Project 1
MFE 405: Computational Finance
Professor Goukasian
Students: Xiahao Wang

This is a summary of the project for data visualisation, for detail implementation and result,

please refer to the print out of the program

Qn 1.

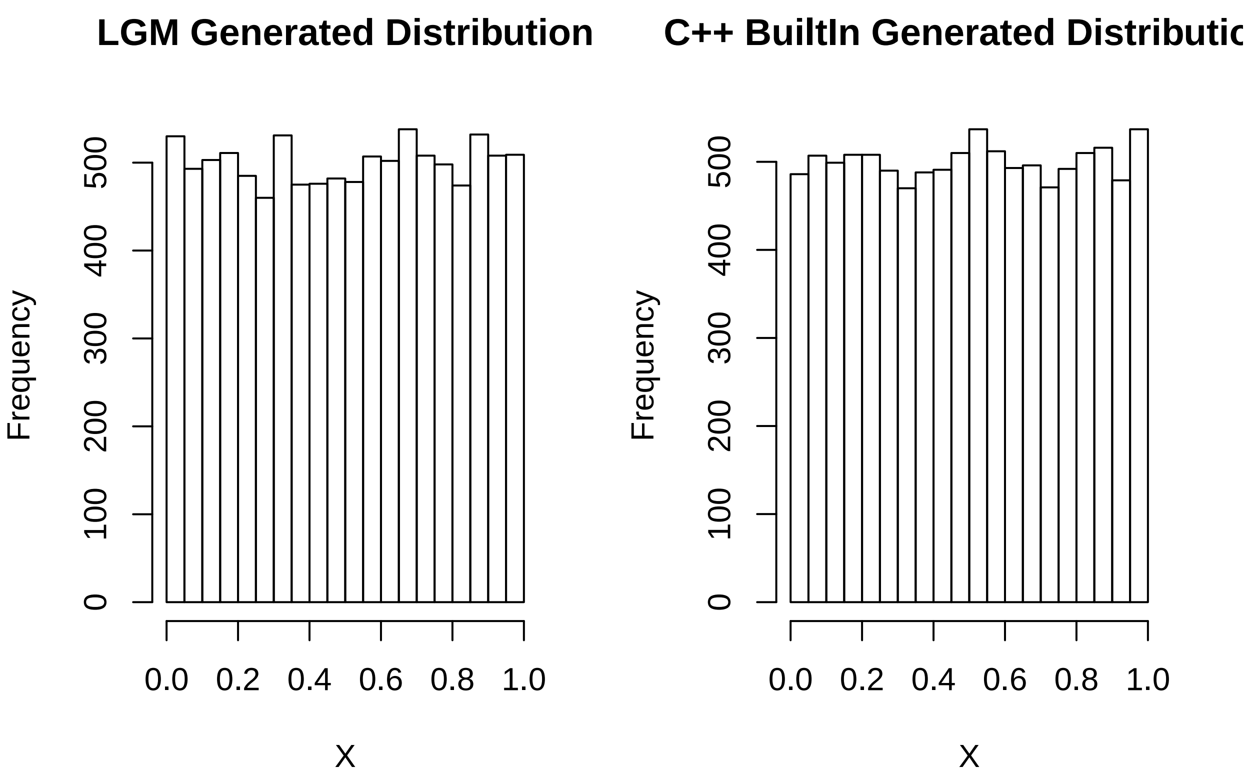
```
library(ggplot2)
library(data.table)

Q1LGM <- read.csv("~/Documents/ucla/Dropbox/Quarter2/Computational Finance/HW/computationalFinanceProje

Q1BuiltIn <- read.csv("~/Documents/ucla/Dropbox/Quarter2/Computational Finance/HW/computationalFinancePr

par(mfrow = c(1,2))

hist(Q1LGM$V1, main = "LGM Generated Distribution", xlab = "X")
hist(Q1BuiltIn$V1, main = "C++ BuiltIn Generated Distribution", xlab = "X")
```




```
Q1DT <- data.table(
  Moments = c("Mean", "Standard Deviation"),
  LGM = c(mean(Q1LGM$V1), sd(Q1LGM$V1)),
  Cplusplus = c(mean(Q1BuiltIn$V1), sd(Q1BuiltIn$V1))
```

