UNIVERSIDADE ABERTA
www.univ-ab.pt

21043

ESTATÍSTICA COMPUTACIONAL

ATIVIDADE FORMATIVA 1

PROPOSTA DE RESOLUÇÃO

DOCENTE: AMÍLCAR OLIVEIRA

2016/2017

1. **Explique qual o efeito quando se introduz o simbolo `#`, numa instrução escrita na linha de comandos do R.**

   R: Tudo o que se escreve à direita do simbolo # , não é considerado pelo programa R. É bastante útil quando se pretende colocar notas/comentários sobre alguma parte da escrita de um comando/script.

2. **Recorra aos comandos do R para indicar como obteria as seguintes sequência de números:**

   R: Para gerar a sequência S1 podemos utilizar a função seq( ), e para gerar S2 a funçã rep( ) como indicado de seguida:

   S1: 0,1,2,3,4,5,6,7,8,9

```
> S1<-seq(from=0,to=9)
> S1
 [1] 0 1 2 3 4 5 6 7 8 9
```

S2: 1,3,5,7,1,3,5,7,1,3,5,7,1,3,5,7,1,3,5,7,1,3,5,7,1,3,5,7,1,3,5,7,1,3,5,7,1,3,5,7

```
> S2<-rep(c(1,3,5,7),9)
> S2
 [1] 1 3 5 7 1 3 5 7 1 3 5 7 1 3 5 7 1 3 5 7 1 3 5 7 1
3 5 7 1 3 5 7 1 3 5 7
```

3. **Use o comando de controlo `if/else` para simular no R, 1000 lançamentos de um dado não viciado.**

Através do código seguinte, é possível gerar uma amostra de 1000 valores simulando o lançamento de um dado não viciado. Em cada iteração geramos um NPA Uniforme [0,1], valor que de seguida é comparado e considerado numa das 6 classes predefinidas de igual amplitude, cada uma correspondendo a uma das faces do dado.

```
> amostra<-NULL
> for (k in 1:1000){
+ u<-runif(1,0,1)
```

```
+ if (u<=1/6) amostra[k]<-1
+ if ((1/6<u)&(u<=2/6)) amostra[k]<-2
+ if ((2/6<u)&(u<=3/6)) amostra[k]<-3
+ if ((3/6<u)&(u<=4/6)) amostra[k]<-4
+ if ((4/6<u)&(u<=5/6)) amostra[k]<-5
+ else if (u>5/6) amostra[k]<-6
+ }
> amostra
```

4. Calcule no R o valor da expressão, $\log(5)+\exp(5)-\sqrt{2^3}+\frac{1}{2}$

```
> log(5)+exp(5)-(2^(3/2))+1/2
[1] 147.6942
```

5. Crie no R, uma matriz A, quadrada (4x4) constituída por valores com distribuição normal reduzida e de seguida obtenha também no R:

Começa-se por gerar os 16 valores X de uma distribuição normal reduzida, ou seja com média 0 e variância 1, seguindo-se a criação da matriz A contendo esses valores.

```
> X<-rnorm(16,0,1)
> A<-matrix(c(X),nrow=4)
> A
           [,1]       [,2]       [,3]       [,4]
[1,]  1.1418182  1.1011410  1.3339215 -0.5677240
[2,] -0.6485893 -1.4080420 -0.7650682 -0.3052205
[3,]  0.6447864 -0.4256799 -1.5792443 -1.1276403
[4,]  0.9665129  0.3879321 -1.1774345  1.1075577
```

5.1 A matriz transposta de A.

```
> t(A)
          [,1]       [,2]       [,3]       [,4]
[1,]  1.141818 -0.6485893  0.6447864  0.9665129
[2,]  1.101141 -1.4080420 -0.4256799  0.3879321
[3,]  1.333922 -0.7650682 -1.5792443 -1.1774345
[4,] -0.567724 -0.3052205 -1.1276403  1.1075577
```

