=19.22
Time ~ Length + Difficulty + Ascent + Elevation

             Df Sum of Sq    RSS    AIC
<none>                    56.207 19.218
- Ascent      1     4.521 60.727 20.777
- Elevation   1    14.236 70.443 27.603
- Difficulty  1    19.661 75.868 31.016
- Length      1    40.937 97.143 42.387
```

```
Call:
lm(formula = Time ~ Length + Difficulty + Ascent + Elevation,
    data = peaks_data)

Residuals:
     Min       1Q   Median       3Q      Max
-1.77942 -0.81216 -0.08647  0.68962  3.06736

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.9567864  2.2307630   2.670  0.01082 *
Length       0.4440084  0.0812523   5.465 2.49e-06 ***
Difficulty   0.8654527  0.2285275   3.787  0.00049 ***
Ascent       0.0006011  0.0003310   1.816  0.07669 .
Elevation   -0.0016703  0.0005183  -3.223  0.00249 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.171 on 41 degrees of freedom
Multiple R-squared:  0.8401,    Adjusted R-squared:  0.8245
F-statistic: 53.84 on 4 and 41 DF,  p-value: 8.738e-16
```

b. Examine plots using the residuals from your fitted model in (a) to assess the regression conditions of linearity, homoscedasticity, and normality in this situation. Comment on whether each of the conditions is reasonable for this model.

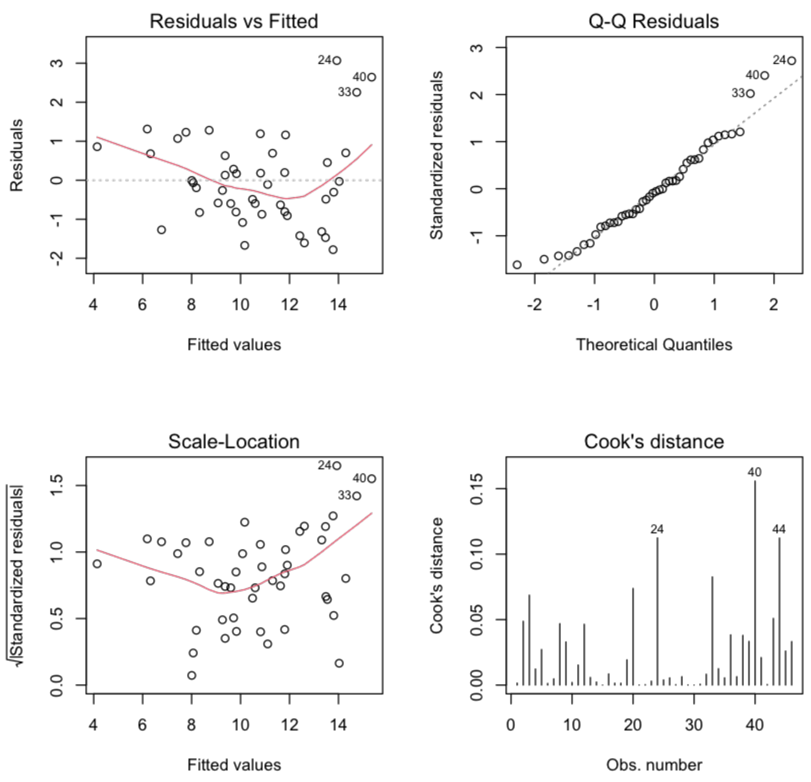
In [23]:
```r
# diagnostic plots
par(mfrow=c(2,2))

# Residuals vs Fitted for linearity and homoscedasticity
plot(final_model, which=1)

# Normal Q-Q plot for normality
plot(final_model, which=2)

# Scale-Location plot for homoscedasticity
plot(final_model, which=3)

# Residuals vs Leverage for influential points
plot(final_model, which=4)
```



Based on the data, I'd say the model is reasonably normal, though not perfect, as we can see some outliers, especially towards the right. For linearity, we see a moderate curve around or x-axis, indicating that we might want to do more to our model data to normalize. Regarding, homoscedasticity, we again see a decent spread around our line, although the pattern seems to follow the same curve.

c. Find the studentized residuals for the model in (a), and comment on which mountains (if any) might stand out as being unusual according to this measure.

In [24]:
```r
studentized_residuals <- rstudent(final_model)

residual_data <- data.frame(
    Peak = peaks_data$Peak,
    Studentized_Residual = studentized_residuals
)

residual_data <- residual_data[order(abs(residual_data$Studentized_Residual)

print("Peaks with largest studentized residuals:")
print(residual_data[abs(residual_data$Studentized_Residual) > 2, ])
```

