A4Q1

a)

```
digital <- read.csv('digitData.csv',header=T)
digitSample <- c(294,133,95,265,154,1,289,232,121,99,129,83,30,56,249,134,46,68,
                 165,279,105,91,248,285,238,45,194,34,44,5,173,87,18,299,167,
                 64,42,266,281,210,27,207,271,181,6,212,176,51,28,243)
```

```
n = length(digitSample)
N = popSize(digital)
inclusionProb <- createInclusionProbFn(1:N, sampSize = n)
inclusionJointProb <- createJointInclusionProbFn(1:N, sampSize = n)

DigitalHTestimator <- createHTestimator(inclusionProb)
HTVarianceEstimator <- createHTVarianceEstimator(1:N,
                         pi_u_fn = inclusionProb,
                        pi_uv_fn = inclusionJointProb)
```

```
createVariateFnAvgBrightness <- function(popData, variate1, N=1, y=NULL) {
  function (u) { popData[u, variate1]/N }
}
```

The HT Estimator is:

```
DigitalAvgBrightness <- createVariateFnAvgBrightness(digital, "Brightness", N=N)
DigitalHTestimator(digitSample, DigitalAvgBrightness)
```

```
## [1] 26.60653
```

An estimate of the variance or the standard error is:

```
sqrt(HTVarianceEstimator(digitSample, DigitalAvgBrightness))
```

```
## [1] 1.337042
```

b)

```
createvariateFnNy <- function(popData, variate1, N=1, y=NULL) {
  function (u) { (popData[u, variate1] <= y )/N}
}

propDigitalBrightness45 =createvariateFnNy(digital, "Brightness", N=N, y=45)
```

The HT estimate is:

```
DigitalHTestimator(digitSample, propDigitalBrightness45)
```

```
## [1] 0.94
```

c)

```
createvariateFnrange <- function(popData, variate1, N=1, y1=NULL, y2=NULL) {
  function (u) { (popData[u, variate1] >= y1 & popData[u, variate1] < y2)/N}
}

propDigitalBrightnessrange =createvariateFnrange(digital, "Brightness",
                                                 N=N, y1=20, y2=25)
```

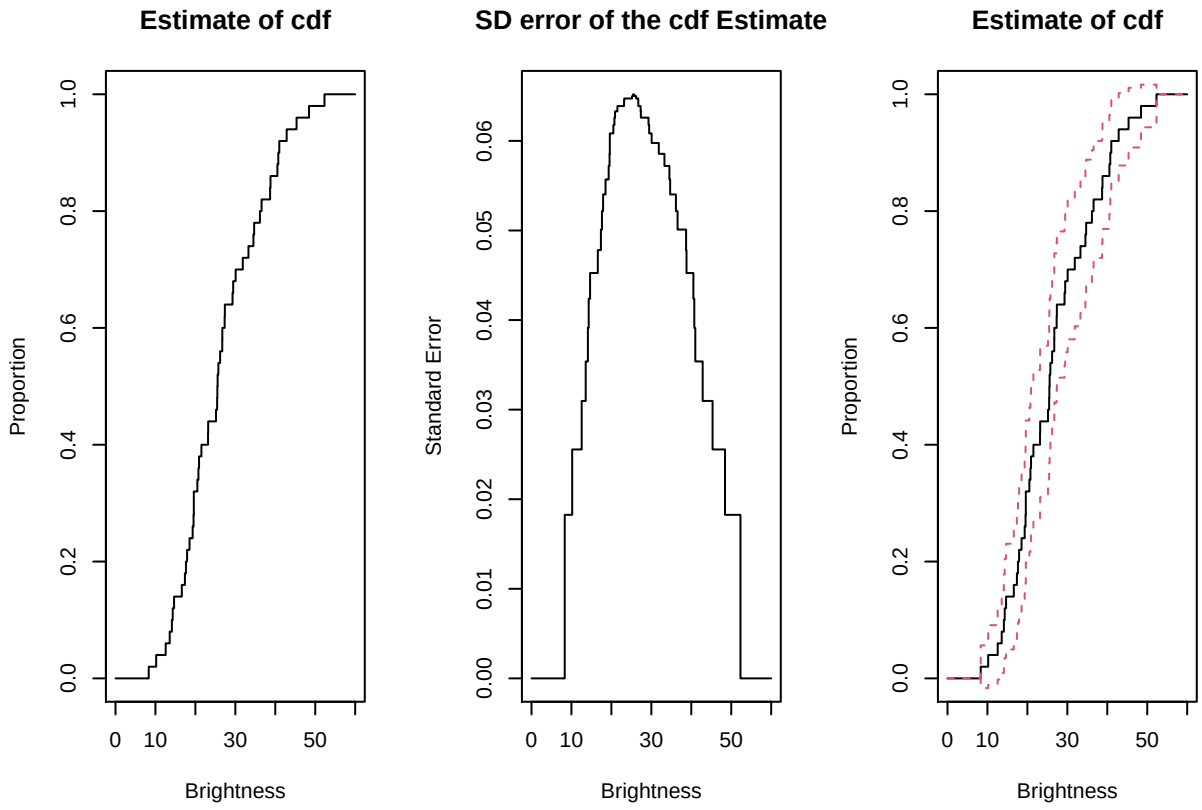
HT estimate of the proportion of digits with brightness in the interval [20, 25) is:

```
DigitalHTestimator(digitSample, propDigitalBrightnessrange)
```

```
## [1] 0.12
```

