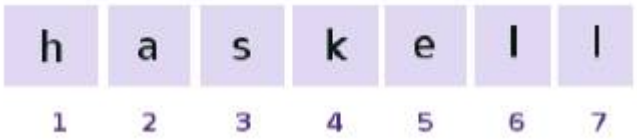
Aula Prática: Vetores e Matrizes

Vetores

Para se definir um vetor em Haskell usa-se a função **`array`**:

```
import Array
vet::Array Int Char
vet = array (1,7) [(1,'h'),(2,'a'),(3,'s'),(4,'k'),(5,'e'),(6,'l'),(7,'l')]
```

A declaração de tipos para `vet` indica que é um vetor (**`Array`**) cujos índices são inteiros e cujos elementos são caracteres.



Não há acesso direto à representação interna do vetor. O acesso e sua criação são possíveis através de funções. Um vetor pode ter como índices elementos negativos e não há restrição para a ordem na qual os índices aparecem na lista.

```
vet = array (1,7) [(5,'e'),(7,'l'),(3,'s'),(1,'h'),(2,'a'),(4,'k'),(6,'l')]
```

Definição de uma Matriz:

```
mat::Array (Int, Int) Int
mat = array ((1,1),(2,3)) [((1,1),4),((1,2),0),((1,3),8),
                           ((2,1),7),((2,2),1),((2,3),7)]
```

4	0	8
7	1	7

Exemplos de funções para manipulação de vetores/matrizes:

(1) O operador **`(!)`** permite acessar diretamente um elemento do vetor/matriz:

```
> vet ! 5
'e'
> mat ! (2,2)
1
```

(2) A função **`bounds`** retorna o índice inicial e a quantidade de células do vetor/matriz:

```
> bounds vet
(1,7)
> bounds mat
(1,1),(2,3)
```

(3) A função **elems** devolve os elementos do vetor/matriz:

```
> elems vet
"haskell"
> elems mat
[4,0,8,7,1,7]
```

(4) O operador **(//)** permite alterar momentaneamente o valor em certas posições:

```
> a // [(1,10)]
array (1,3) [(1,10),(2,6),(3,9)]
> a
array (1,3) [(1,4),(2,6),(3,9)]
```

(5) Uma segunda maneira de se criar uma matriz ou vetor é através da função **listArray**. Esta função exige que sejam indicados os limites dos índices e os valores dos elementos são obtidos a partir de uma lista:

```
> listArray ((1,1),(2,2))[3,4,2,5]
array ((1,1),(2,2)) [((1,1),3),((1,2),4),((2,1),2),((2,2),5)]
```

(6) A função **range** cria uma lista (de índices) a partir dos limites de um vetor/matriz:

```
> range (1,4)
[1,2,3,4]
> range ((1,1),(2,3))
[(1,1),(1,2),(1,3),(2,1),(2,2),(2,3)]
```

Exercícios:

1) Teste a função abaixo e explique sua finalidade:

```
conta [] = 0
conta (a:as) = 1 + conta as

vet l = array (1,conta l) [(i, l!!(i-1))| i<-[1.. conta l]]
```

2) Estude o programa abaixo que calcula a seguinte série simples utilizando vetores. Em seguida complete o programa substituindo os pontos de interrogação:

$$\frac{X_1^2}{1} + \frac{X_2^2}{2} + \ldots + \frac{X_n^2}{n}$$

```
b::Array Int Float
b= array (1,3) [(1,4.0),(2,6.0),(3,9.0)]
```

```
serie::Array Int Float -> Float
serie b = foldr1 (+) [ ((b!a)^2) / (fromIntegral a) |
                                    a <-  ???]
```

3) Faça como no exercício 2, uma função para calcular a série de Taylor utilizando vetores. Neste caso, faça uma função para armazenar em vetor todos os números gerados, e outra para somar tais números.

$$1 + \frac{X}{1!} + \frac{X^2}{2!} + \frac{X^3}{3!} \ldots + \frac{X^n}{n!} = e^x$$

4) Faça uma função que receba uma matriz e devolva a sua transposta.