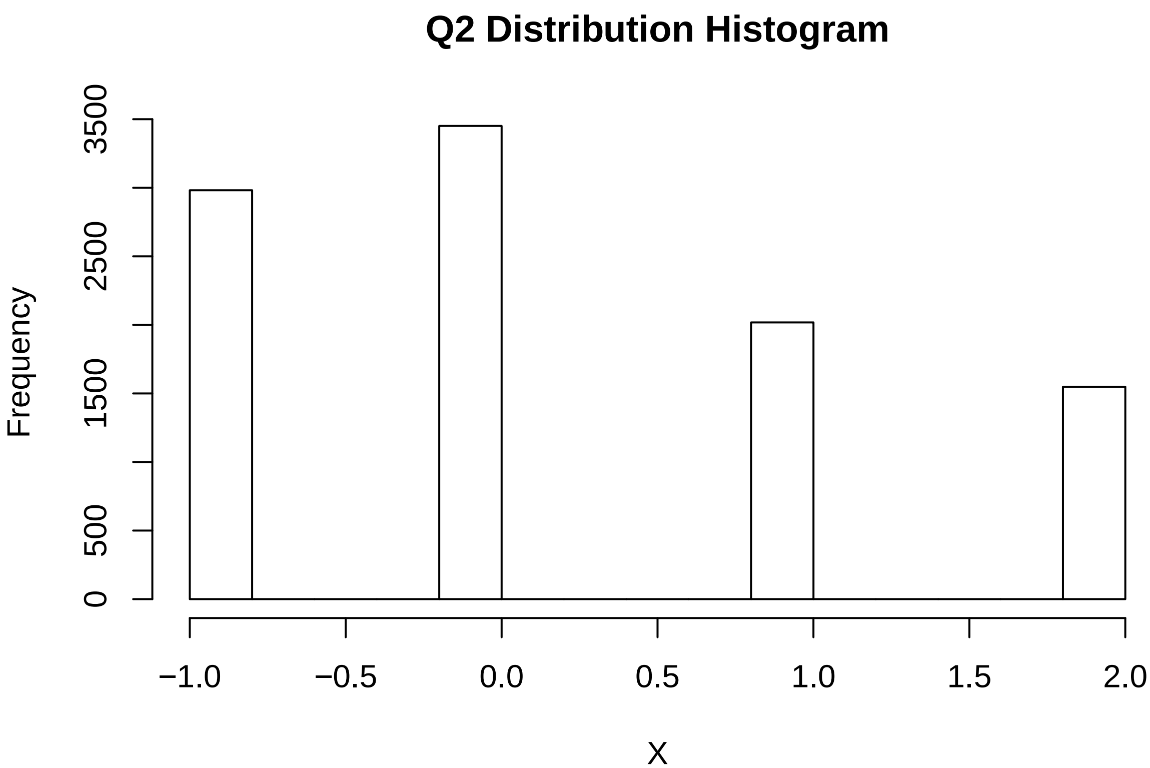
```
)
Q1DT
```

```
##                Moments       LGM Cplusplus
## 1:                Mean 0.5017506 0.5018268
## 2: Standard Deviation 0.2904058 0.2892272
```

From the comparsion above, distribution generated from LGM algorithm is very similar to the one done by C++ builtin function

Qn 2

```
Q2Data <- read.csv("~/Documents/ucla/Dropbox/Quarter2/Computational Finance/HW/computationalFinanceProj
hist(Q2Data$V1, main = "Q2 Distribution Histogram", xlab = "X")
```



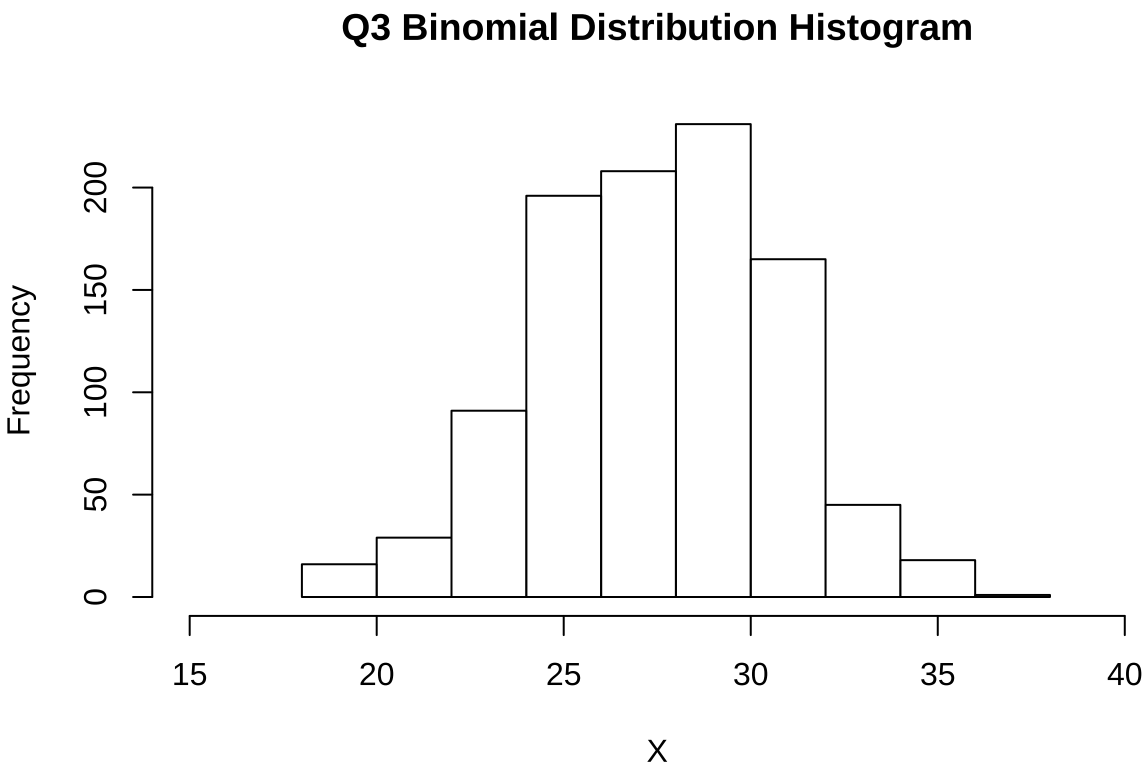

```
Q2DT <- data.table(
  Moments = c("Mean", "Standard Deviation"),
  LGM = c(mean(Q2Data$V1), sd(Q2Data$V1))
)
Q2DT
```

```
##                Moments      LGM
## 1:                Mean 0.213400
## 2: Standard Deviation 1.036421
```

