R Notebook

Firstly, let N be 100 and n be 37

```
prob_given_index <- function(i, k, N, n){
  prob <- (n / (i - 1))
  prob <- prob * ((factorial(N - i) * factorial(N - k))/(factorial(N - 1) * factorial(N - i - k + 1)))
  return(prob / N)
}
```

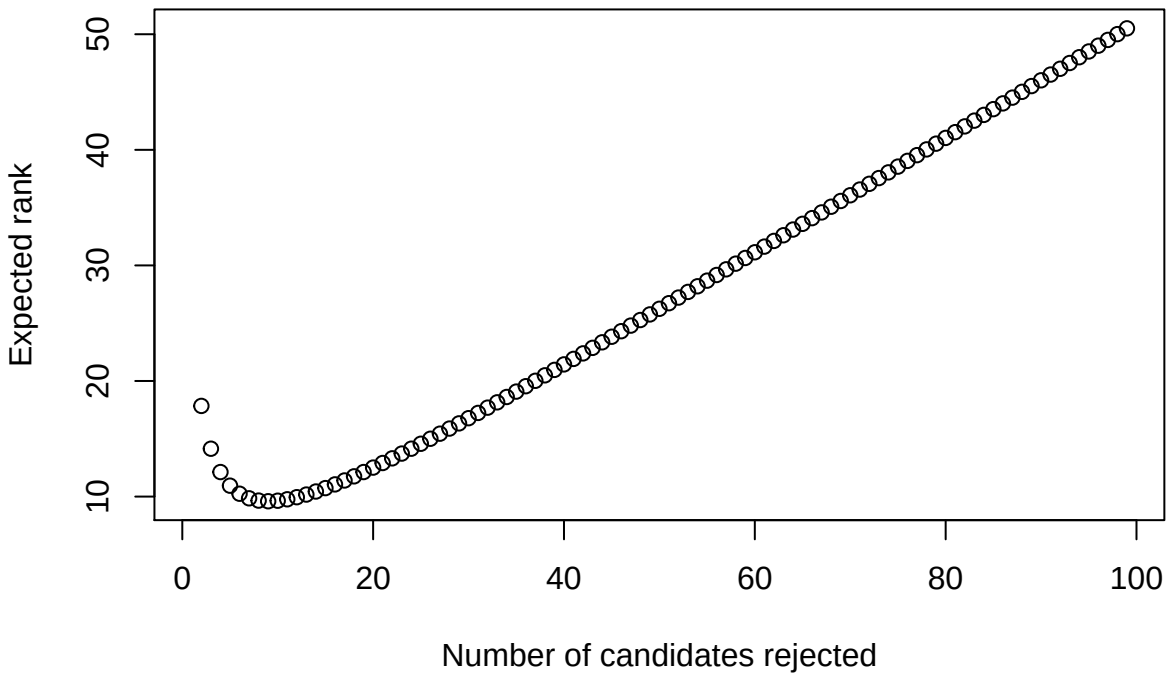
Secondly, let's define function to find $P\{R_{N,n} = k\}$

```
last_candidate <- function(N, n){
  return((n) / (N * (N - 1)))
}
general_way <- function(k, N, n){
  if(k > N - n){
    return(0)
  }else{
    probability <- 0
    for(i in seq(n + 1, N - k + 1, 1)){
      probability <- probability + prob_given_index(i, k, N, n)
    }
    return(probability)
  }
}

whole_prob <- function(k, N, n){
  return(general_way(k, N, n) + last_candidate(N, n))
}
```

Function of calculating and plotting expected ranks using formula derived:

