Final project coding

```
library(combinat)

##
## Attaching package: 'combinat'

## The following object is masked from 'package:utils':
##
##     combn

library(purrr)
library(tidyverse)

## ── Attaching packages ─────────────────── tidyverse 1.2.1 ──

## ✔ggplot2 3.0.0     ✔readr   1.1.1
## ✔tibble  1.4.2     ✔dplyr   0.7.6
## ✔tidyr   0.8.1     ✔stringr 1.3.1
## ✔ggplot2 3.0.0     ✔forcats 0.3.0

## ── Conflicts ──────────────────── tidyverse_conflicts() ──
## ✖dplyr::filter() masks stats::filter()
## ✖dplyr::lag()    masks stats::lag()

win_set <- matrix(c(1,2,3,4,5,6,7,8,9,1,4,7,2,5,8,3,6,9,1,5,9,3,5,7),
#matrix of all possible winning
                  byrow = F,nrow = 3)


#Starting with random playing
rand_strategy <- function(){
  #creating the flag of 9 spots to check whether the spot is occupied or not
  flag <- rep(0,9)
  code <- 0
  players <- matrix(data=0,nrow = 5,ncol = 2) #game starts here
  for(i in 1:9){ #spots where the first player playes in 5 spots and the
second one 4 spotsif the game completed with no winner
    play <- sample(1:9,1)
    while(flag[play]==1){        #here to check if the spot empty or resample
again
      play <- sample(1:9,1)
    }
    players[((i-1)/2+1),ifelse((i%%2)==1,1,2)] = play;  #
    flag[play]=1;
    if(i>4){
      result <- check_win(players,i)
      if(result$code != 0){
        return(result)
```

```
      }
    }
  }
  return(result)
}

check_win <-function(players,ind){   #This function to check the wins
  code <- 0
  ifelse(ind%%2==1,index <- 1,index <- 2)
  player_i <- sort(players[,index])
  player_i <- player_i[! player_i %in% c(0)]
  player_i_matrix <- combn(player_i,3)
  if(length(player_i) == 3){
    player_i_matrix <- matrix(player_i,ncol = 1)
  }
  for(i in 1:dim(win_set)[2]){
    for(j in 1:dim(player_i_matrix)[2]){
      match_vector <- match(win_set[,i],player_i_matrix[,j])
      if(any(is.na(match_vector)) == FALSE){
        ifelse(ind%%2==1,code <- 1,code <- -1)
        return(data.frame(code=code,index=ind))
      }
    }
  }
  return(data.frame(code=code,index=ind))
}
#Simulating 100,1000,and 10000 times for the random playing.
win_rate <- rerun(100,rand_strategy())
win_matrix_100 <- data.frame(matrix(unlist(win_rate), nrow=100, byrow=T))
(t <- table(win_matrix_100))
```

```
##     X2
## X1    5  6  7  8  9
##   -1  0  9  0 15  0
##   0   0  0  0  0 10
##   1  16  0 30  0 20
```

```
rate100 <- rowSums(t)[3]/sum(rowSums(t))
win_rate <- rerun(1000,rand_strategy())
win_matrix_1000 <- data.frame(matrix(unlist(win_rate), nrow=1000, byrow=T))
(t <- table(win_matrix_1000))
```

```
##     X2
## X1     5   6   7   8   9
##   -1   0  89   0 222   0
##   0    0   0   0   0 125
##   1   90   0 258   0 216
```

```
rate1000 <- rowSums(t)[3]/sum(rowSums(t))
win_rate <- rerun(10000,rand_strategy())
```

```r
win_matrix_10000 <- data.frame(matrix(unlist(win_rate), nrow=10000, byrow=T))
(t <- table(win_matrix_10000))
```

```
##     X2
## X1      5    6    7    8    9
##   -1    0  881    0 1959    0
##   0     0    0    0    0 1230
##   1   954    0 2707    0 2269
```

```r
rate10000 <- rowSums(t)[3]/sum(rowSums(t))

#creating data frame to plot the result and the index.

win_matrix <- data.frame(matrix(unlist(win_rate), nrow=10000, byrow=T))
names(win_matrix) <- c('result','index')
#making tables fpr the results.
table(win_matrix[which(win_matrix$result==1),]$index)
```

```
##
##    5    7    9
##  954 2707 2269
```