Qn3

```r
Q3Data <- read.csv("~/Documents/ucla/Dropbox/Quarter2/Computational Finance/HW/computationalFinanceProj

hist(Q3Data$V1, main = "Q3 Binomial Distribution Histogram", xlab = "X", xlim = c(15,40))
```



Calculate p(X>=40) in R:

```r
1 - pbinom(39,size = 44, p=0.64)
```

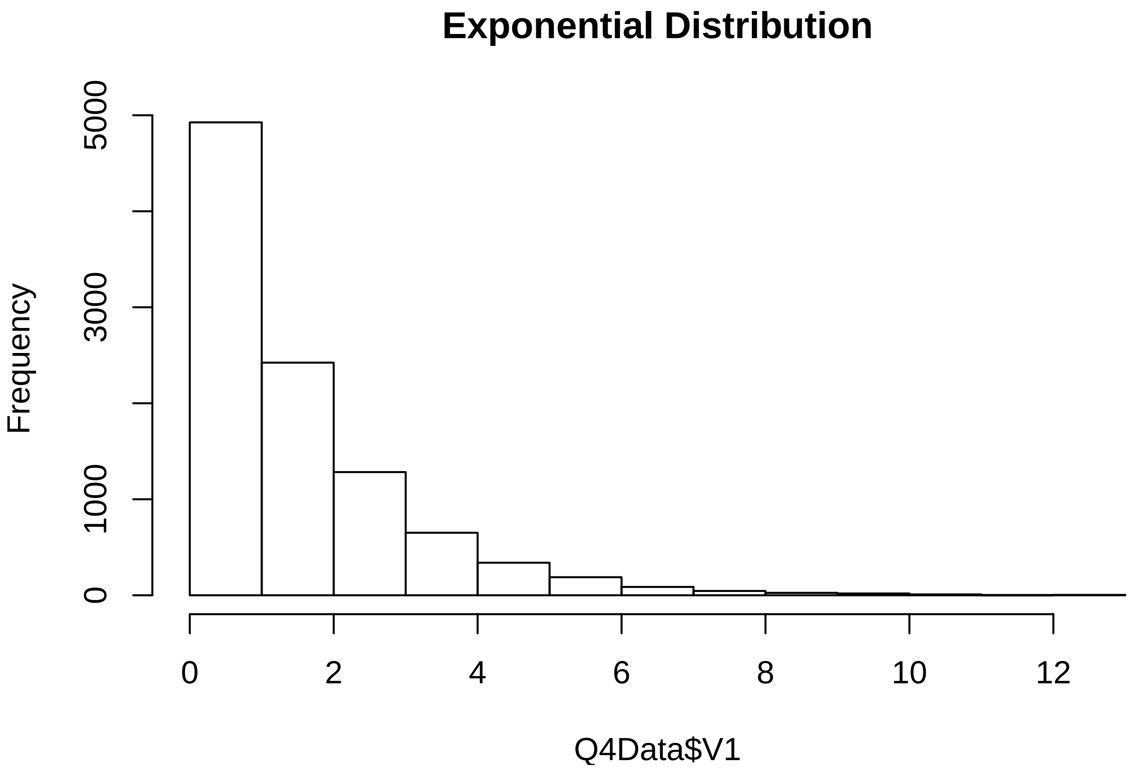
```
## [1] 4.823664e-05
```

In C++ implementation the probablity is 0, which is close to the result we get from R

