In [3]:
```
library("tidyverse")
```

```
── Attaching packages ─────────────────────────────────────── tidyverse
1.3.0 ──

✔ ggplot2 3.3.3     ✔ purrr   0.3.4
✔ tibble  3.0.5     ✔ dplyr   1.0.3
✔ tidyr   1.1.2     ✔ stringr 1.4.0
✔ readr   1.4.0     ✔ forcats 0.5.1

── Conflicts ──────────────────────────────────────────── tidyverse_confl
icts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
```

In [4]:
```
##instantiate all the variables

##create a data frame with the closing prices
df <- read_csv("13.5.csv")

sigma <- 0.2 #the variance parameter

r <- 0.01

t <- 252 #there are 252 trading days in a year

K <- 52 #strike price

print(sigma)
print(r)
print(t)
```

```
── Column specification ────────────────────────────────────────────
────────

cols(
  i = col_double(),
  Zi = col_double(),
  Si = col_double()
)


[1] 0.2
[1] 0.01
[1] 252
```

In [5]:
```
mu <- (0.01 - (sigma^2)/2 ) #drift parameter

print(mu)
```

```
[1] -0.01
```

In [6]:
```
df <- df %>% mutate(i = as.integer(i))
```

In [7]:
```
#write a function that transforms the Zi normal to the price generated
normal_standard <- function(Zi, mu, sigma, t){

    #multiply Zi by the variance and devide by sqrt of 252 trading days
    xi <- Zi*(sigma/sqrt(t))

    #then add mu divided by the variance

    xi <- xi + (mu/252)

}

#test the function

xi <- normal_standard(-0.25,mu,sigma, t)

print(xi)
```

```
[1] -0.003189386
```

In [8]:
```
#map the function over the data frame

df<- df%>% mutate(Xi = normal_standard(Zi, mu, sigma, t))

df %>% head(7)
```

A tibble: 7 × 4

i	Zi	Si	Xi
<int>	<dbl>	<dbl>	<dbl>
0	NA	50	NA
1	-0.25	NA	-0.003189386
2	0.30	NA	0.003739962
3	1.50	NA	0.018858541
4	-1.20	NA	-0.015158261
5	-1.65	NA	-0.020827729
6	1.50	NA	0.018858541

In [10]:
```
#loop over the Xi vector to calculate closing prices
for (i in 1:nrow(df)){

    lead = i +1
    df$Si[[lead]] <- df$Si[[i]]*exp(1)^df$Xi[[lead]]
}
```

```
[1] 1
[1] 2
[1] 50
[1] -0.003189386
[1] 49.84078
[1] 2
[1] 3
```

```
[1] 49.84078
[1] 0.003739962
[1] 50.02754
[1] 3
[1] 4
[1] 50.02754
[1] 0.01885854
[1] 50.97993
[1] 4
[1] 5
[1] 50.97993
[1] -0.01515826
[1] 50.213
[1] 5
[1] 6
[1] 50.213
[1] -0.02082773
[1] 49.17799
[1] 6
[1] 7
[1] 49.17799
[1] 0.01885854
[1] 50.11421
[1] 7
[1] 8
```

```
Error in df$Xi[[lead]]: subscript out of bounds
Traceback:

1. print(df$Xi[[lead]])
```

In [11]:

```
df %>% head(7)
```

A tibble: 7 × 4

i	Zi	Si	Xi
<int>	<dbl>	<dbl>	<dbl>
0	NA	50.00000	NA
1	-0.25	49.84078	-0.003189386
2	0.30	50.02754	0.003739962
3	1.50	50.97993	0.018858541
4	-1.20	50.21300	-0.015158261
5	-1.65	49.17799	-0.020827729
6	1.50	50.11421	0.018858541

Part 2

Generate 6 more simulated daily closing prices

In [87]:
```
#generate 6 random points from a normal distribution

cent <- mu/t
dev <- (sigma^2)/t

print(cent)
print(dev)


#random normal generated values for Xi
rand_Xi <- rnorm(6, mean = cent, sd = dev)

print(rand_Xi)
```

```
[1] -3.968254e-05
[1] 0.0001587302
[1] -2.628115e-04  4.731965e-05  3.698051e-05 -1.415038e-05 -1.262229e-
04
[6] -9.184504e-05
```