5.2 A média dos elementos da 3ª coluna.

```
> mean (A[,3])
```

```
[1] -0.5469564
```

5.3 A variância da amostra constituída por todos os elementos da matriz A.

```
> var(X)
[1] 1.043387
```

5.4 A correlação entre os valores da 1ª e 2ª linhas.

Se considerarmos o cálculo do coeficiente de correlação de Pearson basta utilizar `cor(A[1,],A[2,])` uma vez que por defeito é esse o método a função cor( ) utiliza no R:

```
> cor(A[1,],A[2,])
[1] -0.641632
```

Caso entendêssemos mais conveniente a utilização de outro método, seria necessário especificar o argumento. Como no exemplo seguinte onde foi calculado o coeficiente de Spearman.

```
> cor(A[1,],A[2,], method="spearman")
[1] -0.4
```

6. Recorra aos comandos do R para indicar como calcular cada uma das probabilidades de cada um dos seguintes acontecimentos:

6.1 Uma variável aleatória distribuída como uma Normal (0,1) ser maior que 5.

```
> 1-pnorm(5,0,1)
[1] 2.866516e-07
```

6.2 Uma variável distribuída como uma normal de média 12 e variância 4 ser menor que 15.

```
> pnorm(15,12,2)
[1] 0.9331928
```

6.3 Obter 5 caras em 5 lançamentos de uma moeda sabendo que a probabilidade de sair cara é 0.6 .

```
> dbinom(5,prob=0.6,size=5)
[1] 0.07776
```

6.4 $X > 4$**, sendo *X* uma variável aleatória com distribuição** χ^2 **com 3 graus de liberdade.**

```
> 1-pchisq(4,df=3)
[1] 0.2614641
```

7. Usando os comandos em R indique os procedimentos para resolver cada um dos seguintes problemas:

7.1 Gerar 50 números pseudo-aleatórios com distribuição uniforme [0,5].

```
> runif(50,0,5)

 [1] 2.3695679 4.5354825 1.1895752 2.4672047 0.1261143
4.3316912 2.6769747

 [8] 2.3761980 2.3459825 4.5717042 1.5143237 1.2170188
3.2127373 0.1264629

[15] 0.5685214 4.7221515 0.8815900 4.6380659 0.8477339
3.9018988 0.7188288

[22] 4.2838249 2.2172316 4.3047715 3.7090747 3.6430448
3.8074807 2.2320779

[29] 3.6548988 2.7887706 0.1457198 0.7694856 3.2282513
1.4958000 2.9988139

[36] 4.1785695 4.7240717 4.6334342 0.4537085 3.3342317
4.2152611 3.5645106

[43] 0.7091664 0.3190749 3.6592228 2.3398739 1.8853015
4.6968816 2.7911117

[50] 0.3942452
```

