1)

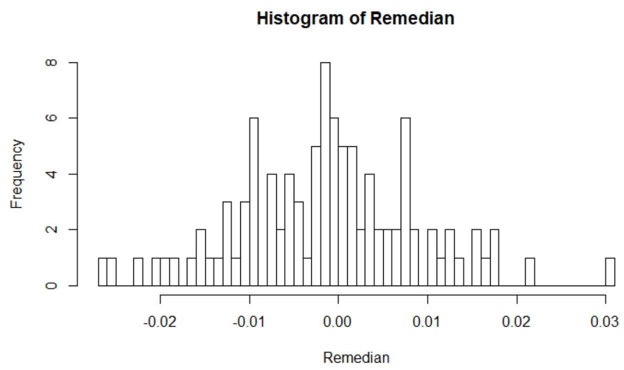
```
> Score<-function(x)
+ {
+     f=c(5/x+log(0.1)+log(0.25)+log(0.5)+log(2)-0.1**x*log(0.1)-0.25**x*log(0.25)-0.5**log(0.5)-2
**x*log(2),-5/x**2-0.1**x*log(0.1)**2-0.25**x*log(0.25)**2-0.5**log(0.5)**2-2**x*log(2)**2)
+     return(f)
+
+ }
>
> newtonraphson <- function(Score, x0,tol=1e-9,max.iter=100)
+ {
+     x<-x0;
+     fx<-Score(x)
+     iter<-0
+     while (((abs(fx[1])>tol))&&(iter<max.iter))
+     {
+         x<-x-fx[1]/fx[2]
+         fx<-Score(x)
+         iter=iter+1
+         cat("At iteration",iter,"value of x is :",x,"\n")
+     }
+     if (abs(fx[1])>tol)
+     {
+         cat("Algorithm failed to converged\n")
+         retrun(NULL)
+
+     }
+     else
+     {
+         cat ("Algorithm converged\n ")
+         return(x)
+     }
+ }

> S<-function(x)
+ {
+     s=5/x+log(0.1)+log(0.25)+log(0.5)+log(2)-0.1**x*log(0.1)-0.25**x*log(0.25)-0.5**log(0.5)-2**
x*log(2)
+     return(s)
+ }
> x<-seq(0,1,0.1)
> S(x)
 [1]        Inf 46.9872648 21.4015445 12.5762557  7.9925933  5.1353443  3.1588354  1.6958836
 [9]  0.5597163 -0.3556021 -1.1151484
> x=seq(0.8,0.9,0.01)
> S(x)
 [1]  0.55971627  0.45955874  0.36150054  0.26546776  0.17138982  0.07919931 -0.01116822
 [8] -0.09977440 -0.18667820 -0.27193608 -0.35560214
> x=seq(0.85,0.86,0.001)
> S(x)
 [1]  0.079199309  0.070081560  0.060981976  0.051900494  0.042837051  0.033791587  0.024764038
 [8]  0.015754344  0.006762443 -0.002211726 -0.011168222
> newtonraphson(Score,x0=0.86,tol=1e-9,max.iter = 100)
At iteration 1 value of x is : 0.8588444
At iteration 2 value of x is : 0.8587601
At iteration 3 value of x is : 0.8587539
At iteration 4 value of x is : 0.8587534
At iteration 5 value of x is : 0.8587534
At iteration 6 value of x is : 0.8587534
At iteration 7 value of x is : 0.8587534
Algorithm converged
 [1] 0.8587534
```

The MLE for λ is 0.8587534.

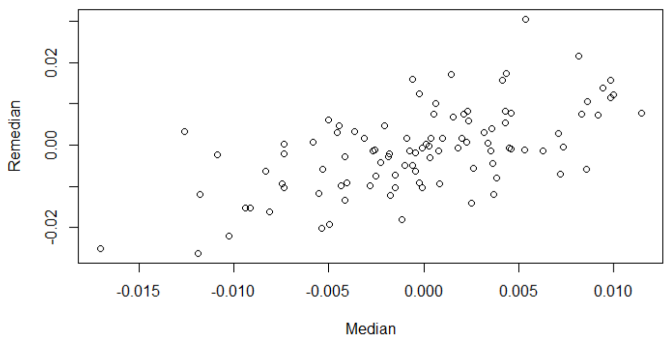
5)

```
> remedian<-function(b=15,k=4,x)
+ {
+     b1<-c()
+     b2<-c()
+     b3<-c()
+     b4<-c()
+     p=0
+     for(i in 1:b)
+     {
+         for(j in 1:b)
+         {
+             for(k in 1:b)
+             {
+                 for(l in 1:b)
+                 {
+                     p<-p+1
+                     b1[l]<-x[p]
+                 }
+                 b2[k]<-median(b1)
+             }
+             b3[j]<-median(b2)
+         }
+         b4[i]<-median(b3)
+     }
+     return(median(b4))
+ }
```

```
> hist(Remedian,breaks=50)
```




```
> plot(Remedian~Median)
```




```
> set.seed(1111)
> Remedian<-c()
> Median<-c()
> for(i in 1:100)
+ {
+     x<-rnorm(50625)
+     Remedian[i] <-remedian(15,4,x)
+     Median[i] <- median(x)
+ }
> summary(Remedian)
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-0.026312 -0.007614 -0.001246 -0.001197  0.004855  0.030589
> summary(lm(Remedian~Median))

Call:
lm(formula = Remedian ~ Median)

Residuals:
       Min         1Q     Median         3Q        Max
-0.0158862 -0.0049369 -0.0000583  0.0046883  0.0258383

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.0009844  0.0008167  -1.205    0.231
Median       1.0684235  0.1435034   7.445 3.79e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.008162 on 98 degrees of freedom
Multiple R-squared:  0.3613,    Adjusted R-squared:  0.3548
F-statistic: 55.43 on 1 and 98 DF,  p-value: 3.794e-11
```

From the above linear regression model of Remedian and True Medain, we see the F-statistic is significant. The intercept is not significant, and the coefficient is significant and approximate to 1. So we can say Remedian is nearly equal to the True Median and we can use Remedian to estimate Median directly.

6) b.

```
> P<-(seq(1,20)-10.5)**2/(10+seq(1,20))
> W<-seq(1,20)/(1+seq(1,20))
> M=15
> Profit<-c()
> Weight<-c()
> BestX<-c()
> BestP=0

> set.seed(999)
>
> for (i in seq(1,1000))
+ {
+     X=rbinom(20,1,0.5)
+     Profit[i]=sum(X*P)
+     Weight[i]=sum(X*W)
+     if (Profit[i]>BestP & Weight[i]<M)
+     {
+         BestP=Profit[i]
+         BestX<-X
+     }
+ }
> BestP
[1] 190
> BestX
 [1] 1 0 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1
```

The randomized suboptimal solution is X=[1 0 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1]

with profit=190.