# EXERCÍCIOS DE FIXAÇÃO Nº 06.1
## Básico sobre ponteiros

1) Qual a função dos operadores **&** e ***** quando associados a ponteiros? Exemplifique com código em C.
2) Por que é importante inicializar um ponteiro antes do seu uso?
3) As variáveis são sempre armazenadas nos mesmos endereços?
4) O que é indireção?
5) Como o compilador distingue o * usado para a multiplicação do * usado para "desreferenciamento" (acesso às informações existentes no endereço contido em um ponteiro) e do * usado para declarar um ponteiro?
6) Como os elementos de uma matriz são armazenados na memória?
7) Mostre duas maneiras de obter o endereço do primeiro elemento da matriz data[].
8) Quando uma matriz é passada para uma função, quais são as duas maneiras de determinar onde a matriz termina?
9) Cite seis operações que podem ser efetuadas com ponteiros e duas que não podem.
10) Suponha que você tenha dois ponteiros. Se o primeiro estiver apontando para o terceiro elemento de uma matriz do tipo *int* e o segundo para o quarto elemento da mesma matriz, que valor será obtido quando você subtrair o primeiro ponteiro do segundo?
11) Suponha que a matriz da questão anterior contenha valores do tipo *float*, que valor seria obtido com a subtração dos dois ponteiros?
12) Escreva uma declaração de um ponteiro chamado **char_ptr** para uma variável do tipo *char*.
13) Se um programa contivesse uma variável *int* chamada **coast**, como você declararia e utilizaria um ponteiro chamado **p_coast** para apontar para esta variável?
14) Continuando com o exercício 13, como você atribuiria o valor 100 à variável **coast** usando acesso direto e indireto?
15) Continuando com o exercício 14, como você imprimiria o valor do ponteiro juntamente com o valor para o qual ele está apontando?
16) Mostre como atribuir o endereço de um valor do tipo *float*, chamado **radius**, a um ponteiro.
17) Mostre duas maneiras de atribuir o valor 100 ao terceiro elemento da matriz **data[]**.
18) Explique o que faz cada linha do trecho do programa abaixo:

```
int x=1, y=2, z[10];
int *ip;
ip = &x;
y = *ip;
*ip = 0;
ip=&z[0];
```

19) Supondo o mesmo trecho de código do exercícios anterior, explique cada uma das operações aritméticas abaixo, que utilizam ponteiros:

```
a)   y = *ip+1;
b)   *ip += 1;
c)   ++*ip;
d)   (*ip)++;
```

20) Considerando o fragmento de programa abaixo

```
{
      int a[10];
      int *pa;
      int aux;
      ...
      pa = a;
}
```

Complete as equivalências abaixo usando os conceitos de aritmética de ponteiros:

| | |
|---|---|
| `aux = a[2]` | `aux = _____, usando "pa"` |
| `aux = a[i]` | `aux = _____, usando "pa"` |
| `aux = a[2]` | `aux = _____, usando "a"` |
| `aux = a[i]` | `aux = _____, usando "a"` |
| `( a+2 )` | `_____, usando "a"` |
| `(pa+1)` | `_____, usando "a"` |