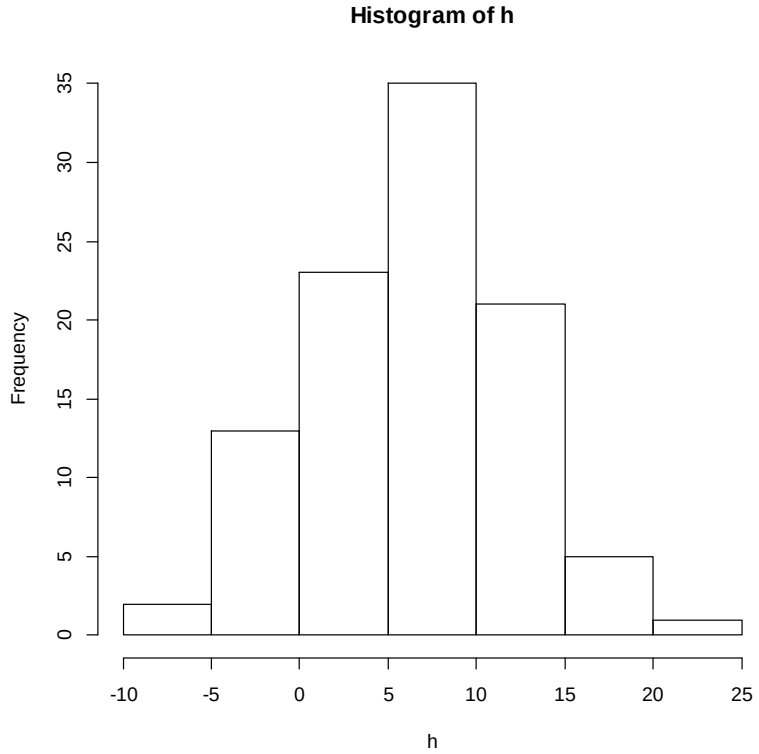
7.2 Gerar 100 números pseudo-aleatórios com distribuição normal (5,5).

```
> h <-rnorm(100,5,5)

> h

  [1]   0.74362033 14.98284949   5.21502873 10.36735455
5.67662597  8.81428068

  [7] -5.80728560   0.63056293 12.40960610   3.67359213
9.35689533  4.33743172
```

```
 [13]   8.45730104 -0.83231815 -2.11369643  5.36855601
10.24151150 -0.73461491

 [19]   4.63447214  0.29574397 -1.40880125  3.19586386
5.71004282  6.29089924

 [25]   8.33400973  3.06592356  4.80159685  7.91349666
14.38207526  8.71099015

 [31]  8.07327851  3.65178955 15.81074672 10.83124973 -
0.22778461  7.67966037

 [37]   4.08242504 -6.02283645 10.93876180 11.21511607
2.68532199 12.54224141

 [43] 12.32119169 13.76837352  3.33590507 -1.96034760
11.65849937  6.08723262

 [49]  0.81101948  6.50755669  6.47217486  2.81295864
0.99068433 -0.75436632

 [55] 13.42770431 17.13573176 10.98149569  5.86683635 -
0.68252580  5.48516147

 [61] -0.38594672  9.79554109 10.29705832  7.97148056 -
1.94037233 14.20861895

 [67] 13.81103474  7.04610084  8.48734812 17.94808050
9.49516280  3.51211958

 [73]  5.12737978 15.34324659 10.16792502  6.15436551
4.17683080  5.03946183

 [79]  8.10659318 -2.90521480 15.65308577  4.88815891
10.82925372 -0.07933854

 [85] 11.42922007  6.80226158 -1.01632773  2.19635548
3.19318379 21.86535655

 [91] 11.85287320  8.27352405  6.89939125  6.89554779
3.94962135  7.39403321

 [97]  7.39444911  6.21183581  7.15200043  1.64033209
```

7.3 Represente num histograma os valores obtidos na alínea anterior.

```
> hist(h)
```



8. Elabore uma rotina e implemente-a no R, por forma a obter números pseudo-aleatórios com distribuição X:

k	1	2	4	5
P(X=k)	2/ 9	1 / 9	5 / 9	1 / 9

```
> X<-NULL
> n<-100
> for (k in 1:n){
+ u<-runif(1,0,1)
+ if (u<=2/9) X[k]<-1
+ if ((2/9<u)&(u<=3/9)) X[k]<-2
+ if ((3/9<u)&(u<=8/9)) X[k]<-4
+ if (u>8/9) X[k]<-5
+ }
> X
  [1] 5 2 2 4 4 4 1 1 4 4 4 1 4 4 2 4 1 2 4 2 1 4 4 2
4 1 2 4 4 4 1 4 5 1 2
 [36] 1 4 1 4 4 4 4 4 4 4 4 1 5 1 1 1 4 1 4 5 4 4 5 2
4 4 1 4 4 4 5 1 5 4 4
 [71] 4 5 1 4 4 5 5 4 1 1 4 4 4 4 1 4 4 4 4 4 1 4 4 1
1 4 1 2 5 1
```

FIM