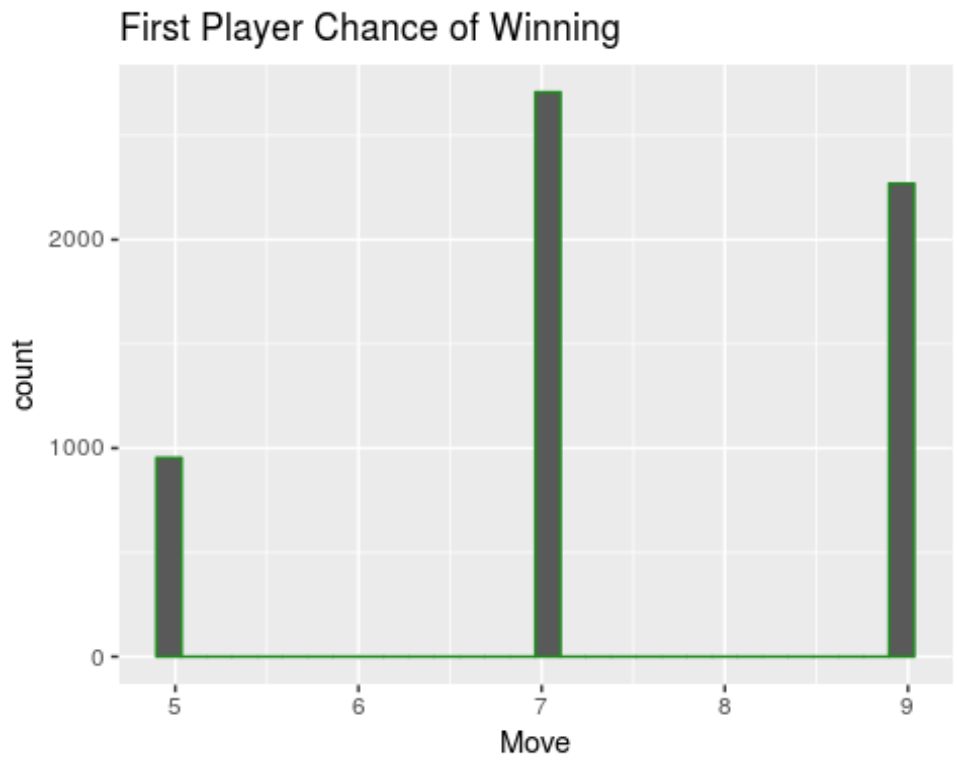
```r
table(win_matrix$result)
```

```
##
##   -1    0    1
## 2840 1230 5930
```

```r
#plotting first player chance of winning when the number of simulations is
10000
df_win1 <- data.frame(move = win_matrix[which(win_matrix$result==1),]$index)

ggplot(df_win1) +
  geom_histogram(mapping = aes(x = move), col = "forestgreen") +
  xlab("Move")+
  ggtitle("First Player Chance of Winning")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

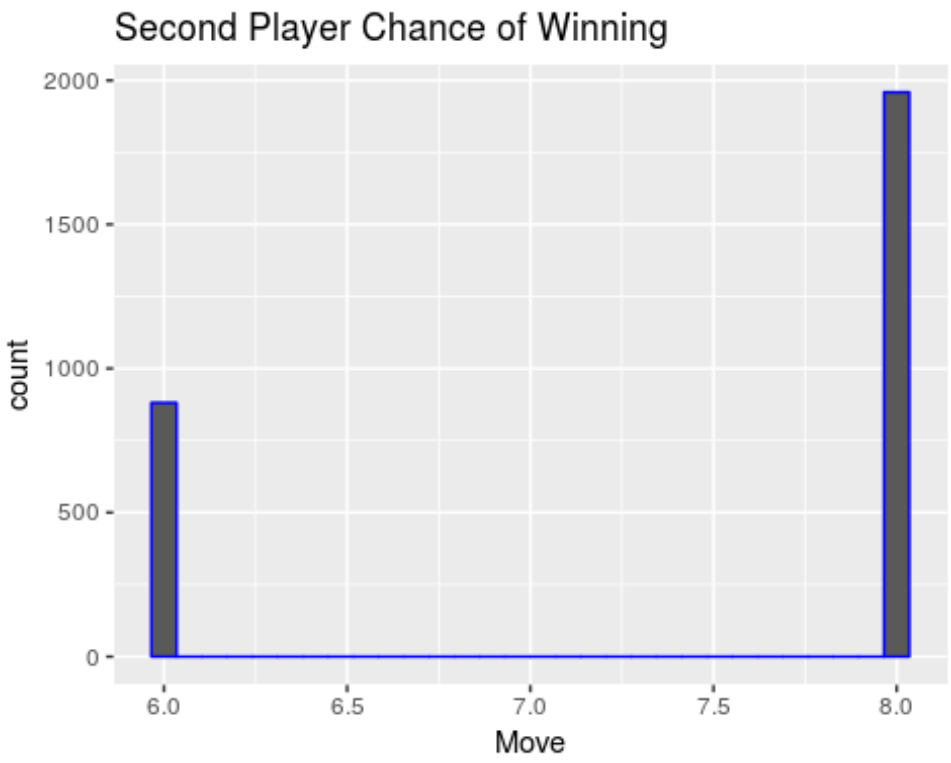



