Exercícios Nomes: _____ e _____

1. Desenvolva uma função que recebe como parâmetro o número de elementos para um novo vetor que deverá ser alocado e retornado pela função. O vetor deve ser inicializado com letras minúsculas do alfabeto. Segue a função main().

```
main() {
   int i, tam=10;    char *vetor;
   vetor = preencheAlfabeto(tam);
   for (i=0; i<tam; i++)
       printf("%c ", vetor[i]);
}
```

2) Tendo a estrutura:

```
typedef struct funcionario {
    char nome[80];
    int idade;
    char sexo; // 'm' ou 'f'
    float salario;
} Funcionario;
```

a) Aloque na main um novo funcionário e atribua esse espaço a um ponteiro 'f' (somente a linha de alocação).

b) Desenvolva uma função que atribui os valores passados como parâmetro para os campos da estrutura Funcionario. Na main, a chamada é: `preencheFunc(f, "Fulano de tal", 40, 'm',2000.00);`

c) Implemente uma função que mostre todos os campos da estrutura Funcionario. Na main, é chamada: `mostraFunc(f);`

3) Corrija 7 erros **na main**. Não faltam linhas de código.

```
typedef struct ponto{
 int x;
 int y;
}Ponto;

typedef struct circulo{
   Ponto p;
   int raio;
}Circulo;


typedef struct pontoColorido {
   int x;
   int y;
   int cor;
} PontoColorido;

typedef struct circuloColorido {
   PontoColorido *p;
   int raio;
} CirculoColorido;


void preencheVetor(Circulo *ptr, int n) {
   int i;
   for (i=0; i<n; i++) {
      ptr[i].p.x = 10*i;
      ptr[i].p.y = 20*i;
      ptr[i].raio = 10;
   }
}

void mostraVetor(Circulo *ptr, int n) {
   int i;
   for (i=0; i<n; i++)
      printf("Elemento %d: x=%d, y=%d, raio=%d\n",
i, ptr[i].p.x, ptr[i].p.y, ptr[i].raio);
}

void imprimePC(CirculoColorido *cc) {
   printf("\nX=%d",cc->p->x);
   printf("\nY=%d",cc->p->y);
   printf("\nCor=%d",cc->p->cor);
   printf("\nRaio=%d\n",cc->raio);
}

CirculoColorido *criaCC(int x, int y, int cor, int
raio) {
   CirculoColorido *cc = (CirculoColorido *)
malloc(sizeof(CirculoColorido));
   cc->p = (PontoColorido *)
malloc(sizeof(PontoColorido));
   cc->p->x = x;
   cc->p->y = y;
   cc->p->cor=cor;
   cc->raio = raio;
   return cc;
}

int main (void) {
   Ponto p, p2, *p3, *p4;
   int i;

   // vetor de circulos
   Circulo vetorCirc[10];

   vetorCirc[0].p.x=10;
   vetorCirc[0].p.y=20;
   vetorCirc[0].raio=50;
   vetorCirc[9].p.x=20;

   preencheVetor(vetorCirc, 10);
   mostraVetor(vetorCirc, 10);

   p.x = 2;
   p2->x = 4;

   p3 = &p;
   p3->x = 3;

   p4 = (Ponto *) malloc(sizeof(PontoColorido));
   p4->x = 8;
   p4->y = 10;

   Circulo c, c2, *c3, *c4;
   c.p.x = 20;
   c.p.y = 50;
   c.raio = 40;

   c2.p.x = 2;
   c2.p.y = 5;
   c2.raio = 4;

   c3 = &c;
   c3->p.x = 40;   // c3->p.x = c3->p.x * 2;
   c3->p.y = 100;
   c3.raio = 80;

   c4 = (Circulo *) malloc(sizeof(Circulo));
   c4->p.x = 4;
   c4->p.y = 10;
   c4->raio = 8;

   CirculoColorido cc1;
   cc1.p = (PontoColorido *)
malloc(sizeof(PontoColorido));
   cc1.p->x = 10;
   cc1.p->y = 20;  cc1.p->cor=5;
   cc1->raio = 30;

   CirculoColorido cc2;
   cc2.p = (PontoColorido *)
malloc(sizeof(PontoColorido));
   cc2.p->x = 10;
   cc2.p->y = 20;  cc2.p->cor=5;
   cc2.raio = 30;
   imprimePC(&cc1);

   CirculoColorido *cc = criaCC(1,2,3,4);
   imprimePC(&cc);

   CirculoColorido *cc3;
   cc3 = &c2;
   cc3->p->x = 10;
   cc3->p->y = 20;
   cc3->raio = 30;

   CirculoColorido *cc4;
   cc4 = (CirculoColorido *)
malloc(sizeof(CirculoColorido));
   cc4->p = (Ponto *) malloc(sizeof(Ponto));
   cc4->p->x = 10;
   cc4->p->y = 20;
   cc4->p->cor = 2;
   cc4->raio = 30;
}
```