Universidade Federal de Minas Gerais
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Algoritmos e Estruturas de Dados – 1º. Semestre 2010 – 1a. Prova

1. Mostre o que é impresso na execução do trecho de programa abaixo (observe os espaços!):

```
int a[]={ 1,0,4,2,3}; int N=5;
int i,j;
for(i=0; i<N; i++) printf("%d ", a[i]);
printf("\n");
for(i=0; i<N; i++){
  for(j=0; j<N; j++)
     if (a[j]==i|i==j) printf("* ");
     else printf(". ");
  printf("\n");
}
```

2. Faça um desenho, conforme as convenções de sala de aula, que mostra as estruturas em memória construidas a partir das sentenças abaixo. O desenho deve mostrar o que é a1, a2 e a3.
(i) `int a1[]={10,20};` (ii) `int a2[]={30, 40, 50};`. (iii) `int *a3[]={a1, a2};`

3. Na linguagem C podemos passar uma estrutura ou um ponteiro para estrutura como parâmetro de função, além disso podemos ter funções retornando uma estrutura ou ponteiro para estrutura. Não podemos passar um arranjo como parâmetro de função, nem podemos ter funções que retornem arranjos. Escreva a função correta, utilizando ponteiros, correspondente à função embaixo à direita (que não funciona!) observando a analogia das funções f e g da parte de cima.

sem ponteiro	correspondente com ponteiro

`struct di {int i1; int i2;} //ok!`
`struct di f(int a, int b){`
`  struct di aux;`
`  aux.i1=a;  aux i2=b;`
`  return aux;`
`}`
	`struct di *g(int a, int b){ ok!`
	`  static struct di aux;`
	`  aux.i1=a;`
	`  aux.i2=b;`
	`  return &aux;`
	`}`

`int f[](int a, int b){ //não funciona!`
`  int aux[2];`
`  aux[0]=a;`
`  aux[1]=b;`
`  return aux;`
`}`	

4. Considere que `a[i][j]=v` modela que `v` alunos escolheram a opção `j` da questão `i` de uma prova. Complete o trecho de programa abaixo que preenche o arranjo gf. O elemento `gf[k]=x` indica que a letra `'A'+x` é a opção mais frequente para questão `k`.

```
//a[i][j]= numero de alunos que responderam letra 'A'+j na questão i
static int gf[5];
int indMaior, i, j;
for(i=0; i<MAXQUEST; i++){ //iteracao de questoes
  indMaior=0;
  for(j=0; j<MAXOP; j++) // iteracao de opcoes
     _____________________________;
  gf[i]=indMaior;
}
```

5. O método abaixo retorna a nota de um aluno a partir de um gabarito e respostas modelados como arranjos de caracteres. Considere `char *gab="ABCD"; char *r1="AAAA";` e `char *r2="A   ";` Qual o valor retornado por a) `nota(gab,r1,4)` e b) `nota(gab,r2,4)`?

```
int nota(char *gabarito, char *respostas, int MAXQUEST){
    int i,questoesCertas=0, questoesEmBranco=0;
    for(i=0; i<MAXQUEST; i++){
      if(gabarito[i]==respostas[i]) questoesCertas++;
      if(respostas[i]==' ')questoesEmBranco++;

    }
    return questoesCertas-(MAXQUEST-questoesEmBranco-questoesCertas)/3;
}
```

6. Um ***invariante*** de um laço corresponde a uma expressão lógica que pode utilizar as variáveis de um programa. O estabelecimento de um invariante auxilia no entendimento e nas provas de correção de programas. O invariante deverá ser verdadeiro quando o fluxo de controle chegar no corpo do laço e verdadeiro no final do laço. Isto significa que o invariante é verdadeiro quando o fluxo de controle abandona o laço (término da iteração). Considere o laço externo (for) do algoritmo de ordenação por inserção::

```
    for(i=1; i<N; i++){
      aux=a[i];
      j=i;
      while(j>0&&a[j-1]>aux) a[j]=a[--j];
      a[j]=aux;
    }
```

Um invariante deste "for" corresponde a: "os elementos em a[0]..a[i-1] são os elementos originalmente presentes mas ordenados". Qual é o invariante correspondente ao laço interno (invariante do "while")? Cuidado com o uso de < ou <=

7. O valor do arranjo correspondente à seguinte relação (conforme convenções da disciplina) : {(a,b), (a,c), (b,c)}. é `{{0,1,1},{0,0,1},{0,0,0}}` para uma relação com 3 elementos: `a`, `b` e `c` correspondentes aos índices 0,1 e 2 respectivamente. Qual o valor do arranjo correspondente à relação { (b,a), (c,a), (c,b)}?

8. Complete o método abaixo que recebe dois arranjos ordenados em ordem crescente. O método devolve um ponteiro para um arranjo ordenado, resultado da intercalação dos valores dos arranjos com ponteiros passados como parâmetro. Os parâmetros M1 e M2 correspondem ao número de elementos de a1 e a2 respectivamente.

```
int *intercalacao(int *a1, int M1, int *a2, int M2){
  if(M1+M2>MAX) return NULL;
  static int aux[MAX];
  int ind1=0;
  int ind2=0;
  while(_______(a)_______){
    if(a1[ind1]<a2[ind2])
      aux[ind1+ind2]=a1[ind1++];
    else
      aux[ind1+ind2]=a2[ind2++];
  }
  while(____(b)________) aux[ind1+ind2]=a1[ind1++];
  while(____(c)________) aux[ind1+ind2]=a2[ind2++];
  return aux;
}
```