Corrida de Rua

```
corrida=function(idade, sexo, distancia=5, duracao=4, dir="") #declara a
função e seus parâmetros
{
    ################################
    ### validacao dos parâmetros ###
    #idade
    if ((round(idade)!=idade) || (idade<18) || (idade>60))    #verifica se
arredondar idade é igual a idade, para garantir que o valor seja inteiro,
além de garantir que está entre 18 e 60
    {
        stop("O parâmetro 'idade' deve ser um valor inteiro entre 18 e 60.")
#exibe mensagem de erro e sai da função
    }
    #sexo
    x=c("feminino", "masculino")   #lista com as possibilidades
    sexo=match.arg(sexo, x) #vê se o que foi passado como parâmetro se
encaixa em alguma delas
    if ((sexo!="feminino") && (sexo!="masculino"))    #verifica o match do
que foi preenchido no parâmetro
    {
        stop("O parâmetro 'sexo' é obrigatório e deve ser preenchido com
'feminino, fem, f' ou 'masculino, masc, m'.")  #exibe mensagem de erro e sai
da função
    }
    #distância
    dist=c(5,10,16) #cria vetor com os valores para distância aceitos pela
função
    if (is.element(distancia, dist) == FALSE)   #verifica se a distância
informada é 5, 10 ou 16
    {
        stop("Escolher entre os seguintes valores para distância de treino:
5, 10 ou 16.")    #exibe mensagem de erro e sai da função
    }
    #duração do treino em semanas
    dur=c(4,8,12) #cria vetor com os valores para duração do treino aceitos
pela função
    if (is.element(duracao, dur) == FALSE)   #verifica se a duracao do
treino informada é 4, 8, ou 12 semanas
    {
        stop("Escolher entre os seguintes valores para a duração do seu
treino: 4, 8 ou 12 semanas.")    #exibe mensagem de erro e sai da função
    }
    #diretório para gravação do arquivo de output
    if (dir=="") #verifica se o parâmetro foi preenchido
    {
        stop("O parâmetro 'diretório' é obrigatório e deve ser preenchido
com o local de gravação do arquivo pdf.") #exibe mensagem de erro e sai da
função
```

```
    }
    ################################


    ################################
    ###         abre PDF         ###
    pdf(file=paste(dir,'/Planilha_de_treino.pdf',sep=""), #caminho do
arquivo PDF que será gerado
        onefile=T, #armazena todos os gráficos num único arquivo
        paper='A4r', #o arquivo tem tamanho A4 no formato "paisagem"
        width=10, #largura: 10 inches
        height=7 #altura: 7 inches
        )
    ################################


    ################################
    ###      cálculo da FC      ###
    #FCM: frequência cardíaca máxima
    if (sexo =="feminino") #verifica o sexo. se "feminino"
    {
        FCM=226-idade #cálculo estimado, baseado nas literaturas esportivas
e nas referências citadas na contextualização da função
    }
    else #se o sexo não é "feminino", é "masculino"
    {
        FCM=220-idade #cálculo estimado, baseado nas literaturas esportivas
e nas referências citadas na contextualização da função
    }
    ################################


    ######################################################
    ### Faixa de utilização da FCM por tipo de treino ###
    treino_leve=round(0.65*FCM) #Treino leve=70% da FCM, arredondado
    treino_longo=round(0.80*FCM) #Treino longo=80% da FCM, arredondado
    treino_tiro=round(0.90*FCM) #Treino tiro=90% da FCM, arredondado
    ######################################################


    #define alguns parâmetros GERAIS padrões para os gráficos
    par(    family="sans", #família da fonte será "sans"
            cex=1.1 #tamanho será 10% a mais que o tamanho padrão
        )


##########################################################################
#############
    ###                                G R Á F I C O S
###
```

```
#######################################################################
#############
    #monta gráfico de barra com FCM e tipo de treino
    graph=barplot(  c(treino_leve, treino_longo, treino_tiro), #valores do
eixo y
                    col=adjustcolor(c("green3", "royalblue1", "firebrick1"),
alpha.f = 0.3), #define cores específicas para cada barra e a transparência
                    ylim=c(0,FCM), #limite do eixo y
