

# Lab1 - Aula 1

*Vanderlin Amorim Júnior - Curso Verão 2016 - R*

*25 de fevereiro de 2016*

## R Markdown

### Aula 01 - Laboratório

- 1) Uma função em R, chamada `moeda` com o parâmetro `p` a probabilidade de ser cara. A função deve retornar 1 se o resultado do lançamento for cara e 0 caso contrário. usando a função `runif(n, min = 0, max = 1)` A função `runif(1)` gera um número aleatório no intervalo unitário toda vez que a condição é avaliada Função `moedas`

```
moeda <- function(p){  
  x<- runif(1)  
  if(x < p){  
    1  
  } else {  
    0  
  }  
}
```

- 2) Uma função chamada `moedas` com os parâmetros `n` e `p` que simula o lançamento de `n` moedas com probabilidade `p` de serem cara e retorna um vetor de 0's e 1's com os resultados obtidos em cada um dos lançamentos. Função `moedas`

```
moedas <- function(n, p){  
  y <- c(1:n)  
  for(i in 1:n) {  
    x <- runif(1)  
    if(x < p){  
      y[i]<-1  
    } else {  
      y[i]<-0  
    }  
  }  
  return(y)  
}
```

- 3) Uma função chamada `proporcao` com parâmetro `resultados` (que é um vetor de 0's e 1's) e retorna a proporção de 1's neste vetor.

```
resultados <- c(1,1,1,1,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,1)  
  
proporcao <- function(resultados) {  
  x = 0  
  y = 0  
  for(i in seq_along(resultados)) {  
    if(resultados[i] == 1) {
```

```

        x = x +1
    }
    else
    {
        y = y+1
    }
}
return (x/y)
}

proporcao(resultados)

```

```
## [1] 0.3888889
```

- 4) Uma função chamada simulacao que tem os parâmetro k, n e p. Essa função deverá repetir o seguinte processo k vezes: simular o lançamento de n moedas com probabilidade p de ser cara. calcular a proporção de caras (1's) obtidos nesses n lançamentos. salvar a proporção calculada em um elemento do vetor. A função deve retornar um vetor com todas as proporções obtidas.

```

simulacao <- function(k, n, p) {
  vetor <- c(1:k)
  for(i in 1:k) {
    ##   cat("Processo nº = ",i," ")
    vetor[i]<- proporcao(moedas(n,p))
    ##   print(i)
  }
  print(vetor)
}

simulacao(25,45,0.8)

```

```

## [1] 4.000000 3.090909 3.500000 8.000000 6.500000 3.090909 5.428571
## [8] 4.000000 4.000000 4.625000 6.500000 3.500000 2.000000 4.000000
## [15] 3.500000 3.090909 3.500000 4.000000 6.500000 2.214286 3.500000
## [22] 3.500000 3.090909 2.214286 4.000000

```