Lista 4 - Introdução a Análise de Dados
Funções & Loops
Gabarito

Guilherme Masuko

April 2023

Questão 1

Crie dois loops. O primeiro deverá printar na tela os valores de 1 à 10 fazendo uso do `for`. O segundo deverá printar no console os valores de 1 à 5 usando o comando `while`.

Solução

```
rm(list=ls())


# declarando a variável a ser utilizada
x <- 1

for (i in 1:11) {
  print(x)
  # precisamos ir incrementando 1 a cada iteração
  x <- x + 1
}


# declarando a variável a ser utilizada
x <- 1

while (x <= 5) {
  print(x)
  # precisamos ir incrementando 1 a cada iteração
  x <- x + 1
```

```
}
```

Questão 2

Crie duas funções, `ate_n_for` e `ate_n_while`, que recebam um parâmetro n e executam os comandos que fizemos na questão anterior. Isso é, a primeira função deverá printar na tela os valores de 1 à n fazendo uso do `for`. A segunda deverá printar no console os valores de 1 à n usando o comando `while`.

Solução

```
rm(list=ls())

# funções

ate_n_for <- function (n) {
  # declarando a variável a ser utilizada
  x <- 1

  for (i in 1:n) {
    print(x)
    # precisamos ir incrementando 1 a cada iteração
    x <- x + 1
  }

}

# teste

ate_n_for(15)


ate_n_while <- function (n) {
  # declarando a variável a ser utilizada
  x <- 1

  while (x <= n) {
    print(x)
    # precisamos ir incrementando 1 a cada iteração
    x <- x + 1
  }
}
```

```
# teste

ate_n_while(15)
```

Questão 3

Crie um loop que imprima no console a tabuada de 1 à 10.

Solução

```
rm(list=ls())


# loop para o primeiro elemento do produto
for (x1 in 1:10) {
  # loop para o segundo elemento do produto
  for (x2 in 1:10) {
    # resultado do produto
    y = x1 * x2
    print(paste(x1,'x',x2,'=',y))
  }
}
```

Questão 4

a) Faça um loop para fazer o somatório de um vetor que vai de 1 à 10.

$$1 + 2 + 3 + ... + 10$$

Solução

```
rm(list=ls())

# declarando uma variável que guardará o valor da soma
soma = 0

# loop
for (i in 1:10) {
  soma = soma + i
}
```

```
print(soma)
```

b) Agora crie uma função que recebe o valor n e retorna o valor da soma.

$$1 + 2 + 3 + ... + n$$

Solução

```
rm(list=ls())



somaSequencia <- function(n) {
  # declarando uma variável que guardará o valor da soma
  soma <- 0
  # loop
  for (i in 1:n) {
    soma = soma + i
  }
  return(soma)
}

# teste

somaSequencia(18)
```

Questão 5

a) Faça um loop que realize o cálculo do fatorial de 10.

$$10! = 1 \cdot 2 \cdot 3 \cdot ... \cdot 10$$

Solução

```
rm(list=ls())

# declarando uma variável que guardará o valor do produtório
```

```
fatorial = 1

# loop
for (i in 1:10) {
  fatorial = fatorial*i
}

print(fatorial)
```

b) Agora crie uma função que recebe o valor n e retorne o valor do fatorial.

$$n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$$

Solução

```
rm(list=ls())


calculaFatorial <- function (n) {
  # declarando uma variável que guardará o valor do produtório
  fatorial = 1

  # loop
  for (i in 1:n) {
    fatorial = fatorial*i
  }
  return(fatorial)
}


calculaFatorial(14)
```

Questão 6

a) Use o loop `while` para investigar o número de termos necessários antes que o produto

$$1 \cdot 2 \cdot 3 \cdot ...$$

alcance mais que um milhão.

Solução

```
rm(list=ls())

# declarando variáveis

# variável que guardará o valor do produtório
x <- 1
# variável que guardará o número de termos
count <- 0

# loop
while (x <= 1000000) {
  count <- count + 1

  # a cada iteração multiplica-se o valor agregado pelo número
     à frente do que se multiplicou na iteração passada
  x = x*(count+1)
}

print(count)
```

b) Agora faça uma função que receba o valor n e retorne o número de termos necessários antes que o produto acima alcance mais que n.

