Exercício de Programacão 1: Balcão de Atendimento com Fila

Caio Camelo (NUSP 12623911) & Rafael Mendes (NUSP 12543141)

2024-05-12

Exercício de Programacão 1: Balcão de Atendimento com Fila

Introdução

Função auxiliar para cálculo do erro padrão

```
erro_padrao <- function(dados) {
  dp <- sd(dados)
  st_err <- dp / sqrt(length(dados))
  return(st_err)
}
```

Função para encontrar probabilidade de tm maior que determinado valor

```
calc_probab_tm <- function(tmax_values, value) {
  tm <- tmax_values
  tamanho <- length(tm)
  cont <- 0
  for (i in 1:tamanho) {
    if (tm[i] > value) {
      cont <- cont + 1
    }
  }
  return(cont / tamanho)
}
```

Função que faz a simulação de uma sequêcia de atendimento

```
atendimento <- function(tempo_total, rate, n, mi) {
  # Inicialização das variáveis de interesse e dos vetores (tdisp e tcheg)
  tempo_acum <- 0
  atendidos <- 0
  naoatendidos <- 0
  clientes_novos <- 0
  tempo_max <- 0
  tdisp <- numeric(n)
```

```
  tcheg <- numeric(0)

  # Gera intervalos de tempo entre atendimentos até seu encerramento
  while (TRUE) {
    z <- rexp(1, rate)
    tempo_acum <- tempo_acum + z
    if (tempo_acum > tempo_total)
      break
    clientes_novos <- clientes_novos + 1
    tcheg <- c(tcheg, tcheg[clientes_novos])
    tcheg[clientes_novos] <- tempo_acum

    while ((min(tdisp) <= tempo_acum) && (atendidos < clientes_novos)) {
      atendidos <- atendidos + 1
      j <- which.min(tdisp)
      a <- rexp(1, mi)
      tdisp[j] <- max(tdisp[j], tcheg[atendidos]) + a
      tempo_max <- max(tempo_max, (tdisp[j] - tcheg[atendidos]))
    }
    tam_fila <- max(0, (clientes_novos - 1) - atendidos)
    prob <- tam_fila / (tam_fila + n)
    s <- rbinom(1, 1, prob)

    if (s == 1) {
      clientes_novos <- clientes_novos - 1
      naoatendidos <- naoatendidos + 1
    }
    tam_fila <- clientes_novos - atendidos
    prop_naoatendidos <- naoatendidos / (atendidos + naoatendidos + tam_fila)

  }
  return(list(x = atendidos, y = naoatendidos, r = tam_fila,
              w = prop_naoatendidos, tmax = tempo_max))
}
```

Função que simula N atendentimentos por blocos de nb=500 e aplica o teorema do limite central

```
simula_atendimentos <- function(t, rate, n, mi, nb, c) {
  # Inicialização de variáveis e vetores de interesse
  inicio <- 0
  fim <- nb
  tam <- integer(0)
  ws <- 0
  x_elem <- numeric(nb)
  y_elem <- numeric(nb)
  w_elem <- numeric(nb)
  tmax_elem <- numeric(nb)
  vetor_ws <- numeric(0)
  vetor_w_err <- numeric(0)
  vetor_tmax_err <- numeric(0)
  vetor_medias_x <- numeric(0)
```

```
  vetor_medias_y <- numeric(0)
  vetor_medias_w <- numeric(0)
  vetor_medias_tmax <- numeric(0)

  while (TRUE) {
    # obtem os dados cumulados a cada nb simulações de atendimento
    for (i in (inicio + 1):fim) {
      resultado <- atendimento(t, rate, n, mi)
      x_elem[i] <- resultado$x
      y_elem[i] <- resultado$y
      w_elem[i] <- resultado$w
      tmax_elem[i] <- resultado$tmax
    }
    tam <- c(tam, fim) # atualiza o vetor de tamanho parcial

    # vetores resultantes com as médias efetuadas a cada nb iterações
    vetor_medias_x <- c(vetor_medias_x, mean(x_elem))
    vetor_medias_y <- c(vetor_medias_y, mean(y_elem))
    vetor_medias_w <- c(vetor_medias_w, mean(w_elem))
    vetor_medias_tmax <- c(vetor_medias_tmax, mean(tmax_elem))

    # erros padrão de tmax e de w a cada nb iterações
    tmax_err <- erro_padrao(tmax_elem)
    vetor_tmax_err <- c(vetor_tmax_err, tmax_err)
    w_err <- erro_padrao(w_elem)
    vetor_w_err <- c(vetor_w_err, w_err)

    # A cada amostra de tamanho i x nb ( sendo i o número da iterações
    # geraradoras de novos dados ), guadaremos o valor ws que corresponde ao
    # limite superior do intervalo de confiança calculado para W.
    ws <- mean(w_elem) + (1.96 * w_err)
    vetor_ws <- c(vetor_ws, ws)

    # faz a convergência em relação ao erro padrão de w
    threshold <- 2 * 1.96 * w_err
    if (threshold < c)
      break

    # aloca mais posições aos vetores de elementos
    x_elem <- c(x_elem, numeric(nb))
    y_elem <- c(y_elem, numeric(nb))
    w_elem <- c(w_elem, numeric(nb))
    tmax_elem <- c(tmax_elem, numeric(nb))
    inicio <- inicio + nb
    fim <- fim + nb
  }
  return(list(x_k = vetor_medias_x, y_k = vetor_medias_y, w_k = vetor_medias_w,
              tmax_k = vetor_medias_tmax, k = tam, tmax_ep = vetor_tmax_err,
              w_ep = vetor_w_err, w = w_elem, tm = tmax_elem, ws_k = vetor_ws))
}
```

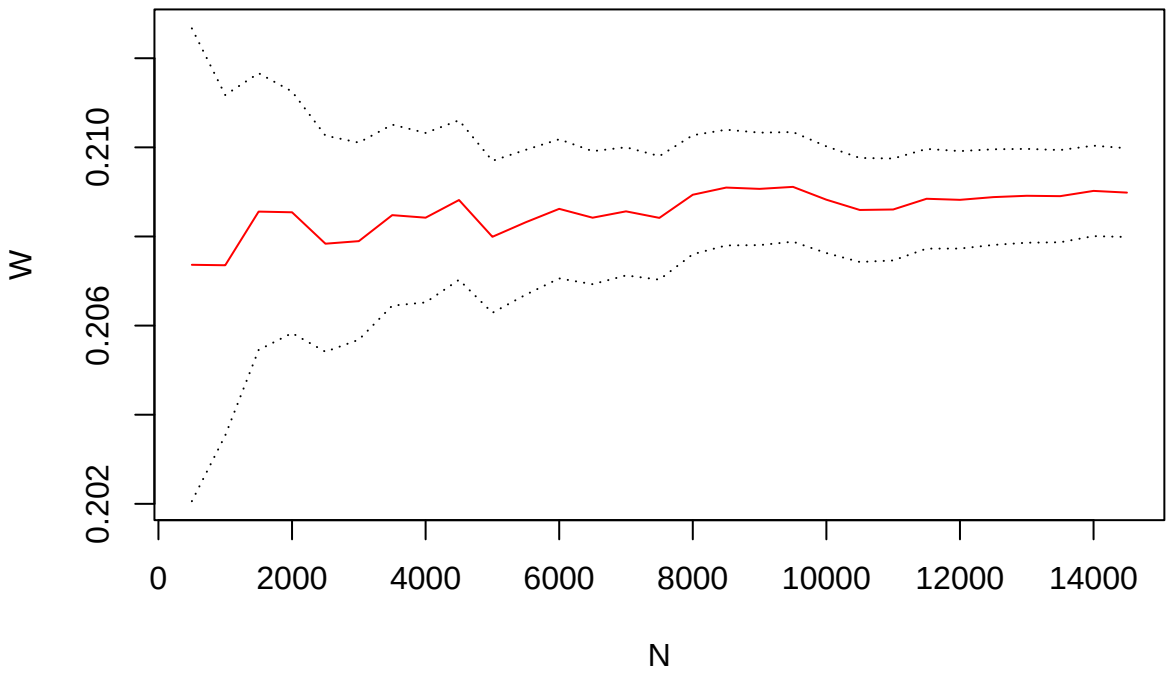
Subproblema 1

Grava o resultado do experimento na variável simulacao

```
simulacao <- simula_atendimentos(t = 50, rate = 3, n = 5, mi = 0.5,
                                 nb = 500, c = 0.002)
```

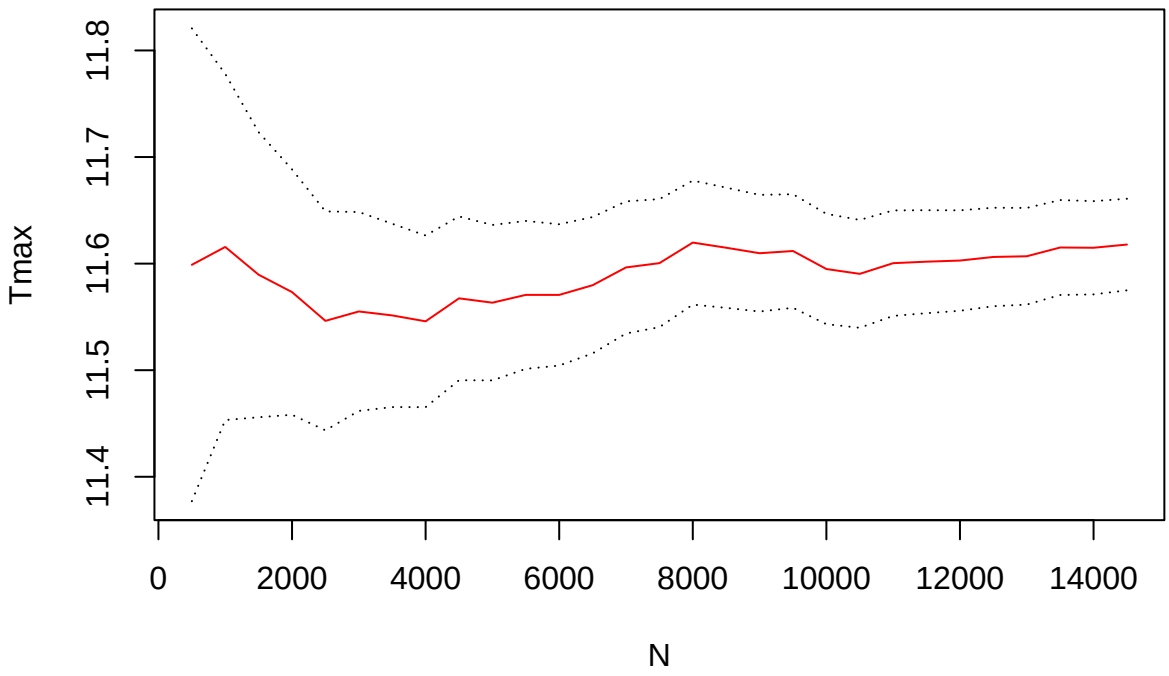
Item 1 - Gráficos de Linha do W

```
# Gera dataframe com a variável de interesse referente a W
df1 <- cbind.data.frame(simulacao$k, simulacao$w_k, simulacao$w_ep)
names(df1)[names(df1) == "simulacao$w_k"] <- "W"
names(df1)[names(df1) == "simulacao$k"] <- "N"
names(df1)[names(df1) == "simulacao$w_ep"] <- "erro_padrao"

# Adiciona ao df1 as colunas que definem o intervalo de confiança de 95%
df1$lim_inf <- df1$W - 1.96 * df1$erro_padrao
df1$lim_sup <- df1$W + 1.96 * df1$erro_padrao

# Gera o gráfico de linha com as médias parciais de W e o intervalo de confiança
plot(df1$N, df1$W, type = "l", col = "red", xlab = "N", ylab = "W",
     ylim = range(c(df1$W, df1$lim_inf, df1$lim_sup)))
lines(df1$N, df1$lim_inf, lty = "dotted")
lines(df1$N, df1$lim_sup, lty = "dotted")
```



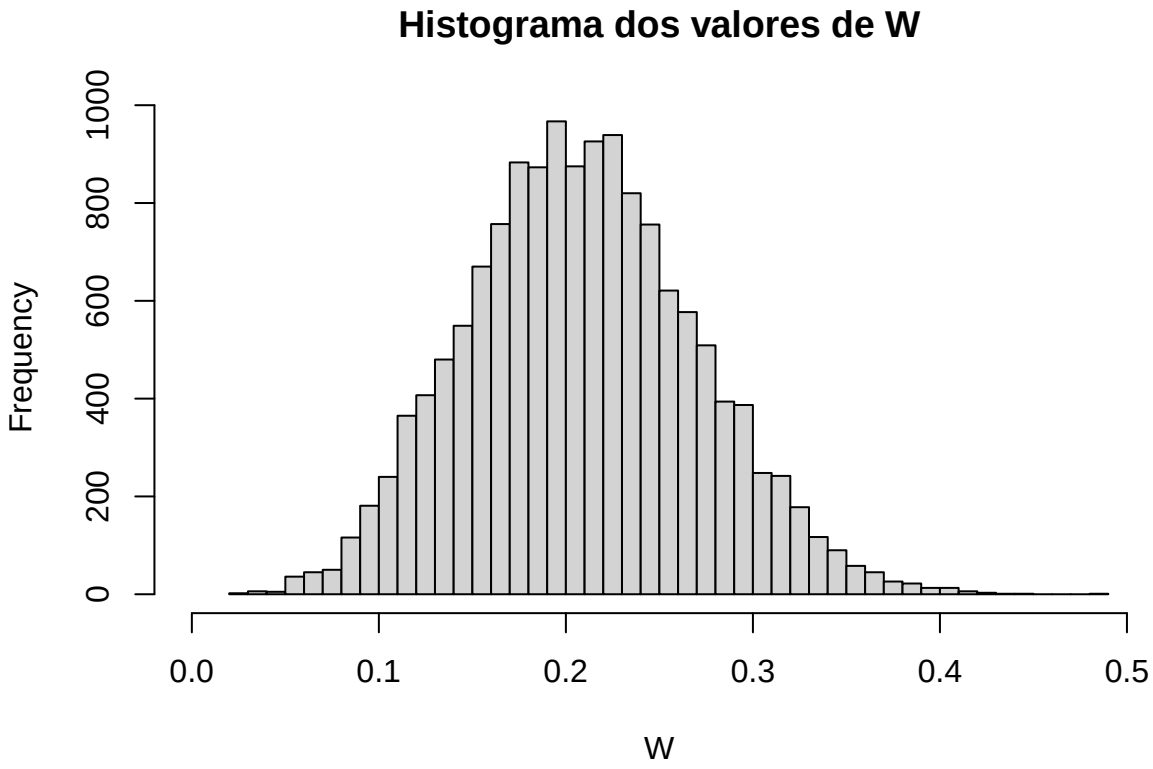
Item 2 - Gráficos de Linha do TM

```r
# Gera dataframe com a variável de interesse referente a média de tempo máximo
df2 <- cbind.data.frame(simulacao$k, simulacao$tmax_k, simulacao$tmax_ep)
names(df2)[names(df2) == "simulacao$tmax_k"] <- "Tmax"
names(df2)[names(df2) == "simulacao$k"] <- "N"
names(df2)[names(df2) == "simulacao$tmax_ep"] <- "erro_padrao"

# Adiciona ao df2 as colunas que definem o intervalo de confiança de 95%
df2$lim_inf <- df2$Tmax - 1.96 * df2$erro_padrao
df2$lim_sup <- df2$Tmax + 1.96 * df2$erro_padrao

# Gera o gráfico de linha com médias parciais de Tmax e o intervalo de confiança
plot(df2$N, df2$Tmax, type = "l", col = "red", xlab = "N", ylab = "Tmax",
     ylim = range(c(df2$Tmax, df2$lim_inf, df2$lim_sup)))
lines(df2$N, df2$lim_inf, lty = "dotted")
lines(df2$N, df2$lim_sup, lty = "dotted")
```

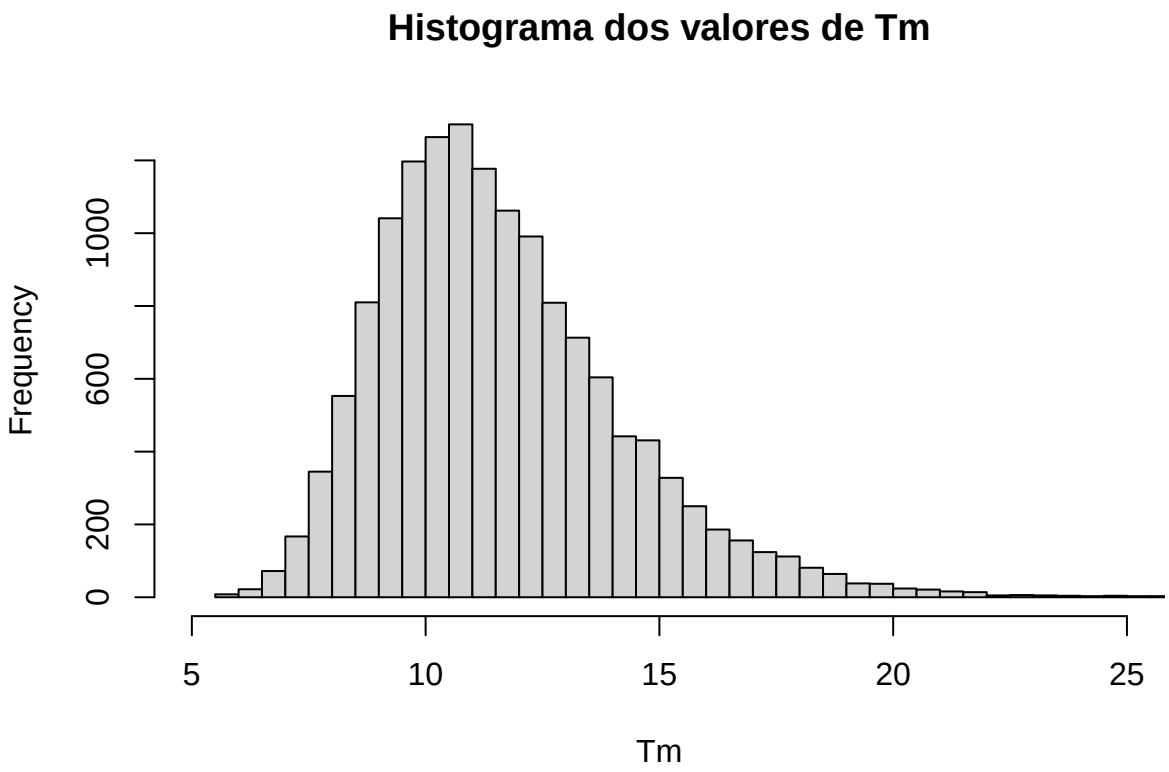


Item 3 - Histogramas

```
# Histograma de W nas N iterações
hist(simulacao$w, breaks = 40, main = "Histograma dos valores de W",
     xlab = "W", xlim = c(0, 0.5))
```




```r
# Histograma de Tm nas N iterações
hist(simulacao$tm, breaks = 40, main = "Histograma dos valores de Tm",
     xlab = "Tm", xlim = c(5, 25))
```