                    names.arg=c("Treino Leve", "Treino Longo", "Treino de
Tiro"), #nomes de cada barra (eixo x)
                    border="black", #cor da borda de cada barra
                    axes=FALSE, #não plota nem traço nem valores dos eixos
                    main="Frequência Cardíaca (FC) ideal\nem cada tipo de
treino" #nome do gráfico
                )
    text(   graph, #no grafico montado acima, plota:
            c(treino_leve, treino_longo, treino_tiro), #na posição y
            labels=c(treino_leve, treino_longo, treino_tiro), #o conteúdo
(valor) de cada barra
            pos=3, #define que será acima das barras
            font=2, #fonte em negrito
            cex=0.9 #um pouco menor (90%) do tamanho definido em
par(cex=1.1)
        )

    #monta gráfico de linha exemplificando como deve ser um ciclo de treino
de tiro (frequência cardíaca X tempo)
    ########################################
    ### definição dos parâmetros do gráfico
    tempo=c(0,5,seq(from=6, to=35), 37,40) #5 minutos de aquecimento e
descanso final e 10 ciclos de tiro (2 minutos fortes + 1 minuto descanso)
    zona_tiro=c(round(0.85*FCM), treino_tiro) #define a zona de tiro para a
FCM, que deve variar entre 85 a 95% da FCM
    fr_mod=round(0.5*FCM) #frequência cardíaca da zona de exercício moderado
    fr_leve=round(0.35*FCM) #frequência cardíaca da zona de exercício leve
    fr=c(fr_leve, fr_mod, rep(c(zona_tiro, fr_mod), 10), 0.8*fr_mod,
fr_leve) #monta vetor com as zonas de frequencia cardíaca para construção do
gráfico (0.8*fr_mod é apenas para demonstrar a FC diminuindo)
    ########################################

    ### começa a desenhar o gráfico
    par(    lwd=2, #largura da linha
            cex.axis=0.5, #diminuir proporcionalmente o tamanho do label dos
eixos
            tcl=-0.5, #tamanho do tick (menor que o default)
            mgp=c(3,0.5,0) #distância da margem para o label dos eixos
        )
    plot(   fr~tempo, #plota gráfico da frequência cardíaca em função do
tempo
            ylim=c(40, FCM), #delimita o tamanho do eixo y
            xlab="Tempo do treino (em minutos)", #label do eixo x
```

```
            ylab="", #sem legenda do eixo y
            main="Detalhe da variação da FC\ndurante o treino de tiro",
#explicação do gráfico
            pch=20, #tipo de caracter que será plotado (bolinha pequena)
            bty="n", #sem borda no gráfico
            xaxt="n", #remove eixo x
            yaxt="n" #remove eixo y
        )

    ### montagem do eixo x
    i=6 #começo do contador do loop para montar os ticks do eixo x
    while(i <= 35) #loop para preencher os labels do eixo x dinamicamente
    {   #loop vai diferenciar os dois minutos do tiro e um minuto no
descanso nos ticks do eixo x
        axis(1, at=i, tck=0.02, mgp=c(0,-1.5,0), tcl=0.5) #primeiro minuto
no treino de tiro, tick e label internos
        i=i+1 #acrescenta 1 no contador
        axis(1, at=i, tck=0.02, mgp=c(0,-1.5,0), tcl=0.5) #segundo minuto no
treino de tiro, tick e label internos
        i=i+1 #acrescenta 1 no contador
        axis(1, at=i, tck=NA) #último minuto, descanso - tick e label de
volta pra fora do plot
        i=i+1 #acrescenta 1 no contador
    }
    axis(1, at=seq(5,35, by=30), labels=F) #traça só a linha do eixo X, sem
label nem ticks
    axis(1, at=seq(0,5, by=5)) #marca os primeiros 5 minutos de aquecimento
    axis(1, at=seq(35,40, by=5)) #marca os últimos 5 minutos de resfriamento
    ### conexão dos pontos plotados (liga os pontos)
    lines(tempo[1:2], fr[1:2], col="royalblue1") #conecta os pontos iniciais
(fora do trecho do treino de tiro)
    lines(tempo[2:32],fr[2:32], col="firebrick1") #conecta os pontos do
treino de tiro
    lines(tempo[32:34], fr[32:34], col="royalblue1") #conecta os últimos
pontos do gráfico
    rect(5.5, 44, 34.6, 37.5, lwd=1, border="firebrick1", lty="dashed")
#desenha um retângulo na zona de maior intensidade do treino de Tiro
    require(plotrix) #carrega o pacote plotrix para usar a função
boxed.labels()
    boxed.labels(   #destaca a zona de tiro no gráfico
                    x=20, #posição com relação ao eixo x
                    y=50, #posição com relação ao eixo y