d)

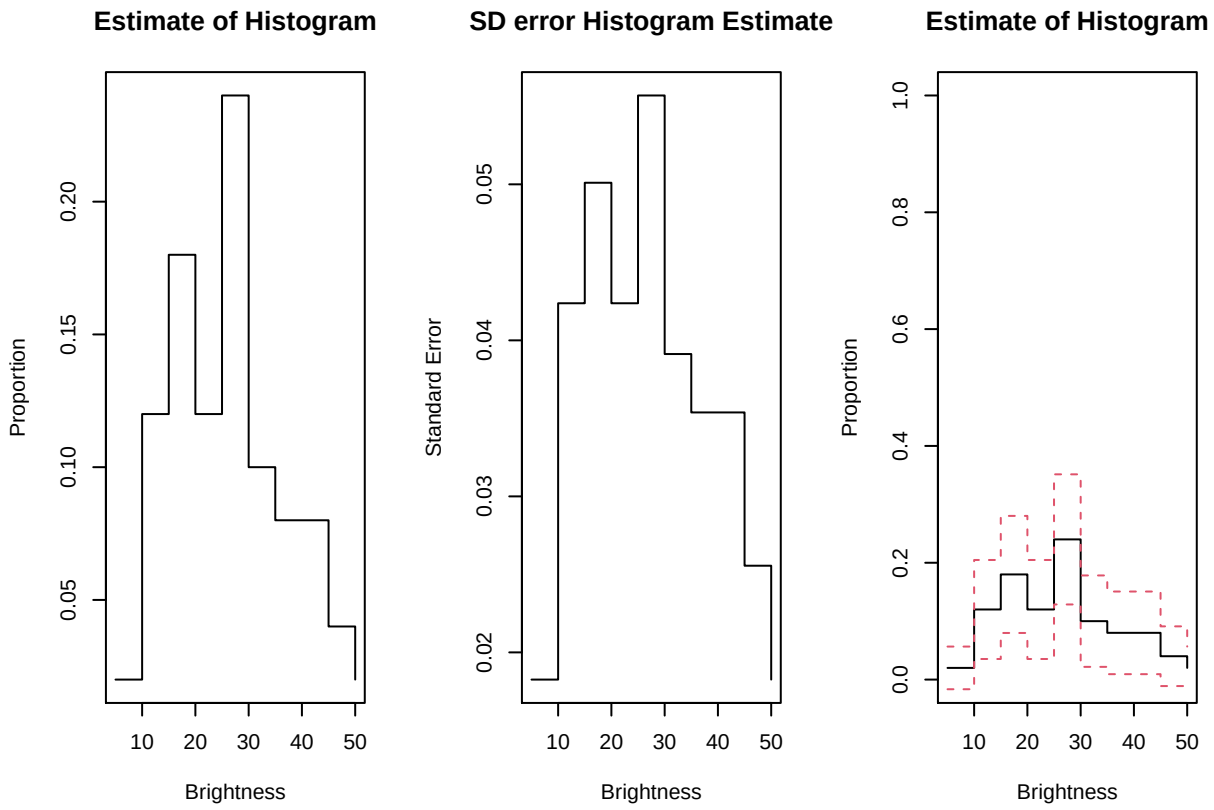
```
yseq = c(0, sort(digital$Brightness[digitSample]), 60)

cdf.estimate.sd = sapply(yseq, function(y) {
  propBrightness <- createvariateFnNy(digital, "Brightness", N=N, y=y)

  c( DigitalHTestimator(digitSample, propBrightness),
   sqrt( round(HTVarianceEstimator(digitSample, propBrightness), 14) ) )
   } )


par(mfrow=c(1,3) )
plot(yseq, cdf.estimate.sd[1,], type='s', ylab="Proportion",
     xlab="Brightness",
     main="Estimate of cdf")
plot(yseq, cdf.estimate.sd[2,], type='s', ylab="Standard Error",
     xlab="Brightness",
     main="SD error of the cdf Estimate")

plot(yseq, cdf.estimate.sd[1,], type='s', ylim=c(0,1),
     ylab="Proportion", xlab="Brightness",
     main="Estimate of cdf")

cdf.lower = cdf.estimate.sd[1,] - 2*cdf.estimate.sd[2,]
cdf.upper = cdf.estimate.sd[1,] + 2*cdf.estimate.sd[2,]

lines(yseq, cdf.lower, type='s',col=2, lty=2)
lines(yseq, cdf.upper, type='s',col=2, lty=2)
```