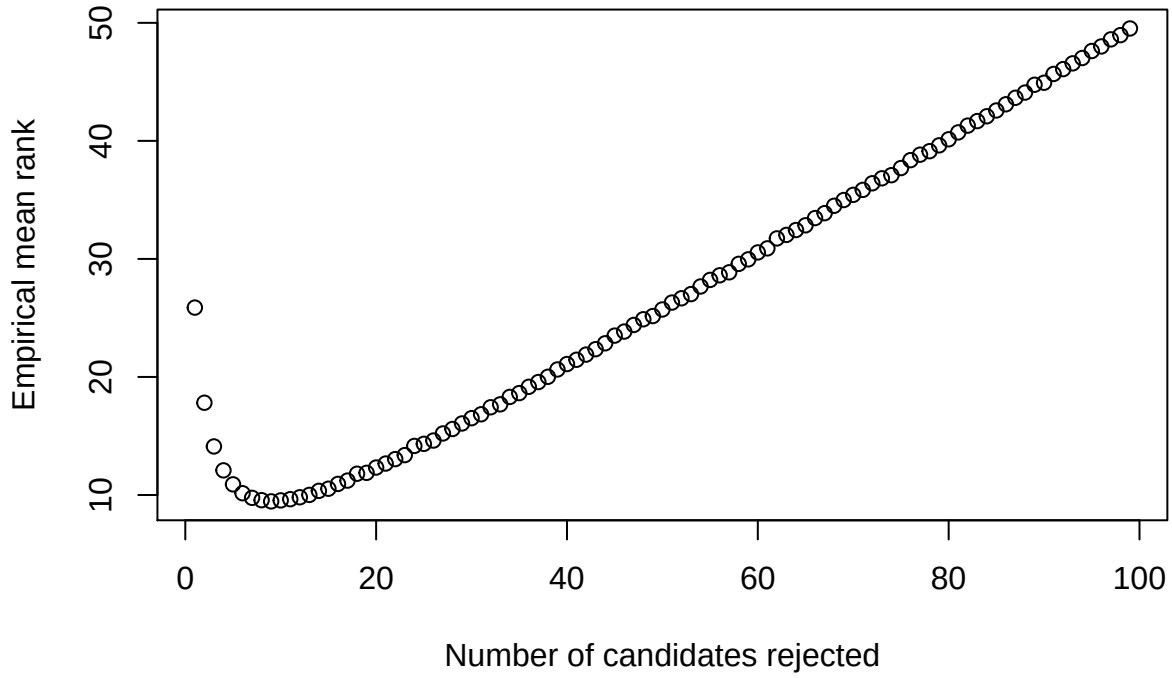
```
expected_rank <- function(N, n){
  rank <- 0
  for(k in seq(1, N)){
    rank <- rank + k * whole_prob(k, N, n)
  }
  return(rank)
}
e <- exp(1)
plot_expected_ranks <- function(N){
  y <- c()
  for(n in 1:N-1){
    y[n] = expected_rank(N, n)
  }
  plot(y, xlab = "Number of candidates rejected", ylab = "Expected rank")
}
plot_expected_ranks(100)
```




```
rank <- function(experiment, N, n){
    max_skipped <- max(experiment[1:n])
    for(i in seq(n+1, N)){
      if(experiment[i] > max_skipped){
        return(N - experiment[i] + 1)
      }
    }
    return(experiment[N])
  }

empirical_mean_rank<-function(N, n){
  num_experiments <- 100000
  successes <- 0
  candidates <- seq(1, N, 1)
  experiments <- replicate(num_experiments, sample(candidates))
  for(ind_exp in 1:num_experiments){
    experiment <- experiments[, ind_exp]
    successes <- successes + rank(experiment, N, n)
  }
  return((successes/num_experiments))
}
plot_empirical_mean_ranks <- function(N){
  y <- c()
  for(n in 1:N-1){
    y[n] = empirical_mean_rank(N, n)
  }
```

```
  plot(y, xlab = "Number of candidates rejected", ylab = "Empirical mean rank")
}
plot_empirical_mean_ranks(100)
```



In conclusion, we can say, that my hypothesis was proven by experiments even thoough I could not derive a simpler formula for $E(R_{n,N})$. And that "golden rule" was not so useful for the task of minimizing rank of chosen candidate, because this algorithm was made for choosing **the best** candidate, and every other candidate was considered a failure.