                    "Aumentar a intensidade do treino durante os minutos
destacados", #texto exibido
                    cex=0.7, #fonte 30% menor que a padrão
                    border=F, #sem borda
                    col="black",
                    bg="white" #cor de fundo: branco
                )
    ### delimita a zona de tiro traçando linhas tracejadas
```

```
    segments(   #desenha linha preta tracejada mais fina no limite debaixo
da zona de tiro
                tempo[1], #coordenada x inicial
                zona_tiro[1:1], #coordenada y inicial
                tempo[length(tempo)], #coordenada x final
                zona_tiro[1:1], #coordenada y final
                lty=2, #linha tracejada
                lwd=1 #linha mais fina que o padrão definido anteriormente
            )
    segments(   #desenha linha preta tracejada mais fina no limite de cima
da zona de tiro
                tempo[1], #coordenada x inicial
                zona_tiro[2:2], #coordenada y inicial
                tempo[length(tempo)], #coordenada x final
                zona_tiro[2:2], #coordenada y final
                lty=2, #linha tracejada
                lwd=1 #linha mais fina que o padrão definido anteriormente
            )
    segments(   #desenha linha preta tracejada mais fina no limite debaixo
da zona de descanso
                tempo[1], #coordenada x inicial
                fr_mod, #coordenada y inicial
                tempo[length(tempo)], #coordenada x final
                fr_mod, #coordenada y final
                lty=2, #linha tracejada
                lwd=1 #linha mais fina que o padrão definido anteriormente
            )

    ### escreve no gráfico onde é a zona de tiro e descanso para recuperação
    boxed.labels(   #destaca a zona de tiro no gráfico
                    tempo[1], #localização vertical do box
                    (mean(zona_tiro)), #localização horizontal do box
                    paste("Zona de tiro \n FC: ", zona_tiro[1:1], "~",
zona_tiro[2:2], "bpm"), #texto exibido
                    cex=0.7, #fonte 30% menor que a padrão
                    border=F, #sem borda
                    bg="white", #cor de fundo: branco
                    col="firebrick1" #cor da fonte: vermelho firebrick1
                )
    boxed.labels(   #destaca a zona de descanso/recuperação no gráfico
                    tempo[1], #localização vertical do box
                    mean(c(fr_mod,zona_tiro[1:1])), #localização horizontal
do box
                    paste("Descanso para recuperação \n FC deve diminuir
para ~ ", fr_mod, "bpm"), #texto exibido
                    cex=0.7, #fonte 30% menor que a padrão
                    border=F, #sem borda
                    bg="white", #cor de fundo: branco
                    col="firebrick1" #cor da fonte: vermelho firebrick1
                )
    #algumas legendas importantes
```

```
    boxed.labels(   #legenda do primeiro ponto do gráfico
                    tempo[1],  #localização vertical do box
                    fr_leve-5, #localização horizontal do box
                    paste("FC ~",fr_leve), #texto exibido
                    cex=0.7, #fonte 30% menor que a padrão
                    border=F #sem borda
                )
    boxed.labels(   #legenda do último ponto do gráfico (igual anterior)
                    tempo[length(tempo)], #localização vertical do box
                    fr_leve-5, #localização horizontal do box
                    paste("FC ~",fr_leve), #texto exibido
                    cex=0.7, #fonte 30% menor que a padrão
                    border=F #sem borda
                )
    boxed.labels(   #legenda do penúltimo ponto do gráfico (igual anterior)
                    tempo[length(tempo)-2], #localização vertical do box
                    (0.8*fr_mod)-5,  #localização horizontal do box
                    paste("FC ~",0.8*fr_mod), #texto exibido
                    cex=0.7, #fonte 30% menor que a padrão
                    border=F #sem borda
                )
    boxed.labels(   #legenda do primeiro ponto da zona de
descanso/recuperação
                    tempo[1], #localização vertical do box
