Prática 01 - Ponteiros

1. Explique a diferença entre:
   a. p++;
   b. (*p)++;
   c. *(p++);
   d. *(p+10);
2. Qual o valor de ***y*** no final do programa? Procure primeiro descobrir o resultado e só após verifique no computador. A seguir, escreva um /* comentário */ em cada comando de atribuição, explicando o que ele faz e o valor da variável à esquerda do '=' após sua execução.

```
int main() {
   int y, *p, x;
   y = 0;
   p = &y;
   x = *p;
   x = 4;
   (*p)++;
   x--;
   (*p) += x;
   printf ("y = %d\n", y);
   return 0;
}
```

3. Qual o objetivo do programa do trecho de código em **negrito**?

```
int main ( ){
      float matrx [50][50];
      float *p;
      int count;
      p=matrx[0];
      for (count=0;count<2500;count++){
            *p=0.0;
            p++;
      }
      return 0;
}
```

4. Seja um vetor declarado por **int vet[10].** Qual elemento deste vetor é acessado quando se escreve vet[2] (primeiro, segundo, terceiro, etc.)?
5. Se declararmos um vetor como **int vet[30],** a instrução abaixo acessa corretamente os elementos deste vetor?
   for (j=0; j <= 30; j++)

```
vet[j] = j*j;
```

6. Se uma string for declarada como **char str[20],** qual o número máximo de caracteres que poderão ser lidos e armazenados nela?
7. Qual função pode ser usada para determinar o comprimento de uma string?
8. Qual das instruções abaixo é correta para declarar um ponteiro para inteiro?
    a. *int pti
    b. *pti
    c. &i
    d. int_pti pti
    e. int *pti
9. Para a seguinte seqüência de instruções em um programa C, qual afirmativa é falsa:

```
int *pti;
int i = 10;
pti = &i;
```

    a. pti armazena o endereço de i
    b. *pti é igual a 10
    c. ao se executar *pti = 20; i passará a ter o valor 20
    d. ao se alterar o valor de i, *pti será modificado
    e. pti é igual a 10
10. Se i e j são variáveis inteiras e pi e pj são ponteiros para inteiro, qual atribuição é ilegal?
    a. pi = &i
    b. *pj = &j
    c. pj = &*&j
    d. i = *&*&j
    e. i = (*pi)+++*pj
11. Para a seguinte seqüência de instruções em um programa C, qual afirmativa é falsa?

```
int *pti;
int veti[]={10,7,2,6,3};
pti = veti;
```

    a. *pti é igual a 10
    b. *(pti+2) é igual a 2
    c. pti[4] é igual a 3
    d. pti[1] é igual a 10
    e. *(veti+3) é igual a 6
12. Na seqüência de instruções abaixo:

```
float f;
float *pf;
pf = &f;
scanf("%f", pf);
```

a. Efetuamos a leitura de f
b. Não efetuamos a leitura de f
c. Temos um erro de sintaxe
d. Deveríamos estar usando &pf no scanf
e. Nenhuma das opções anteriores

13. Para a seguinte seqüência de instruções, qual expressão não é válida?

```
int i=10, j=20;
int *pti, *ptj;
pti = &i;
ptj = &j;
```

a. j = pti = = ptj
b. i = pti-ptj
c. pti += ptj
d. pti++
e. i = pti || ptj;

14. Para a declaração **int matr[][4] = {1,2,3,4,5,6,7,8,9,10,11,12}**, qual afirmativa é falsa?
a. **matr é igual a 1
b. *(*(matr+1)+2) é igual a 7
c. *(matr[2]+3) é igual a 12
d. (*(matr+2))[2] é igual a 11
e. *((*matr)+1) é igual a 5

15. Dada a função StrCpy() abaixo, faça uma função StrLen() e StrCat() que funcionem como as funções strlen() e strcat() de string.h respectivamente.

```
#include <stdio.h>
void StrCpy (char *destino,char *origem){
      while (*origem!='\0') {
      *destino=*origem;
      origem++;
      destino++;
   }
      *destino='\0';
}
```