```
#plotting second player chance of winning when the number of simulations is
10000
#Make the -1 output a dataframe and plot

df_win2 <- data.frame(move = win_matrix[which(win_matrix$result==-1),]$index)


ggplot(df_win2) +
  geom_histogram(mapping = aes(x = move), col = "blue") +
  xlab("Move")+
  ggtitle("Second Player Chance of Winning")

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
##########################################
# Creating function where First player has a stratagy but the second one play
random:
strategy_player_1 <- function(){
  flag <- rep(0,9)
  code <- 0
  players <- matrix(data=0,nrow = 5,ncol = 2)
  for(i in 1:9){
    if(i%%2 == 0 | i==9){
      #player 2 going for random choice
      play <- sample(1:9,1)
      while(flag[play]==1){
        play <- sample(1:9,1)
      }
    }else{
      #player 1
      play <- strategy_move_p1(players,i)
    }
    players[((i-1)/2+1),ifelse((i%%2)==1,1,2)] = play;
    flag[play]=1;
    if(i>4){
      result <- check_win(players,i)
      if(result$code != 0){
        return(result)
      }
    }
  }
```

```
  return(result)
}

p1_second_move_matrix <- matrix(
  c(
    2, 9,
    4, 3,
    6, 7,
    8, 1,
    1, 3,
    9, 3,
    3, 1,
    7, 1
  ),byrow = T, ncol = 2
)

match_rows_second_move <- function(x, y){
  stopifnot(ncol(x) == ncol(y))
  stopifnot(nrow(y) == 1)
  matched <- which(x[,1] == y[1,1])
  x[matched,2]
}

p1_second_move <- function(players){
  p2_move <- players[1,2]
  p1_move <-
match_rows_second_move(p1_second_move_matrix,matrix(c(p2_move,NA),nrow = 1))
  p1_move
}

p1_third_move_matrix <- matrix(
  c(
    2, 1, 3,
    2, -1, 1,
    4, 7, 1,
    4, -1, 7,
    6, 3, 9,
    6, -1, 3,
    8, 9, 7,
    8, -1, 9,
    1, 7, 4,
    1, -1, 7,
    9, 7, 8,
    9, -1, 7,
    3, 9, 6,
    3, -1, 9,
    7, 9, 8,
    7, -1, 9
  ),byrow = T, ncol = 3
```

```
)

match_rows_third_move <- function(x, y){
  stopifnot(ncol(x) == ncol(y))
  stopifnot(nrow(y) == 1)
  matched1 <- which(x[,1] == y[1,1])
  matched2 <- which(x[matched1,2] == y[1,2])
  if(is.integer(matched2) && length(matched2) == 0L){
    row_index <- matched1[which(x[matched1,2] == -1)]
  }else{
    row_index <- matched1[matched2]
  }
  #row_index <- matched2[which(!is.na(match(matched2,matched1)))]
  x[row_index,3]
}

p1_third_move <- function(players){
  p2_move1 <- players[1,2]
  p2_move2 <- players[2,2]
  p1_move <-
match_rows_third_move(p1_third_move_matrix,matrix(c(p2_move1,p2_move2,NA),nro
w = 1))
  p1_move
}

p1_forth_move_matrix <- matrix(
  c(
    2, 1, 7, 6,
    2, 1, -1, 7,
    4, 7, 9, 2,
    4, 7, -1, 9,
    6, 3, 1, 8,
    6, 3, -1, 1,
    8, 9, 3, 4,
    8, 9, -1, 3,
    1, 7, 6, 2,
    1, 7, -1, 6,
    9, 7, 2, 4,
    9, 7, -1, 2,
    3, 9, 4, 2,
    3, 9, -1, 4,
    7, 9, 2, 4,
    7, 9, -1, 2
  ),byrow = T, ncol = 4
)

match_rows_forth_move <- function(x, y){
  stopifnot(ncol(x) == ncol(y))
  stopifnot(nrow(y) == 1)
```

```
  matched1 <- which(x[,1] == y[1,1])
  matched2 <- which(x[matched1,3] == y[1,3])
  if(is.integer(matched2) && length(matched2) == 0L){
    row_index <- matched1[which(x[matched1,3] == -1)]
  }else{
    row_index <- matched1[matched2]
  }
  #row_index <- matched2[which(!is.na(match(matched2,matched1)))]
  x[row_index,4]
}

p1_fourth_move <- function(players){
  p2_move1 <- players[1,2]
  p2_move2 <- players[2,2]
  p2_move3 <- players[3,2]
  p1_move <-
match_rows_forth_move(p1_forth_move_matrix,matrix(c(p2_move1,p2_move2,p2_move
3,NA),nrow = 1))
  p1_move
}

strategy_move_p1 <- function(players,ind){
  if(ind==1){
    p1_move <- 5
  }else if(ind == 3){
    p1_move <- p1_second_move(players)
  }else if(ind==5){
    p1_move <- p1_third_move(players)
  }else if(ind==7){
    p1_move <- p1_fourth_move(players)
  }
  p1_move
}