                    fr_mod-5, #localização horizontal do box
                    paste("FC ~",fr_mod),#texto exibido
                    cex=0.7, #fonte 30% menor que a padrão
                    border=F #sem borda
                )

    #montadem dos treinos
    if (duracao==4) #4 semanas de duracao do treino até a prova
    {
        treino=read.table("treinos_4.csv", as.is=T, header=T, sep=";",
check.names=F) #carrega o arquivo com as definicoes de treino para 4 semanas
de duração
    }
    else if (duracao==8) #8 semanas de duracao do treino até a prova
    {
        treino=read.table("treinos_8.csv", as.is=T, header=T, sep=";",
check.names=F) #carrega o arquivo com as definicoes de treino para 8 semanas
de duração
    }
    else #12 semanas de duracao do treino até a prova
    {
        treino=read.table("treinos_12.csv", as.is=T, header=T, sep=";",
check.names=F) #carrega o arquivo com as definicoes de treino para 12
semanas de duração
        #par(cex=0.9, cex.main=1.2) #como o gráfico vai ser um pouco maior,
diminui um pouco o tamanho de todos os labels
```

```
    }
    #verifica a distancia da prova para buscar o treino
    if (distancia==5) #prova de 5 quilômetros (5K) como objetivo final do
treino
    {
        #monta tabela com o treino de 5k
        treino=treino[,1:3] #atualiza data.frame com os treinos para 5K
    }
    else if (distancia==10) #prova de 10 quilômetros (10K) como objetivo
final do treino
    {
        treino=treino[,c(1,2,4)] #atualiza data.frame com os treinos para
10k
    }
    else #prova de 16 quilômetros (16K) como objetivo final do treino
    {
        treino=treino[,c(1,2,5)] #atualiza data.frame com os treinos para
16k
    }
    dias_semana=unique(treino$dias) #dias da semana de acordo com a duração
do treino
    qtde_dias=length(dias_semana) #conta o tamanho do vetor para dias da
semana distintos (quantos dias da semana são utilizados no treino)

    #monta o gráfico
    plot(   x=NULL, y=NULL, #plota gráfico sem informação
            xlim=c(1,duracao+3), #delimita o tamanho do eixo x
            ylim=c(1,qtde_dias+1), #delimita o tamanho do eixo y
            type="n", #não plota
            xaxt="n", yaxt="n", #sem desenhar eixo x e y
            xlab="", ylab="", #sem label de eixo x e y
            bty="n", #sem borda
            main=paste("Treino de",duracao,"semanas para prova
de",distancia,"K") #título do gráfico
        )

    segments(   #desenha linha preta no gráfico separando a primeira coluna
(esq)
                1, #coordenada x inicial
                1, #coordenada y inicial
                1, #coordenada x final
                qtde_dias+1, #coordenada y final
                lwd=1 #linha mais fina que o padrão definido anteriormente
            )
    segments(   #desenha linha preta no gráfico separando a primeira coluna
(dir) -- primeira coluna é mais larga
                3, #coordenada x inicial
                1, #coordenada y inicial
                3, #coordenada x final
                (qtde_dias+1), #coordenada y final
                lwd=1 #linha mais fina que o padrão definido anteriormente
```

```
            )

    i=4 #inicia o contador do loop (em 4 pq já desenhou as duas primeiras
linhas até a coord 3 do eixo x)
    dur=duracao+3 #soma 3 para desconsiderar as 3 coord do eixo x
    while (i <= (dur)) #desenha os últimos traços separando as colunas
    {
        segments(   #desenha linha preta no gráfico
                    i, #coordenada x inicial
                    1, #coordenada y inicial
                    i, #coordenada x final
                    qtde_dias+1, #coordenada y final
                    lwd=1 #linha mais fina que o padrão definido
anteriormente
                )
        mtext(side=3, at=(i-0.5), text=paste("Semana", i-3), cex=0.6) #acima
do gráfico, na margem superior, informa qual a semana do treino
        i=i+1 #atualiza o valor do contador
    }

    i=1 #inicia o contador do loop
    while (i <= (qtde_dias+1)) #desenha as linhas do gráfico
    {
        segments(   #desenha linha preta no gráfico
                    1, #coordenada x inicial
                    i, #coordenada y inicial
