1.2

a)

```
tob = read.table("tobacco.txt", header=T, sep="\t")
set.seed(123)

fit = lm(tob[,14]~ tob[,5], data=tob)
summary(fit)

plot(tob[,5], tob[,14], xlab="CONSUMPTION", ylab="ILL")
abline(fit)
```

b)

```
beta1 = cor(tob[,14], tob[,5]) *  sd(tob[,14]) / sd(tob[,5])
beta0 = mean(tob[,14]) - beta1 * mean(tob[,5])
rbind(c(beta0, beta1), coef(lm(tob[,14] ~ tob[,5])))
```

OUTPUT:

```
      (Intercept) tob[, 5]
[1,]    65.74886 0.2291153
```

Output tells us that the interception point is 65.8 and the slope is 0.23.

c)

Coefficient of determination of the model is R-squared = 0.549. It is a measure of goodness of the model.

d)

```
var.test(tob[,14], tob[,5])
```

OUTPUT:

```
F = 0.09561, num df = 10, denom df = 10, p-value = 0.0009439
```

Since the p-value < 0.01 the model is statistically significant according to F-test with the level of significance at 1%. The values have similar variance.
`summary(fit)` gives F-statistic and p-values that are approx one magnitude larger than with `var.test`, but p-value is still less than 0.001. `f = ((n-k-1)/k)*ssm/sse` is also an option to acquire the F-statistic.

e)

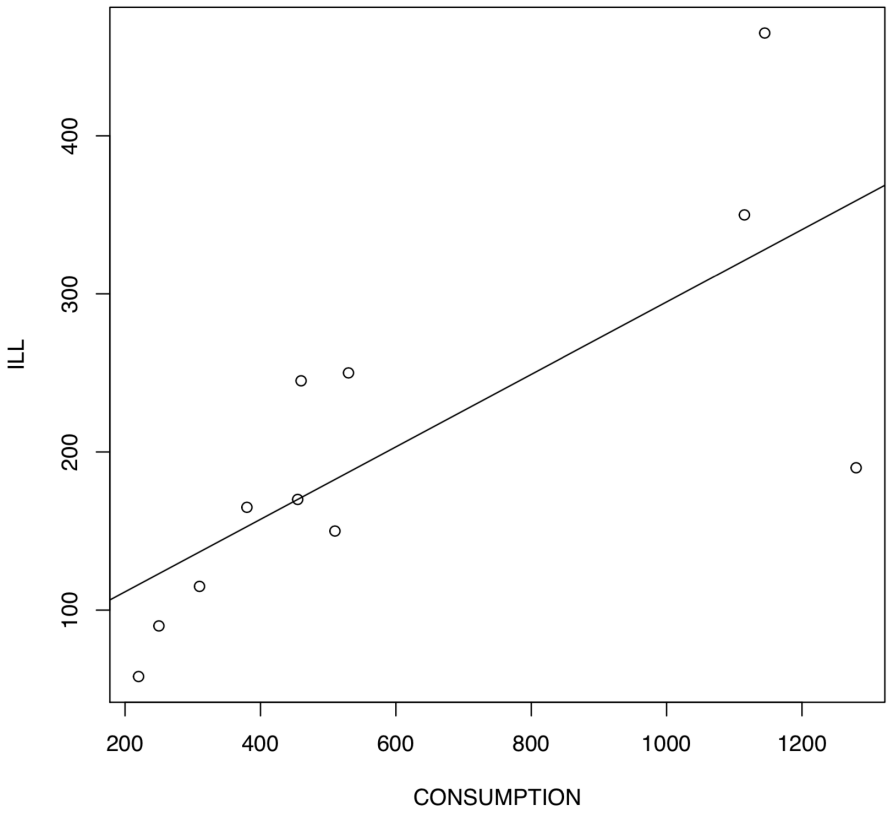
```
t.test(tob[,14], tob[,5])
```

OUTPUT:

```
t = 3.3027, df = 11.895, p-value = 0.006379
```

Since the p-value < 0.01 the model is statistically significant according to t-test with the level of significance at 1%. Hence, values have similar mean. The p-value of t-test is higher than that of a F-test. The difference between these tests is that F-test tells us about the variability of the samples and t-test is testing if the samples have the same mean, assuming the normality of the distribution.

f)



g)

Confidence interval of level (1-b) for parameter x is a random interval that contains the true value of the of x with the probability of (1-b).

h)

```
confinterval95 = confint(fit, level=0.95)
confinterval99 = confint(fit, level=0.99)
```

OUTPUT:

```
                2.5 %      97.5 %
(Intercept) -45.00344053 176.5011546
tob[, 5]      0.07254024   0.3856904

                0.5 %      99.5 %
(Intercept) -93.358900306 224.8566143
tob[, 5]      0.004178126   0.4540525
```

The confidence interval is notably wide for the constant term because large variations in intercept that lead to big errors in regression line can be “fixed” with relatively small changes to the slope of the regression line.

i)

```
tmp = as.matrix(tob[,c(14)])
n = nrow(tob)
Intercept = rep(1,n)
tob2 = cbind(Intercept,tmp)

k = 2000
bootmat = matrix(NA,nrow=k,ncol=2)
y = tob[,5]
X = tob2

set.seed(123)
for(i in 1:(k-1)){

  ind = sample(1:n,replace = TRUE)
  Xtmp = X[ind,]
  ytmp = y[ind]

  btmp = solve(t(Xtmp)%*%Xtmp)%*%t(Xtmp)%*%ytmp
  bootmat[i,] = t(btmp)
}

boriginal = solve(t(X)%*%X)%*%t(X)%*%y
bootmat[k,] = t(boriginal)
boot1 = sort(bootmat[,1])
boot2 = sort(bootmat[,2])

qconst = quantile(boot1, probs = c(0.025,0.975))
qcons = quantile(boot2, probs = c(0.025,0.975))

hist(bootmat[,1])
hist(bootmat[,2])
qconst
qcons
```

OUTPUT:

```
    2.5%     97.5%
-81.10919 417.10609
    2.5%    97.5%
1.360024 3.711072
```

These intervals seem a little off, especially for the slope of the line. I maybe have something wrong in my code but at least I can’t find the mistake :(

j)

Bootstrapping is great for larger datasets and larger number of bootstrap samples. It is also free of distribution assumptions.