```
[1] "Peaks with largest studentized residuals:"
             Peak Studentized_Residual
24   Seward Mtn.             2.964556
40    Mt. Emmons             2.562853
33 Mt. Donaldson             2.103006
```

d. Are there any mountains that have high leverage or may be influential on the fit? If so, identify the mountain(s) and give values for the leverage or Cook's D, as appropriate

In [27]:
```
leverage <- hatvalues(final_model)

cooks_d <- cooks.distance(final_model)

influence_data <- data.frame(
    Peak = peaks_data$Peak,
    Leverage = leverage,
    Cooks_D = cooks_d
)

high_leverage <- influence_data[leverage > 2 * mean(leverage), ]
print("Peaks with high leverage (> 2 * mean):")
print(high_leverage)

influential <- influence_data[cooks_d > 4/nrow(peaks_data), ]
print("Peaks with high Cook's D (> 4/n):")
print(influential)
```

```
[1] "Peaks with high leverage (> 2 * mean):"
            Peak  Leverage     Cooks_D
1     Mt. Marcy  0.2231269 0.001517841
36 Cascade Mtn.  0.2177385 0.038385934
44   Cliff Mtn.  0.2178475 0.112313909
45     Nye Mtn.  0.2759267 0.026015365
[1] "Peaks with high Cook's D (> 4/n):"
           Peak   Leverage   Cooks_D
24 Seward Mtn.  0.07072971 0.1124280
40  Mt. Emmons  0.11873558 0.1558283
44  Cliff Mtn.  0.21784754 0.1123139
```

The output above suggests that Cliff Mtn is particularly noteworth, having both high Cook's D and high leverage. Additionaly, Mt. Emmons has the highest Cook's D and Nye Mtn. has the highest leverage; all 3 are worthy of closer examination to understand why they differ from the typical hiking patterns we see with other peaks.

Problem 4

Create a model to predict the resting systolic blood pressure, SystolicBP, but with the added inclusion of the indicator variable for smoking, ISmoke, as one of the predictors, along with the indicator variables for weight, IOverweight and IObese. You do not need and can ignore the variable Overwt.

In [28]:
```
blood_data <- read.csv("./data/Blood1.csv")

bp_model <- lm(SystolicBP ~ ISmoke + IOverweight + IObese, data = blood_data

summary(bp_model)
```

```
Call:
lm(formula = SystolicBP ~ ISmoke + IOverweight + IObese, data = blood_data)

Residuals:
    Min      1Q  Median      3Q     Max 
-64.897 -18.425  -4.337  17.308  71.845 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  131.897      2.271  58.076  < 2e-16 ***
ISmoke         9.181      2.416   3.800 0.000163 ***
IOverweight    8.259      3.215   2.569 0.010503 *  
IObese        15.614      2.721   5.739 1.66e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 26.68 on 496 degrees of freedom
Multiple R-squared:  0.09737,	Adjusted R-squared:  0.09191 
F-statistic: 17.83 on 3 and 496 DF,  p-value: 5.237e-11
```

Our 3 coefficients are all statistically significant, though *IOverweight* seems to carry the least weight (no pun intended). Within our model, *Ismoke* adds 9 points to SystolicBP, with *Ioverweight* and *IObese* contributing another 8 and 15, respectively. All standard errors hover between 2.2 and 3.2, R^2 indicates that our model explains roughly 10% of the variance in our data

b. Look at the residuals using a histogram, normal quantile plot, and a summary of their values. By also considering the value of R^2, discuss how appropriate your final model seems to be. Using your knowledge of human health, what other variables are likely to be important in predicting a person's resting systolic blood pressure?

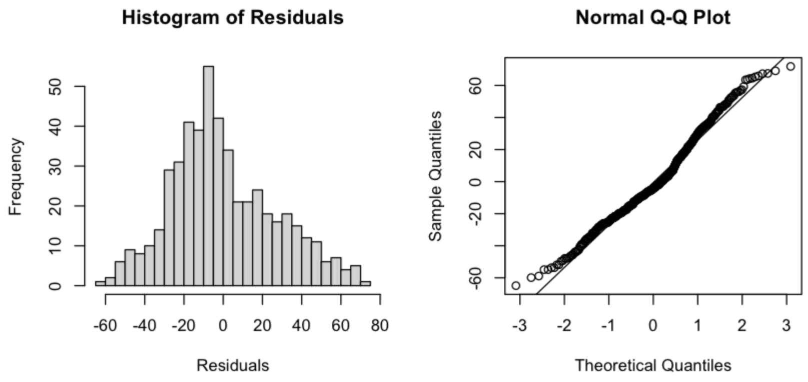
In [30]:
```
# Create diagnostic plots
par(mfrow=c(2,2))