e)

```
yseq = cbind(seq(5, 50, 5), seq(10, 55, 5))

hist.estimate.sd = apply(yseq, 1, function(y) {
  propBrightness <- createvariateFnrange(digital, "Brightness", N=N, y1=y[1],
                                         y2=y[2])

  c( DigitalHTestimator(digitSample, propBrightness),
   sqrt( round(HTVarianceEstimator(digitSample, propBrightness), 14) ) )
   } )


par(mfrow=c(1,3) )
plot(yseq[, 1], hist.estimate.sd[1,], type='s', ylab="Proportion",
     xlab="Brightness",
     main="Estimate of Histogram")
plot(yseq[, 1], hist.estimate.sd[2,], type='s', ylab="Standard Error",
     xlab="Brightness",
     main="SD error Histogram Estimate")

plot(yseq[, 1], hist.estimate.sd[1,], type='s', ylim=c(0,1),
     ylab="Proportion", xlab="Brightness",
     main="Estimate of Histogram")
```

```
hist.lower = hist.estimate.sd[1,] - 2*hist.estimate.sd[2,]
hist.upper = hist.estimate.sd[1,] + 2*hist.estimate.sd[2,]

lines(yseq[, 1], hist.lower, type='s',col=2, lty=2)
lines(yseq[, 1], hist.upper, type='s',col=2, lty=2)
```



f)

```
digit.prop <- createVariateFnAvgBrightness(digital, "Digit1", N=N)
```

The HT Estimator is:

```
DigitalHTestimator(digitSample, digit.prop)
```

```
## [1] 0.62
```

An estimate of the variance or the standard error is:

```
sqrt(HTVarianceEstimator(digitSample, digit.prop))
```

```
## [1] 0.06329931
```