Introducão

Depois do estudo do tipos básicos de dados em C (char, int, float e double) e as estruturas de controle (if, switch, for e do ... while), vamos agora estudar a forma de processar conjunto de dados do mesmo tipo.

Um vetor (ou array) não é mais que um conjunto de elementos consecutivos, do mesmo tipo, que podem ser acessados individualmente a partir de um único nome. Por exemplo:

1. **Conjunto de notas dos alunos em determinada prova.**
2. **Conjunto de pagamentos feito por zezinho no decorrer do ano.**

Declaracão de vetores

Um vetor é declarado da mesma forma que uma variável imples. `int n;`
Já a declaracão de um vetor de uma dimensão segue seguinte sintaxe: `int vet[5];` **ou mais generalizado** `tipo nome_variavel[n_elementos]`**.**

In [0]:

```
```

In [1]:

```
// exemplo de declaracao de um vetor de inteiros com 5 elementps
int vet[5] = {3, 2, 12, 42, 23};
// v[0] = 3.. v[1] = 2 .. v[4] = 23
```

Cada elemento de um vetor é representado por um index. A contagem dos index de um vetor com N elementos vai de 0 a n-1.

In [3]:

```
printf("%d %d %d", vet[1], vet[3], vet[4]);
```

```
2 42 23
```

Podemos iterar pelo vetor e acessar para acessar os seus elementos

In [10]:

```
for(int i = 0;i < 5;i++){
    printf("%d ", vet[i]);
}
```

```
3 2 12 42 23
```

Lendo os elementos do vetor a partir do scanf

In [0]:

```
for(int i = 0;i < 5;i++){
    scanf("%d", &v[i]);
}
```

Ordenação de vetores: Existem vários algoritmos de ordenação de vetores. Dois deles são o bubble sort e o selection sort

In [0]:

```
int n = 5;
```

```
for(int i = 0;i < n - 1;i++){
    for(int j = 0;j < n - i - 1;j++){
      if(vet[j] > vet[j + 1]){
        int temp = vet[j];
        vet[j] = vet[j + 1];
        vet[j + 1] = temp;
      }
    }
  }
```

Algoritmo de ordenação bubble sort.

In [0]:

```
for(int i = 0;i < n;i++){
    int ind = i;
    for(int j = i + 1;j < n;j++){
      if(vet[j] < vet[ind]){
        ind = j;
      }
    }
    int temp = vet[ind];
    vet[ind] = vet[i];
    vet[i] = temp;
 }
```

Algoritmo de ordenação selection sort.

Matrizes

A definição de matriz é bem semelhante a de vetor, adicionando que as matrizes contém mais dimensões.
Exemplo:

- **Jogo da velha é uma matriz 3x3**

In [1]:

```
char mat[3][3] = { {1, 2, 3}, {4, 5, 6}, {7, 8, 9} };
```

Da mesma forma que os vetores, podemos acessar os elementos do vetor usando os indexes.

In [2]:

```
printf("%d %d %d\n", mat[0][1], mat[1][1], mat[2][2]);
```

```
2 5 9
```

Podemos iterar a matriz para acessar os seus elementos

In [3]:

```
for(int i = 0;i < 3;i++){
    for(int j = 0;j < 3;j++){
        printf("%d ", mat[i][j]);
    }
    printf("\n");
}
```

```
1 2 3
4 5 6
7 8 9
```

Lendo os elementos da matriz a partir do scanf

In [4]:

```
for(int i = 0;i < 3;i++){
    for(int j = 0;j < 3;j++){
        scanf("%d", &mat[i][j]);
    }
}
```

Ex: Multiplicação de matrizes

In [4]:

```
int mat2[3][3] = { {9, 8, 7}, {6, 5, 4}, {3, 2, 1} };

for(int i = 0;i < 3;i++){
    for(int j = 0;j < 3;j++){
        int valor = 0;
        for(int k = 0;k < 3;k++){
            valor += mat[i][k] * mat2[k][j];
        }
        printf("%d ", valor);
    }
    printf("\n");
}
```

```
30 24 18
84 69 54
138 114 90
```

In [0]:

```

```