# histogram of residuals
hist(bp_model$residuals,
     main="Histogram of Residuals",
     xlab="Residuals",
     breaks=20)

# normal Q-Q plot
qqnorm(bp_model$residuals)
qqline(bp_model$residuals)

# summary of residuals
summary(bp_model$residuals)
```

```
   Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
-64.897 -18.425  -4.337    0.000  17.308  71.845 
```



Overall, the model is lacking; while the predictors are statistically significant and play a role in predicting SystolicBP, there are clearly other factors that are just as or even more important in predicting SystolicBP. Some other variables that are likely important given what we know about general health and blood pressure:

- alcohol intake
- sleep
- genetic predisposition and history
- stress levels
- exercise (intensity, consistency, and duration)
- dietary factors (not just general intake, but specific foods like salt)

c. Cross validation: repeat the creation of a model, by first randomly splitting the dataset into about 75% to be used for training the model and remainder to be used for testing the model.

In [32]:
```
set.seed(42)

train_index <- sample(1:nrow(blood_data), 0.75 * nrow(blood_data))

train_data <- blood_data[train_index, ]
test_data <- blood_data[-train_index, ]

train_model <- lm(SystolicBP ~ ISmoke + IOverweight + IObese, data = train_d
summary(train_model)
```

```
Call:
lm(formula = SystolicBP ~ ISmoke + IOverweight + IObese, data = train_data)

Residuals:
    Min      1Q  Median      3Q     Max 
-59.308 -19.062  -4.062  18.027  71.362 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  131.308      2.697  48.683  < 2e-16 ***
ISmoke         8.475      2.768   3.062  0.00236 ** 
IOverweight    9.330      3.626   2.573  0.01046 *  
IObese        18.279      3.175   5.758 1.79e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 26.6 on 371 degrees of freedom
Multiple R-squared:  0.1099,	Adjusted R-squared:  0.1027 
F-statistic: 15.26 on 3 and 371 DF,  p-value: 2.193e-09
```

d. Discuss the differences between the coefficients for the first model and the model you just created in part c.

For the most part, the results are pretty similar - we see similar p-values, relatively close estimates (though IObese's coefficient has increased), and a statistically significant model that still only explains roughly 11% of the data.

e. Similar to problem 2 above, use the predict() command to calculate the predicated values and also the residuals for the testing set. Do a summary() of the residuals and describe how well the model seems to predict on the testing data.

In [33]:
```r
test_predictions <- predict(train_model, newdata = test_data)
test_residuals <- test_data$SystolicBP - test_predictions

print("Summary of test set residuals:")
print(summary(test_residuals))

test_correlation <- cor(test_data$SystolicBP, test_predictions)
print(paste("Correlation between predicted and actual values:", round(test_c
```

```
[1] "Summary of test set residuals:"
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
-64.308 -19.308  -5.783  -1.426  15.692  69.692 
[1] "Correlation between predicted and actual values: 0.2531"
```

Overall, the model's predictive performance on the test set is poor, which aligns with our earlier assessment that it was only picking up roughly 10% of the variance in the data, and that other factors likely have a more important role. The residuals span from -64 to 69, indicating large prediction errors, and the correlation of .2531 between predicted and actual values is weak. Squaring the correlation gives us a value of .0641, indicating that our model only accounts for about 6.4% of the variance in the training set, an even worse performance than the 11% we observed on our training set.