                    dur, #coordenada x final
                    i, #coordenada y final
                    lwd=1 #linha mais fina que o padrão definido
anteriormente
                )
        i=i+1 #atualiza o valor do contador
    }

    par(cex=0.8) #diminui um pouco o tamanho da fonte para caber dentro dos
quadros desenhados
    i=1 #inicia o contador do loop
    j=qtde_dias #inicia o contador para plotar o texto no gráfico
    while (i <= qtde_dias) #preenche no quadro os dias do treino
    {
        if (is.element("segunda",dias_semana)) #se o vetor tiver o valor
'segunda', entra nessa condicional
        {
            dia="Segunda-feira" #define o dia da semana que será escrito no
gráfico
            dias_semana=dias_semana[-(which(dias_semana=="segunda"))]
#remove essa informação do vetor... dessa forma consigo plotar no gráfico os
dias da semana na ordem certa

            ##########################################
```

```
            ###            W A R N I N G            ###
            #essa parte poderia ser uma outra função  #
            #  menor, mas não sabia se poderia criar  #
            #   duas no mesmo arquivo, então o código #
            #    está repetido                        #
            ##########################################
            x=1 #inicia o contador do loop
            while (1==1) #sem controle de loop no while, o controle é
interno, por isso, 1==1
            {
                if (treino[x,2]=="segunda") #varre o data.frame procurando
as "segunda"s
                {
                    k=treino[x,1]+2.5 #define a coordenada do eixo x
                    if (treino[x,3]=="Leve") cor="green3" #segue o padrão de
cor de acordo com cada treino
                    else if (treino[x,3]=="Longo") cor="royalblue1" #segue o
padrão de cor de acordo com cada treino
                    else if (treino[x,3]=="Tiro") cor="firebrick1" #segue o
padrão de cor de acordo com cada treino
                    else cor="green3" #cor diferente para o último dia do
treino
                    text(x=k, y=((j)+0.5), labels=treino[x,3], col=cor,
font=4) #preenche o treino das segundas-ferias
                }
                x=x+1 #atualiza o valor do contador
                if (x>nrow(treino)) #se o contador ultrapassou a quantidade
de linhas do data.frame
                {
                    break #termina o loop
                }
            }
        }
        else if (is.element("terca",dias_semana)) #se o vetor tiver o valor
'terca', entra nessa condicional
        {
            dia="Terça-feira" #define o dia da semana que será escrito no
gráfico
            dias_semana=dias_semana[-(which(dias_semana=="terca"))] #remove
essa informação do vetor... dessa forma consigo plotar no gráfico os dias da
semana na ordem certa

            x=1 #inicia o contador do loop
            while (1==1) #sem controle de loop no while, o controle é
interno, por isso, 1==1
            {
                if (treino[x,2]=="terca")  #varre o data.frame procurando as
"terca"s
                {
                    k=treino[x,1]+2.5 #define a coordenada do eixo x
                    if (treino[x,3]=="Leve") cor="green3" #segue o padrão de
```

```
cor de acordo com cada treino
                    else if (treino[x,3]=="Longo") cor="royalblue1" #segue o
padrão de cor de acordo com cada treino
                    else if (treino[x,3]=="Tiro") cor="firebrick1" #segue o
padrão de cor de acordo com cada treino
                    else cor="orangered" #cor diferente para o último dia do
treino
                    text(x=k, y=((j)+0.5), labels=treino[x,3], col=cor,
font=4) #preenche o treino das terças-ferias
                }
                x=x+1 #atualiza o valor do contador
                if (x>nrow(treino)) #se o contador ultrapassou a quantidade
de linhas do data.frame
                {
                    break #termina o loop
                }
            }
        }
        else if (is.element("quarta",dias_semana)) #se o vetor tiver o valor
'quarta', entra nessa condicional
        {
            dia="Quarta-feira" #define o dia da semana que será escrito no
gráfico
            dias_semana=dias_semana[-(which(dias_semana=="quarta"))]