Solução

```
rm(list=ls())

# função

termos_necessarios <- function(n) {
  # declarando variáveis

  # variável que guardará o valor do produtório
```

```
  x <- 1
  # variável que guardará o número de termos
  count <- 0

  while (x <= n) {
    count <- count + 1

    # a cada iteração multiplica-se o valor agregado pelo número
       à frente do que se multiplicou na iteração passada
    x = x*(count+1)
  }
  return(count)
}

# teste

termos_necessarios(121)
```

Questão 7

A sequência de Fibonacci[1] é uma sequência de números inteiros, começando normalmente por 0 e 1, na qual o termo subsequente corresponde à soma dos dois anteriores.

Os números que compõem a sequência de Fibonacci são:

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, ...$$

A sequência é definida recursivamente pela fórmula abaixo.

$$F_n = F_{n-1} + F_{n-2}$$

e valores iniciais $F_1 = 0$ e $F_2 = 1$.

Notação: A notação $(F_n)_{n\in A}$ é usada para denotar a sequência F, cujos índices são tomados no conjunto A. Quando o conjunto dos índices A está subentendido, normalmente escrevemos $(F_n)_n$ ou, simplesmente, (F_n). Por extenso, escrevemos $(F_n)_n = (F_1, F_2, F_3, \ldots)$. Observamos, ainda, que as notações $\{F_n\}_{n=1}^{\infty}$ e $\{F_n\}$ também são encontradas.

[1]<https://pt.wikipedia.org/wiki/Sequ%C3%AAncia_de_Fibonacci>

a) Crie um vetor contendo os primeiros 50 valores da sequência de Fibonacci. Utilize o comando `for` para adicionar novos elementos ao vetor.

Solução

```
rm(list=ls())


# criando um vetor que guardará a sequência de Fibonacci com os
   dois primeiros elementos da sequência
fib = c(0,1)

# o loop começa no índice 3 porque já definimos os dois
   primeiros elementos
for (i in 3:50) {

  # cada elemento subsequente é definimo pela soma dos dois
     anteriores
  fib[i] = fib[i-1] + fib[i-2]
}

print(fib)
```

b) Crie uma função que recebe como parâmetro o valor n. A função deve retornar um vetor contendo os primeiros n valores da sequência de Fibonacci utilizando o programa feito a partir do comando `for`.

Solução

```
rm(list=ls())


# função

fibonacciFor <- function (n) {
  # criando um vetor que guardará a sequência de Fibonacci com
     os dois primeiros elementos da sequência
  fib = c(0,1)

  # o loop começa no índice 3 porque já definimos os dois
     primeiros elementos
```

```
  for (i in 3:n) {

    # cada elemento subsequente é definimo pela soma dos dois
       anteriores
    fib[i] = fib[i-1] + fib[i-2]
  }
  return(fib)
}

# teste

fibonacciFor(10)
```

Questão 8

a) Crie um vetor contendo os primeiros 50 valores da sequência de Fibonacci. Utilize o comando `while` para adicionar novos elementos ao vetor.

Solução

```
rm(list=ls())



# criando um vetor que guardará a sequência de Fibonacci com os
   dois primeiros elementos da sequência
fib = c(0,1)

# criando uma variável que conta a quantidade de elementos na
   sequência
i <- 3
while (i <= 50) {
  # cada elemento subsequente é definimo pela soma dos dois
     anteriores
  fib[i] = fib[i-1] + fib[i-2]

  # a cada iteração a sequência ganha um novo elemento
  i <- i + 1
}

print(fib)
```

b) Crie uma função que recebe como parâmetro o valor n. A função deve retornar um vetor contendo os primeiros n valores da sequência de Fibonacci utilizando o programa feito a partir do comando `while`.

Solução

```
rm(list=ls())



fibonacciWhile <- function (n) {
  # criando um vetor que guardará a sequência de Fibonacci com
     os dois primeiros elementos da sequência
  fib = c(0,1)

  # criando uma variável que conta a quantidade de elementos na
     sequência
  i <- 3
  while (i <= n) {
    # cada elemento subsequente é definimo pela soma dos dois
       anteriores
    fib[i] = fib[i-1] + fib[i-2]

    # a cada iteração a sequência ganha um novo elemento
    i <- i + 1
  }
  return(fib)
}

# teste

fibonacciWhile(10)
```