EXERCÍCIOS DE FIXAÇÃO Nº 02.2
Continuação sobre ponteiros

1) Como os elementos de uma matriz são armazenados na memória?

Após a declaração, um endereço inicial na memória é alocado, juntamente com o endereço para cada um dos elementos subsequentes da matriz, levando em consideração o tamanho de cada um deles, em bytes. Apesar das linhas e colunas, os elementos são armazenados em sequência na memória.

2) Mostre duas maneiras de obter o endereço do primeiro elemento da matriz data[].

&data[0][0] ; data.

3) Quando uma matriz é passada para uma função, quais são as duas maneiras de determinar onde a matriz termina?

&mat[x] ; mat[x]

4) Cite seis operações que podem ser efetuadas com ponteiros e duas que não podem.

Podem ser realizadas: Adição, subtração, multiplicação. Não podem ser realizadas: divisão e módulo.

5) 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?

1.

6) Suponha que a matriz da questão anterior contenha valores do tipo float, que valor seria obtido com a subtração dos dois ponteiros?

1.

7) Explique o que faz cada linha do trecho do programa abaixo:

```
int x=1, y=2, z[10];
```
Declara duas variáveis do tipo inteiro, e uma matriz de inteiros de 10 elementos. Atribui o valor 1 para a primeira variável e 2 para a segunda.

```
int *ip;
```
Declara um ponteiro de nome "ip" do tipo inteiro.

```
ip = &x;
```

```
Atribui o endereço de "x" ao ponteiro "ip".

y = *ip;
Atribui o valor apontado por "ip" (1) à variável "y".

*ip = 0;
Altera o valor de "x" para 0 indiretamente, utilizando referenciação pelo ponteiro "ip".

ip=&z[0];
Atribui o endereço inicial da matriz "z" para o ponteiro "ip".
```

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

a)
```
y = *ip+1;
variável "y" recebe o valor apontado por "ip" + 1.
```

b)
```
*ip += 1;
Valor apontado por "ip" incrementa 1.
```

c)
```
++*ip;
Valor apontado por "ip" incrementa 1.
```

d)
```
(*ip)++;
Valor apontado por "ip" incrementa 1.
```

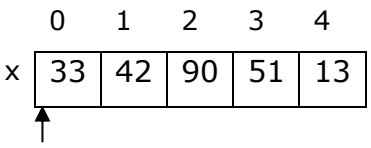
9) Considerando o fragmento de programa abaixo

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

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

aux = a[2]	aux = *(pa+2), usando "pa"
aux = a[i]	aux = *(pa+i), usando "pa"
aux = a[2]	aux = a+2, usando "a"
aux = a[i]	aux = a+i, usando "a"
(a+2)	a[2], usando "a"
(pa+1)	a[1], usando "a"

10) Considere a matriz x abaixo e responda às questões a seguir.



Suponha endereço 1234 para o início da matriz e considere que cada valor inteiro ocupe 4 bytes.

a) Explique as linhas de comando do programa abaixo.

```
...
main()
{
        int x[5];
        ...
        int *px;
        int i;
        px = x;
        for(i=0;i<5;i++)
        {
                printf("\n%i",*px);
                px++;
        }
        px = x;
        for(i=0;i<5;i++)
        {
                printf("\n%i",*(px+i));
        }
}
```

Inicializa a matriz, um ponteiro para o endereço inicial da matriz, e um contador para ser utilizado em 2 laços de repetição for, que exibem os valores do vetor na tela.

Explique a diferença no uso de ponteiro entre a 1ª e a 2ª estrutura de repetição no programa anterior.

O primeiro exibe o conteúdo do vetor a partir do ponteiro px, sempre somando 1 no endereço armazenado em px antes de iniciar uma nova repetição, enquanto o segundo utiliza o endereço armazenado em px como referência, e soma o valor de i em cada iteração, para gerar o endereço que será acessado, enquanto o valor de i é aumentado dentro do looping.

b) Explique as linhas de comando da estrutura de repetição do programa abaixo.

```
...
main()
{
        int x[5];
        ...
        int *px;
        int i;

        px = x;
        for(i=0;i<5;i++)
        {
                printf("\n%i",*(px++));
        }
        ...
}
```

Declara um vetor de 5 elementos, um ponteiro que recebe o endereço inicial do vetor, e um laço de repetição que mostra o valor apontado por px + 1, ou o segundo elemento do vetor x, e subsequentemente.