Exercícios Extras - Lista 02

1) Faça um programa que inicie com um vetor de inteiros com 2 posições e recebe uma sequência de inteiros do usuário, até que "-1" seja informado. Armazene neste vetor cada inteiro fornecido, e quando não tiver posição vazia, dobre o tamanho do vetor.

2) Construa um programa (*main*) que aloque em tempo de execução (dinamicamente) uma matriz de ordem m x n (linha por coluna), usando 1+m chamadas da função malloc. Agora, aproveite este programa para construir uma função que recebendo os parametros m e n aloque uma matriz de ordem m x n e retorne um ponteiro para esta matriz alocada. Crie ainda uma função para liberar a área de memória alocada pela matriz. Finalmente, crie um novo programa (main) que teste/use as duas funções criadas acima.

3) Construa um programa (*main*) que aloque em tempo de execução (dinamicamente) uma matriz de ordem m x n (linha por coluna), usando apenas 2 chamadas da função malloc. Agora, aproveite este programa para construir uma função que recebendo os parametros m e n aloque uma matriz de ordem m x n e retorne um ponteiro para esta matriz alocada. Crie ainda uma função para liberar a área de memória alocada pela matriz. Finalmente, crie um novo programa (main) que teste/use as duas funções criadas acima.

4) O código a seguir possui alguns erros. Este procedimento deve solicitar uma matriz de números inteiros e em seguidar imprimir esta matriz na tela. Informe as correções necessárias (minimizando o número de alterações) para que este programa funcione conforme o esperado.

```
01  #include <stdio.h>
02  #include <stdlib.h>
03
04  void imprimeMatriz(int **m, int l, int c) {
05          int i, j;
06          for (i=0; i<l; i++) {
07                  for (j=0; j<c; j++)
08                          printf("%d\t", (m + (i * l) + j));
09                  printf("\n");
10          }
11  }
12  int **leitura(int l, int c) {
13          int *m;
14          int i, j;
15          m = (int *)malloc(l * c * sizeof(int));
16          if (m != NULL) {
17                  printf("Nao foi possivel alocar a matriz\n");
18                  exit(0);
19          }
20          for (i=0; i<l; i++)
21                  for (j=0; j<c; j++) {
22                          printf("Entre m[%d][%d]: ", i, j);
23                          scanf("%d", *(m + (j * c) + i));
24                  }
25          return m;
26  }
27  int main(void)
28  {
29          int **m;
30          int nlin, ncol;
31          int i, j;
32          printf("Entre o numero de linhas e colunas: ");
33          scanf("%d %d", &nlin, &ncol);
34          m = leitura(nlin, ncol);
35          imprimeMatriz(m, nlin, ncol);
36  }
```

5) Por que o código abaixo está errado?

```
01  void troca (int *i, int *j) {
02     int *temp;
03     *temp = *i; *i = *j; *j = *temp;
04  }
```

6) Um ponteiro pode ser usado para dizer a uma função onde ela deve depositar o resultado de seus cálculos. Escreva uma função hm que converta minutos em horas-e-minutos. A função recebe um inteiro mnts e os endereços de duas variáveis inteiras, digamos h e m, e atribui valores a essas variáveis de modo que m seja menor que 60 e que 60*h + m seja igual a mnts. Escreva também uma função main que use a função hm.

7) Responda:

(a) Suponha que os elementos de um vetor v são do tipo int e cada int ocupa 8 bytes no seu computador. Se o endereço de `v[0]` é 55000, qual o valor da expressão `v + 3`?

(b) Suponha que v é um vetor declarado `int v[100];` Descreva, em português, a sequência de operações que deve ser executada para calcular o valor da expressão `&v[k + 9]`.

(c) Suponha que v é um vetor. Descreva a diferença conceitual entre as expressões `v[3]` e `v + 3`.

8) Escreva uma função que conte o número de células de uma lista encadeada. Faça duas versões: uma iterativa e uma recursiva.

9) A altura de uma célula c em uma lista encadeada é a distância entre c e o fim da lista. Mais precisamente, a altura de c é o número de passos do caminho que leva de c até a última célula da lista. Escreva uma função que calcule a altura de uma dada célula.

10) A profundidade de uma célula c em uma lista encadeada é o número de passos do único caminho que vai da primeira célula da lista até c. Escreva uma função que calcule a profundidade de uma dada célula.

11) Escreva uma função que verifique se uma lista encadeada que contém números inteiros está em ordem crescente.

12) Escreva uma função que faça uma busca em uma lista encadeada crescente. Faça versões recursiva e iterativa.

13) Escreva uma função que receba uma lista duplamente encadeada e devolva o endereço de uma célula que esteja o mais próximo possível do meio da lista. Faça isso sem contar explicitamente o número de células da lista.

14) Por que a seguinte versão da função insere não funciona?

```
01  void insere (int x, no *p) {
02     no nova;
03     nova.conteudo = x;
04     nova.prox = p->prox;
05     p->prox = &nova;
06  }
```

15) Escreva uma função que inverta a ordem das células de uma lista simplesmente encadeada (a primeira passa a ser a última, a segunda passa a ser a penúltima etc.). Faça isso sem usar espaço auxiliar, apenas alterando ponteiros.

---

16) Implemente uma lista encadeada sem usar endereços e ponteiros. Use dois vetores paralelos: um vetor `conteudo[0..N-1]` e um vetor `prox[0..N-1]`. Para cada i no conjunto 0..N-1, o par (`conteudo[i]`, `prox[i]`) representa uma célula da lista. A célula seguinte é (`conteudo[j]`, `prox[j]`), sendo `j = prox[i]`. Escreva funções de busca, inserção e remoção para essa representação.

---

17) Escreva uma função que remova de uma lista duplamente encadeada a célula apontada por p. Que dados sua função recebe? Que coisa devolve?

---

18) Escreva uma função que insira uma nova célula com conteúdo x em uma lista duplamente encadeada logo após a célula apontada por p. Que dados sua função recebe? Que coisa devolve?

---

19) Escreva um programa completo que faça uma cópia byte-a-byte do arquivo cujo nome é digitado pelo usuário.

---

20) Escreva um programa que remova os comentários (do tipo `/*...*/` e do tipo `//...`) do arquivo-fonte de um programa C. O resultado deve ser gravado em um novo arquivo-fonte.
