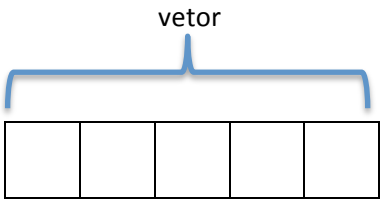
Vetores e matrizes

Vetores e matrizes são coleções de variáveis contínuas na memória e acessadas através de um número de índice. A diferença entre vetores e matrizes é que vetores são de uma única dimensão, enquanto matrizes podem conter várias dimensões.

Por exemplo, para alocar um vetor usamos o comando:

```
int vetor[5];
```

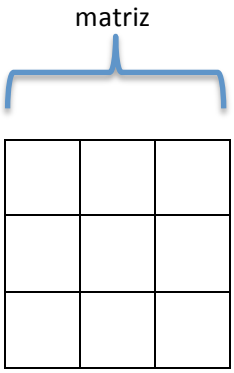
Que irá alocar o espaço de 5 valores inteiros na memória:



Perceba que o vetor foi criado sem nenhum valor. Na prática, será criado esse vetor com valores aleatórios que estão armazendo na posição da memória.

Enquanto uma matriz é criada da seguinte forma:

```
int matriz[3][3];
```



Criando (Declarando) um vetor/matriz

Todos os métodos abaixo são formas válidas de criar (declarar) um vetor/matriz.

```
int valores [6];    // forma 1
int valoresMatriz [6][6];

int pinos [] = {2, 4, 8, 3, 6};   // forma 2

int valores2[6] = {2, 4, 8, 3, 6};  // forma 3
int pinosMatriz[2][3] = {{2, 4, 8},  {3, 6, 9}};

char mensagem[6] = "hello";  // forma 3
```

1. A primeira forma de declaração é feita na forma 1, nesse caso, não são atribuídos valores, caso se queira ler algum dado do vetor/matriz, serão lidos valores quaiqueis que estão armazenados na memória na posição do vetor/matriz
2. Na forma 2, essa anotação se aplica apenas para vetores, em que não é definido o tamanho do vetor, mas sim, os elementos. Nesse caso, o compilador conta a quantidade de elementos e aloca o espaço necessário.
3. Na forma 3, foram especificados a quantidade de elementos e os valores, veja que no caso do vetor, ele foi criado com tamanho 6, porém tem 5 elementos, nesse caso, o elemento faltante terá valor aleatório armazenado em seu local correspondente na memória. Já a matriz foi declarada com 2 linhas e em cada linha com 3 colunas
4. Na forma 4, é um caso especial para a criação de sequências de caracteres para formar textos e mensagens (Strings). Especificamente para o caso de Strings, um espaço deve ser reservado para armazenar o caractere "\0", que representa final de string e não aparece nas mensagens.

Acessando um elemento de um vetor/matriz

Os vetores/matrizes são indexadas em zero, ou seja, referente a inicialização do vetor/matriz, o primeiro elemento da matriz está no índice 0, portanto:

```
valores[0] = 1;
valores[1] = 2;
```

O primeiro elemento do vetor irá receber o valor 1, enquanto o segundo elemento será atribuído o valor 2.

Isso significa que em um vetor com dez elementos, último elemento terá índice 9:

```
 int valores[10] = {9,3,2,4,3,2,7,8,9,11};
// valores[9] contém o valor 11
// valores[10] é inválido e contém informações aleatórias (outro endereço de memória)
```

Ao contrário de BASIC ou JAVA, o compilador do Arduino (derivado da linguagem C) não verifica se o acesso à matriz está dentro dos limites legais do tamanho da matriz que você declarou.

Para atribuir um valor a uma matriz:

```
valores[0] = 10;
```

Para recuperar um valor de uma matriz:

```
x = valores[4] ;
```

Vetores/Matrizes e laços de repetição

Os vetores/matrizes são muitas vezes manipuladas através de laços de repetição, onde o contador do laços de repetição é usado como índice para cada elemento de matriz. Por exemplo, para imprimir os elementos de uma matriz sobre a porta serial, você poderia fazer da seguinte forma:

```
void setup() {
  Serial.begin(9600);
  int i;
  int valores[6] = {2, 4, 8, 3, 6, 9};

  for (i = 0; i<6; i++) {
     Serial.println (valores[i]);
  }
}
void loop() {
}
```

Para matrizes, deve utilizar 2 laços de repetição, um que percorerrá as linhas e outro as colunas:

```
void setup() {
  Serial.begin(9600);
    int i, j;
    int pinosMatriz[2][3] = {{2, 4, 8},  {3, 6, 9}};

    for (i = 0; i<2; i++) {
        for (j = 0; j<3; j++) {
                Serial.println (pinosMatriz[i][j]);
        }
             Serial.println("outra linha");
    }
}
void loop() {
}
```