Qn4

```r
Q4Data <- read.csv("~/Documents/ucla/Dropbox/Quarter2/Computational Finance/HW/computationalFinanceProj

hist(Q4Data$V1, main="Exponential Distribution")
```



From C++ implementation: $P(X \geqslant 1) = 0.5074$ $P(X \geqslant 4) = 0.0717$

```
Q4DT <- data.table(
  Moments = c("Mean", "Standard Deviation"),
  LGM = c(mean(Q4Data$V1), sd(Q4Data$V1))
)
Q4DT
```

```
##                Moments      LGM
## 1:                Mean 1.505545
## 2: Standard Deviation 1.527231
```

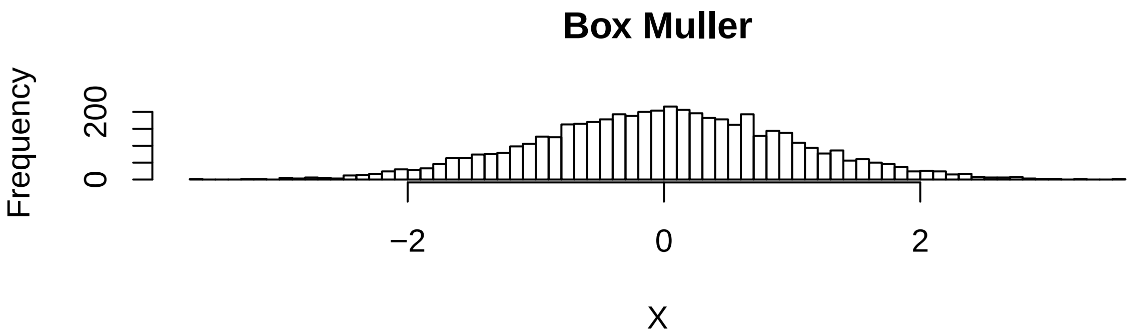
Qn5

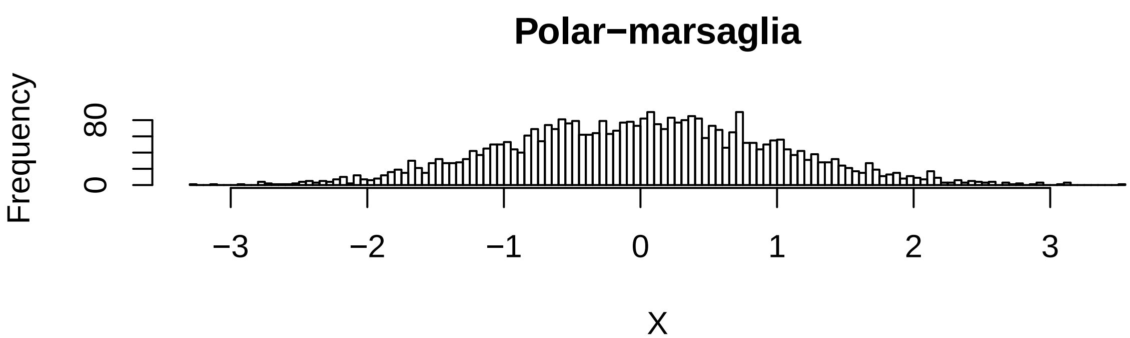
Compare normal distribution generated by Box-Muller and Polar-marsaglia

```
Q5Box <- read.csv("~/Documents/ucla/Dropbox/Quarter2/Computational Finance/HW/computationalFinanceProje

Q5PM <- read.csv("~/Documents/ucla/Dropbox/Quarter2/Computational Finance/HW/computationalFinanceProjec

par(mfrow= c(2,1))
hist(Q5Box$V1, main="Box Muller", breaks=100,xlab="X")
hist(Q5PM$V1, main="Polar-marsaglia",breaks =100, xlab="X")
```






```
Q5DT <- data.table(
  Moments = c("Mean", "Standard Deviation"),
  BoxMuller = c(mean(Q5Box$V1), sd(Q5Box$V1)),
  PolarMarsaglia = c(mean(Q5PM$V1), sd(Q5PM$V1))
)
Q5DT
```

```
##                Moments   BoxMuller PolarMarsaglia
## 1:                Mean -0.00601808     0.00285695
## 2: Standard Deviation  0.99451087     0.97764315
```

Based on my implementation Polar-marsaglia is a faster algorithm:

Comparsion

Simulation with data size: 5000

Time taken for Box-Muller: 660ms

Time taken for Polar-Marsaglia: 361ms

Clearly Polar-Marsaglia is faster than Box-Muller