#remove essa informação do vetor... dessa forma consigo plotar no gráfico os
dias da semana na ordem certa

            x=1 #inicia o contador do loop
            while (1==1) #sem controle de loop no while, o controle é
interno, por isso, 1==1
            {
                if (treino[x,2]=="quarta") #varre o data.frame procurando as
"quarta"s
                {
                    k=treino[x,1]+2.5 #define a coordenada do eixo x
                    if (treino[x,3]=="Leve") cor="green3" #segue o padrão de
cor de acordo com cada treino
                    else if (treino[x,3]=="Longo") cor="royalblue1" #segue o
padrão de cor de acordo com cada treino
                    else if (treino[x,3]=="Tiro") cor="firebrick1" #segue o
padrão de cor de acordo com cada treino
                    else cor="orangered" #cor diferente para o último dia do
treino
                    text(x=k, y=((j)+0.5), labels=treino[x,3], col=cor,
font=4) #preenche o treino das quartas-ferias
                }
                x=x+1 #atualiza o valor do contador
                if (x>nrow(treino)) #se o contador ultrapassou a quantidade
de linhas do data.frame
                {
```

```
                    break #termina o loop
                }
            }
        }
        else if (is.element("quinta",dias_semana)) #se o vetor tiver o valor
'quinta', entra nessa condicional
        {
            dia="Quinta-feira" #define o dia da semana que será escrito no
gráfico
            dias_semana=dias_semana[-(which(dias_semana=="quinta"))] #remove
essa informação do vetor... dessa forma consigo plotar no gráfico os dias da
semana na ordem certa

            x=1 #inicia o contador do loop
            while (1==1) #sem controle de loop no while, o controle é
interno, por isso, 1==1
            {
                if (treino[x,2]=="quinta") #varre o data.frame procurando as
"quinta"s
                {
                    k=treino[x,1]+2.5 #define a coordenada do eixo x
                    if (treino[x,3]=="Leve") cor="green3" #segue o padrão de
cor de acordo com cada treino
                    else if (treino[x,3]=="Longo") cor="royalblue1" #segue o
padrão de cor de acordo com cada treino
                    else if (treino[x,3]=="Tiro") cor="firebrick1" #segue o
padrão de cor de acordo com cada treino
                    else cor="orangered" #cor diferente para o último dia do
treino
                    text(x=k, y=((j)+0.5), labels=treino[x,3], col=cor,
font=4) #preenche o treino das quintas-ferias
                }
                x=x+1 #atualiza o valor do contador
                if (x>nrow(treino)) #se o contador ultrapassou a quantidade
de linhas do data.frame
                {
                    break #termina o loop
                }
            }
        }
        else if (is.element("sexta",dias_semana)) #se o vetor tiver o valor
'sexta', entra nessa condicional
        {
            dia="Sexta-feira" #define o dia da semana que será escrito no
gráfico
            dias_semana=dias_semana[-(which(dias_semana=="sexta"))] #remove
essa informação do vetor... dessa forma consigo plotar no gráfico os dias da
semana na ordem certa

            x=1 #inicia o contador do loop
            while (1==1) #sem controle de loop no while, o controle é
```

```
interno, por isso, 1==1
            {
                if (treino[x,2]=="sexta") #varre o data.frame procurando as
"sexta"s
                {
                    k=treino[x,1]+2.5 #define a coordenada do eixo x
                    if (treino[x,3]=="Leve") cor="green3" #segue o padrão de
cor de acordo com cada treino
                    else if (treino[x,3]=="Longo") cor="royalblue1" #segue o
padrão de cor de acordo com cada treino
                    else if (treino[x,3]=="Tiro") cor="firebrick1" #segue o
padrão de cor de acordo com cada treino
                    else cor="orangered" #cor diferente para o último dia do
treino
                    text(x=k, y=((j)+0.5), labels=treino[x,3], col=cor,
font=4) #preenche o treino das sextas-ferias
                }
                x=x+1 #atualiza o valor do contador
                if (x>nrow(treino)) #se o contador ultrapassou a quantidade