In [88]:
```
#create an empty vector for Xi
Xi <- c()
#
Xi <- append(0,rand_vars)

print(Xi)
```

```
[1]  0.000000e+00 -1.381145e-05 -2.681216e-04  1.764453e-04  4.329975e-
05
[6]  1.345644e-04 -1.725527e-04
[1]  0.000000e+00 -1.381145e-05 -2.681216e-04  1.764453e-04  4.329975e-
05
[6]  1.345644e-04 -1.725527e-04
```

In [89]:
```
#create a dataframe with the new vectors
df2 <-tibble(i = rep(0:6),
            Xi = Xi,
            Si = rep(NA,7) ) #add the closing prices

#adding day 1 price
df2$Si[1] <- 50


df2 %>%head()
```

A tibble: 6 × 3

i	Xi	Si
<int>	<dbl>	<dbl>
0	0.000000e+00	50
1	-1.381145e-05	NA
2	-2.681216e-04	NA
3	1.764453e-04	NA
4	4.329975e-05	NA

i	Xi	Si

In [93]:
```
df2 %>% head(7)
```

A tibble: 7 × 3

i	Xi	Si
<int>	<dbl>	<dbl>
0	0.000000e+00	50
1	-1.381145e-05	NA
2	-2.681216e-04	NA
3	1.764453e-04	NA
4	4.329975e-05	NA
5	1.345644e-04	NA
6	-1.725527e-04	NA

In [99]:
```
for (n in 1:nrow(df2)){

    lead = n +1
    df2$Si[[lead]] <- df2$Si[[i]]*exp(1)^df2$Xi[[lead]]
}
```

```
[1] 0
[1] 50
[1] 1
[1] 49.99931
[1] 2
[1] 49.98591
[1] 3
[1] 49.99473
[1] 4
[1] 49.99689
[1] 5
[1] 50.00362
[1] 6
[1] 49.99499
```

In [100...
```
df2
```

A tibble: 7 × 3

i	Xi	Si
<int>	<dbl>	<dbl>
0	0.000000e+00	50.00000
1	-1.381145e-05	49.99931
2	-2.681216e-04	49.98591
3	1.764453e-04	49.99473
4	4.329975e-05	49.99689

i	Xi	Si

Problem 3

If the strike price of a European call is K =52, and the expiration of this call is at the end of 6 days, what is the payoff on the call?, that is what is the value of $(S_6 - K)^+$?

The answer is zero because S_6 is less than the strike price

Problem 4

Could you use the present value of $(S_6 - K)^+$ in 3) as an approximation of the cost on the call.

In short yes, but a better method is to run the simulation for multiple times and take the average of the day 6 pay offs to get a better approximation of the risk neutral payoff.

In [127…

```
#calculate the PV of the strike price

PvK <- K*exp(1)^(r *(-6/252))

print(PvK)
```

```
[1] 51.98762
```

In [128…

```
#instantiate an empty array
payout <- c()

#run the simulation 100 times
for (i in rep(1:10)){

    #print(i)

    #generate 6 closing prices from the distribution
    Xi <-rnorm(6,  mean = cent, sd = dev)

    #instantiate an empty array of closing prices
    Si <- rep(NA,6)

    #adding day 1 price
    Si[1] <- 50

    #loop through the entire closing prices and convert Sd from the ran
    for(n in rep(1:length(Si))){

        lead = n+1
        Si[lead] = Si[n]*exp(1)^Xi[lead]

        }

    #calculate the payout
    if (Si[6] < PvK){
        pay <- 0
    }else{
        pay <- (Si[6] - PvK)
    }

    payout<- append(payout, pay)




}


#average the payout over the number of times the simulation was ran
valuation <- sum(payout) / length(payout)

#print the valuation
print(valuation)
```

```
[1] 0
```

In [119…

6

In [ ]: