BIE5782

Unidade 7

NOÇÕES DE PROGRAMAÇÃO

function
Programação

function()
conjunto de comandos concatenados, desempenhando um fim!

TAREFA!
LINHA DE MONTAGEM conduzida por um algorítmo!

TODO USUÁRIO É UM PROGRAMADOR!!!

Algorítmo

"..é uma sequência não ambígua de instruções que é executada até que determinada condição se verifique. Mais especificamente, em matemática, constitui o conjunto de processos (e símbolos que os representam) para efetuar um cálculo."

Algorítmo

SEQUÊNCIA DE INSTRUÇÕES PARA REALIZAR UMA TAREFA

METÁFORA DO COZINHEIRO

Como fazer um Petit Gâteau?

- objetos (ovo, manteiga, farinha, chocolate, panelas, forno)

- "Funções" que manipulam os objetos

-tarefas sequenciais

RECEITA ≈ ALGORÍTMO DA COZINHA

FUNÇÕES

OBJETO DA CLASSE 'FUNCTION'

NOME (não trivial)

1. **tarefa realizada: summary()**
2. **revela objetivo: plot()**
3. **acrônimo: lm(); dp()**
4. **memorável sunflowerplot()**
5. **não usual!: caso do pi**

FUNÇÕES

ARGUMENTOS

objetos e parâmetros necessários para executar a tarefa de uma função (cuidado com o padrão!)

```
sum(x, na.rm = FALSE)

read.table(file, header = FALSE, sep = "", quote
    = "\"'", dec = ".", row.names, col.names,
    as.is = !stringsAsFactors, na.strings = "NA",
    colClasses = NA, nrows = -1, skip = 0,
    check.names = TRUE, fill = !blank.lines.skip,
    strip.white = FALSE, blank.lines.skip = TRUE,
    comment.char = "#", allowEscapes = FALSE,
    flush = FALSE, stringsAsFactors =default
    .stringsAsFactors(),encoding = "unknown")
```

function()
Estrutura Básica

```
minha.funcao <- function( argumento1,
                    argumento2,. . .)
                {
                    comando 1
                    comando 2
                    comando 3 . . .
                    return()
                }
```

eda.shape
Estrutura Básica

```
eda.shape <- function(x)
   {
   par(mfrow = c(2,2))
   hist(x)
   boxplot(x)
   iqd <- summary(x)[5] - summary(x)[2]
   plot(density(x,width=2*iqd),xla
   b="x",ylab="",type="l")
   qqnorm(x)
   qqline(x)
   par(mfrow=c(1,1))
   }
```

Senta que la vem história!

Modificando a função EDA.SHAPE

function ()

Funções Simples

```
media <-function(x)
    {
    soma=sum(x)
    nobs=length(x)
    media=soma/nobs
    return(media)
    }
```

If(){}; else{}

Funções Simples

```
If(condição)
    {
    comandos se condição = TRUE
    }
else
    {
    comandos se condição = FALSE
    }
```

for(){}

Funções com ciclos

dados = matrix (spp , parcelas)

```
n.spp<-function(dados)
    {
    nplot=dim(dados)[2]
    resultados=rep(0,nplot)
    names(resultados)<-paste("n.spp",c(1:nplot))
    dados[dados>0]=1
    for(i in 1:(dim(dados)[2]))
        {
        cont.sp=sum(dados[,i])
        resultados[i]=cont.sp
        }
    return(resultados)
    }
```

debug() ; undebug ()

Programando

Função (desespero total!!!!)
Caso tenha conseguido "source" da função e não consegue entender o erro ao aplicar a função...
use:
debug(função)
-roda a função para cada comando e pode mostrar as variáveis
-problema: toda vez que chamar a função ela estará em debug...

-para sair deve dar "Q" e depois:

undebug(função)

Editando Código

A edição de programa pode ser feito por qq. editor de texto.

- NOTEPAD
- WinEdt (pacote no CRAN)

Salvar em formato compativel ASCII
(American Standard Code for Information Interchange)

Para rodar no R deve antes ser carregado pelo source()

```
source("C:\\Users\\Alexandre\\r\\aula\\2008
\\aulaProgramar\\eda.shape.R")
```