de linhas do data.frame
                {
                    break #termina o loop
                }
            }
        }
        else if (is.element("sabado",dias_semana)) #se o vetor tiver o valor
'sabado', entra nessa condicional
        {
            dia="Sábado" #define o dia da semana que será escrito no gráfico
            dias_semana=dias_semana[-(which(dias_semana=="sabado"))] #remove
essa informação do vetor... dessa forma consigo plotar no gráfico os dias da
semana na ordem certa

            x=1 #inicia o contador do loop
            while (1==1) #sem controle de loop no while, o controle é
interno, por isso, 1==1
            {
                if (treino[x,2]=="sabado") #varre o data.frame procurando os
"sabado"s
                {
                    k=treino[x,1]+2.5 #define a coordenada do eixo x
                    if (treino[x,3]=="Leve") cor="green3" #segue o padrão de
cor de acordo com cada treino
                    else if (treino[x,3]=="Longo") cor="royalblue1" #segue o
padrão de cor de acordo com cada treino
                    else if (treino[x,3]=="Tiro") cor="firebrick1" #segue o
padrão de cor de acordo com cada treino
                    else cor="orangered" #cor diferente para o último dia do
treino
                    text(x=k, y=((j)+0.5), labels=treino[x,3], col=cor,
```

```
font=4) #preenche o treino dos sábados
                }
                x=x+1 #atualiza o valor do contador
                if (x>nrow(treino)) #se o contador ultrapassou a quantidade
de linhas do data.frame
                {
                    break #termina o loop
                }
            }
        }
        else #se o vetor tiver o valor 'domingo', entra nessa condicional
(else, pq só pode ser domingo agora...)
        {
            dia="Domingo" #define o dia da semana que será escrito no
gráfico
            dias_semana=dias_semana[-(which(dias_semana=="domingo"))]
#remove essa informação do vetor... dessa forma consigo plotar no gráfico os
dias da semana na ordem certa

            x=1 #inicia o contador do loop
            while (1==1) #sem controle de loop no while, o controle é
interno, por isso, 1==1
            {
                if (treino[x,2]=="domingo") #varre o data.frame procurando
os "domingo"s
                {
                    k=treino[x,1]+2.5 #define a coordenada do eixo x
                    if (treino[x,3]=="Leve") cor="green3" #segue o padrão de
cor de acordo com cada treino
                    else if (treino[x,3]=="Longo") cor="royalblue1" #segue o
padrão de cor de acordo com cada treino
                    else if (treino[x,3]=="Tiro") cor="firebrick1" #segue o
padrão de cor de acordo com cada treino
                    else cor="orangered" #cor diferente para o último dia do
treino
                    text(x=k, y=((j)+0.5), labels=treino[x,3], col=cor,
font=4) #preenche o treino ou prova do domingo
                }
                x=x+1 #atualiza o valor do contador
                if (x>nrow(treino)) #se o contador ultrapassou a quantidade
de linhas do data.frame
                {
                    break #termina o loop
                }
            }
        }
        text(x=2, y=((j)+0.5), labels=dia, font=2) #preenche os dias da
semana
        i=i+1 #atualiza o valor do contador
        j=j-1 #atualiza o valor do contador (vai diminuindo a medida que os
dias vão sendo plotados no gráfico)
```

```
    }

    dev.off() #fecha o arquivo PDF

##########################################################################
#########
    ###                                 O U T P U T
###
##########################################################################
#########
    output=matrix(c(fr_leve, fr_mod, FCM), ncol=3)
    colnames(output)=c("FC treino Leve", "FC treino Moderado", "FC Máxima")
    rownames(output)="(bpm)"
    return(output) #retorna o valor da frequencia cardíaca leve, moderada e
da FCM em uma tabela simples

}
```

From:
http://ecor.ib.usp.br/ - **ecoR**

Permanent link:
http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2018:alunos:trabalho_final:camila.baruchi:codigo_da_funcao

Last update: **2020/08/12 06:04**