Completing the random playing investigations

```
library(combinat)
```

```
##
## Attaching package: 'combinat'

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

```
flag <- array(9,0)  #creating flag to start the game

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),
                  byrow = F,nrow = 3)   #all possible winning spots

flag[sample(1:9,1)]=1


rand_strategy <- function(){
  flag <- rep(0,9)
  code <- 0
  players <- matrix(data=0,nrow = 5,ncol = 2)
  for(i in 1:9){
    play <- sample(1:9,1)
    while(flag[play]==1){
      play <- sample(1:9,1)
    }
    players[((i-1)/2+1),ifelse((i%%2)==1,1,2)] = play; #even number for first player and odd for the se

    flag[play]=1;
    if(i>4){        #since the winning won't happen in the first four steps
      result <- check_win(players,i)  #to check the winner
      if(result$code != 0){
        return(result)
      }
    }
  }
  return(result)
}

check_win <-function(players,ind){
  code <- 0
  ifelse(ind%%2==1,index <- 1,index <- 2)
  player_i <- sort(players[,index])
  player_i <- player_i[! player_i %in% c(0)]    ##extracting the 0 values to start the combination win
  player_i_matrix <- combn(player_i,3)    #using combn function from combinat package to chech all the
  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])   #this step to match the combination with
      if(any(is.na(match_vector)) == FALSE){     #NA means the two vector do not match. no NA means the
        ifelse(ind%%2==1,code <- 1,code <- -1)
        return(data.frame(code=code,index=ind))  #index is the step in the game.
      }
    }
  }
  return(data.frame(code=code,index=ind))
}
```