Item 4 - Médias

```
# Médias finais de X, Y, W e TM
print("Media Final de X")
```

```
## [1] "Media Final de X"
```

```
print(simulacao$x_k)
```

```
##  [1] 115.9120 115.6570 115.9300 115.9705 115.9936 115.9387 115.9086 115.8905
##  [9] 115.8420 115.8658 115.8165 115.7700 115.8318 115.8247 115.7701 115.7470
## [17] 115.7340 115.7603 115.7523 115.7641 115.7617 115.7215 115.7176 115.7171
## [25] 115.7004 115.6978 115.6869 115.6854 115.7077
```

```
print("Media Final de Y")
```

```
## [1] "Media Final de Y"
```

```
print(simulacao$y_k)
```

```
##  [1] 31.53000 31.51100 31.79267 31.82300 31.69720 31.69767 31.79229 31.77775
```

```
##  [9] 31.84289 31.70560 31.75273 31.79917 31.76646 31.78900 31.73853 31.82512
## [17] 31.85612 31.85689 31.86547 31.81220 31.76933 31.76400 31.80704 31.79783
## [25] 31.81240 31.81954 31.81615 31.83714 31.83110
```

```
print("Media Final de W")
```

```
## [1] "Media Final de W"
```

```
print(simulacao$w_k)
```

```
##  [1] 0.2073644 0.2073540 0.2085584 0.2085426 0.2078380 0.2078950 0.2084787
##  [8] 0.2084216 0.2088179 0.2079924 0.2083195 0.2086199 0.2084215 0.2085635
## [15] 0.2084170 0.2089367 0.2090977 0.2090683 0.2091128 0.2088251 0.2085949
## [22] 0.2086056 0.2088461 0.2088223 0.2088835 0.2089129 0.2089047 0.2090227
## [29] 0.2089843
```

```
print("Media Final de TM")
```

```
## [1] "Media Final de TM"
```

```
print(simulacao$tmax_k)
```

```
##  [1] 11.59889 11.61564 11.58958 11.57327 11.54627 11.55510 11.55139 11.54584
##  [9] 11.56741 11.56334 11.57065 11.57063 11.57972 11.59642 11.60044 11.61972
## [17] 11.61489 11.60975 11.61182 11.59494 11.59039 11.60042 11.60181 11.60287
## [25] 11.60624 11.60689 11.61510 11.61487 11.61789
```

Item 5 - Probabilidade Pr(tm > 13)

```
# Resultado da probabilidade de tm > 13
probab_tm <- calc_probab_tm(simulacao$tm, value = 13)
print(paste("P(tm > 13) =", probab_tm))
```

```
## [1] "P(tm > 13) = 0.254"
```

Item 6 - Determinar o ws para cada amostra, tal que, Pr(W <= ws) >= 95%

```
# Exibição dos valores de ws para cada amostra, tal que, P(W <= ws) >= 95%
print("ws para cada iteração")
```

```
## [1] "ws para cada iteração"
```

```
print(paste(simulacao$k, simulacao$ws_k))
```

```
##  [1] "500 0.212670503379577"   "1000 0.211173811239659"
##  [3] "1500 0.211663896538578"  "2000 0.211253380094527"
##  [5] "2500 0.210263126754006"  "3000 0.21010641321025"
##  [7] "3500 0.210509606444497"  "4000 0.210323364115506"
##  [9] "4500 0.210607484000957"  "5000 0.20969786415512"
## [11] "5500 0.209944849091717"  "6000 0.210181426544112"
## [13] "6500 0.209914095512712"  "7000 0.210001671514091"
## [15] "7500 0.209803036086084"  "8000 0.210272101211059"
## [17] "8500 0.210396340255098"  "9000 0.210330423899847"
## [19] "9500 0.210342825876972"  "10000 0.210021991461534"
## [21] "10500 0.209762123214238" "11000 0.209749437767958"
## [23] "11500 0.209964630314804" "12000 0.209915871379257"
## [25] "12500 0.20995684154551"  "13000 0.209965845382964"
## [27] "13500 0.209939312682312" "14000 0.210037751781359"
## [29] "14500 0.209980194846677"
```