#strategy simulation 100, 1000, and 10000
win_rate <- rerun(100,strategy_player_1())
win_matrix_100 <- data.frame(matrix(unlist(win_rate), nrow=100, byrow=T))
(t <- table(win_matrix_100))
```

```
##    X2
## X1   5  7  9
##   0  0  0  1
##   1 81 17  1
```

```
rate100 <- rowSums(t)[3]/sum(rowSums(t))
win_rate <- rerun(1000,strategy_player_1())
win_matrix_1000 <- data.frame(matrix(unlist(win_rate), nrow=1000, byrow=T))
(t <- table(win_matrix_1000))
```

```
##    X2
## X1    5   7   9
##   0   0   0  15
##   1 835 138  12
```

```
rate1000 <- rowSums(t)[3]/sum(rowSums(t))
win_rate <- rerun(10000,strategy_player_1())
win_matrix_10000 <- data.frame(matrix(unlist(win_rate), nrow=10000, byrow=T))
(t <- table(win_matrix_10000))
```

```
##    X2
## X1     5    7    9
##   0    0    0   87
##   1 8405 1407  101
```

```
rate10000 <- rowSums(t)[3]/sum(rowSums(t))
```

```
#####################
win_rate <- rerun(1000,strategy_player_1())
win_matrix2 <- data.frame(matrix(unlist(win_rate), nrow=1000, byrow=T))
names(win_matrix2) <- c('result','index')
table(win_matrix2[which(win_matrix2$result==1),]$index)
```

```
##
##   5   7   9
## 839 139  10
```

```
table(win_matrix2$result)
```

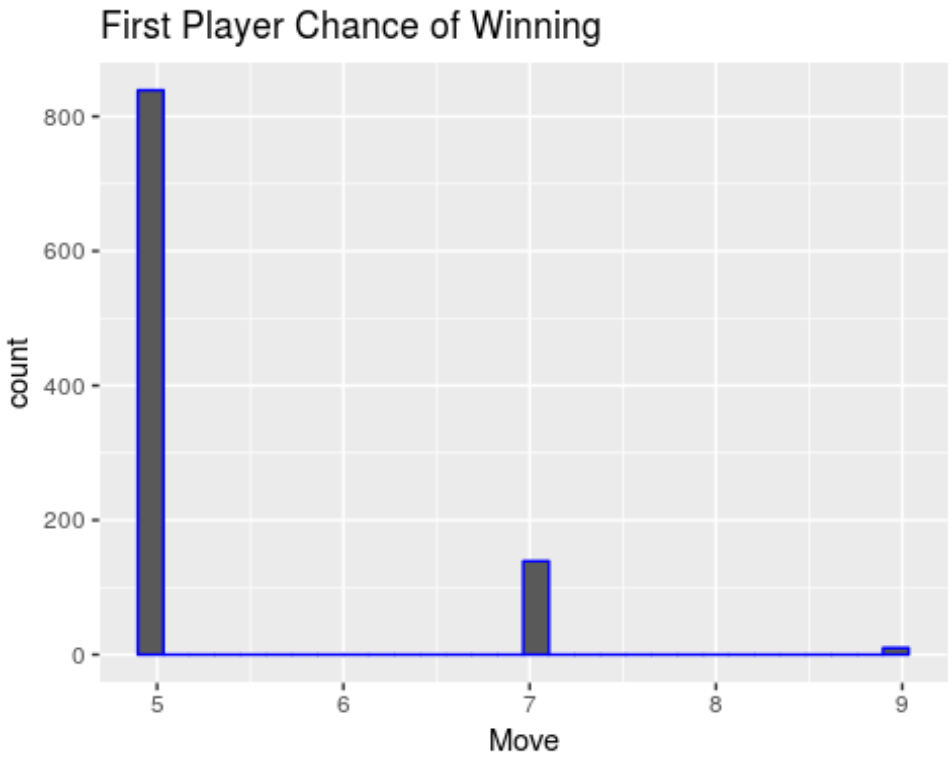
```
##
##   0   1
##  12 988
```

```
#plotting the number of winning for first player when we simulate 1000

df_win3 <- data.frame(move =
win_matrix2[which(win_matrix2$result==1),]$index)

ggplot(df_win3) +
geom_histogram(mapping = aes(x = move), col ="blue")+
  xlab("Move")+
  ggtitle("First Player Chance of Winning")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```


First Player Chance of Winning
800
600
400
200
0
count
5
6
7
8
9
Move