Subproblema 2

Determinar o número de guichês para que Pr(w <= 20%) >= 95%

Função que simula uma amostra de atendimentos

```
# São passsados parâmetros como o número de guichês, o limite superior do
# intervalo de confiança, o tamanho da amostra e o valor crítico, que
# representa o nível de congfiança esperado (95%).

simula_atendimentos_II <- function(t, rate, n, mi, tam_amostra, ls, zc) {
  # Inicialização de vetores de interesse
  x_elem <- numeric(0)
  y_elem <- numeric(0)
  w_elem <- numeric(0)
  vetor_w_err <- numeric(0)


  # obtem os dados de simulações de atendimento (tam_amostra)
  for (i in 1:tam_amostra) {
    resultado <- atendimento(t, rate, n, mi)
    x_elem <- c(x_elem, resultado$x)
    y_elem <- c(y_elem, resultado$y)
    w_elem <- c(w_elem, resultado$w)
  }

  # erro padrão de w a cada 500 iterações
  w_err <- erro_padrao(w_elem)
  vetor_w_err <- c(vetor_w_err, w_err)

  # Como desejamos verificar se o número de guichês(n) garante que o parâmetro w
  # esteja abaixo do limite superior ls, com um certo nível de confiança
  # (valor crítico zc), fazemos as seguintes comparações para a simulação atual

  threshold <- (ls - mean(w_elem)) / w_err
```

```
  if (threshold >= zc){
    saida<-paste("Tam_Amostra:",length(w_elem),"n:", n, "média W:",mean(w_elem))
    print(saida)
    return(1)
  }else{
    saida<-paste("Tam_Amostra:",length(w_elem),"n:", n, "média W:",mean(w_elem))
    print(saida)
    return(0)}

}
```

Função que simula várias amostras de atendimentos para determinar o número ideal de guichês

Parâmetros aplicados à simulação: (t=60, rate=4, n=n_guiches, mi=0.5, tam_amostra=500, ls=0.2, zc=1.96)

```
subproblema_II <- function(){

  n_guiches <- 1

  # Realizamos a simulçao até encontrarmos um valor de n_guiches que
  # garanta as condições de entrada
  while(TRUE){
    simulacao <- simula_atendimentos_II(t = 60, rate = 4, n = n_guiches, mi = 0.5,
                                        tam_amostra = 500, ls = 0.2, zc= 1.96)

    if(simulacao == 1)
      break
    n_guiches = n_guiches + 1
  }

  print(paste("O número de guichês necessário deve ser maior ou igual a", n_guiches))
  return(n_guiches)
}
```

Função que exibe o resultado para o subproblema 2

```
Teste <- subproblema_II()
```

```
## [1] "Tam_Amostra: 500 n: 1 média W: 0.839422480562427"
## [1] "Tam_Amostra: 500 n: 2 média W: 0.713776632277274"
## [1] "Tam_Amostra: 500 n: 3 média W: 0.589367776982486"
## [1] "Tam_Amostra: 500 n: 4 média W: 0.471611271441149"
## [1] "Tam_Amostra: 500 n: 5 média W: 0.362129343441377"
## [1] "Tam_Amostra: 500 n: 6 média W: 0.260460597604868"
## [1] "Tam_Amostra: 500 n: 7 média W: 0.17521605517017"
## [1] "O número de guichês necessário deve ser maior ou igual a 7"
```