MC102—Algoritmos e Programação de Computadores

Lista de exercícios 03

Turmas L, O, T e U — 2^o semestre de 2005

Os problemas indicado com uma estrela ($\star$) são mais complicados que os demais

1. Escreva uma função recursiva que exiba, em ordem crescente, os números do intervalo [1, n] na saída padrão.

2. Escreva uma função recursiva que exiba, em ordem decrescente, os números do intervalo [1, n] na saída padrão.

3. Escreva funções recursivas para manipulação de vetores de números inteiros:

   (a) Função que exibe, na saída padrão, os elementos cujos índices estão no intervalo [0, n-1]

   (b) Função que exibe, na saída padrão, os elementos cujos índices estão no intervalo [0, n-1], na ordem inversa

   (c) Função que retorna a soma dos elementos de um vetor.

   (d) Função que retorna o índice do menor elemento de um vetor.

   (e) Função que retorna o índice do maior elemento de um vetor.

   (f) Função que procura um valor em um vetor, não importando a ordem em que os elementos estão distribuídos. Deve retornar o índice do elemento ou -1 caso não o encontre.

   (g) Função que encontra um elemento de um vetor, sendo que seus elementos estão em ordem crescente (busca binária). Deve retornar o índice do elemento ou -1 caso não o encontre.

4. Escreva um procedimento recursivo que recebe uma string como parâmetro e a exibe invertida.

5. Escreva uma função recursiva que dado um parâmetro n, imprima todos os possíveis subconjuntos do conjunto $\{1, 2, \cdots, n\}$. Exemplo: Para $n = 4$, temos o conjunto $S = \{1, 2, 3, 4\}$, cujos subconjuntos são $\{\}$, $\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, $\{1, 2\}$, $\{1, 3\}$, $\{1, 4\}$, $\{2, 3\}$, $\{2, 4\}$, $\{3, 4\}$, $\{1, 2, 3\}$, $\{1, 2, 4\}$, $\{1, 3, 4\}$, $\{2, 3, 4\}$ e $\{1, 2, 3, 4\}$. Quantos subconjuntos devem ser impressos?

6. Altere a função do exercício anterior para que ela só escreva os subconjuntos de tamanho k (k passado como parâmetro para a função recursiva juntamente com n). Quantos subconjuntos devem ser impressos?

7. Escreva uma função que dado um vetor v, com n posições, escreva todas as permutações possíveis do vetor n. Quantas permutações devem ser impressas?

8. $\star$ **Problema do troco:** Embora pareça trivial, compor uma determinada soma em dinheiro utilizando o menor número de moedas nem sempre é uma tarefa simples. Para o nosso sistema monetário, com moedas de R$ 1,00, R$ 0,50, R$ 0,25, R$ 0,10, R$ 0,05 e R$ 0,01, basta utilizarmos primeiro a maior moeda (R$ 1,00) enquanto isso for possível, depois, pegar a quantia que falta e utilizarmos a segunda maior moeda, e assim por diante, até que todas as moedas tenham acabado. Isso resultará no menor número de moedas possível.

   Agora, imagine um sistema monetário baseado no dinheiro (D$),com moedas de D$ 8 (oito dinheiros), D$ 5 (cinco dinheiros) e D$ 1 (um dinheiro). Utilizando a técnica acima, dariamos o troco para D$ 10 com uma moeda de oito e duas de um, que utiliza uma moeda a mais do que se utilizassemos somente duas moedas de cinco dinheiros.

   A solução para esse problema é tentar todas as possíveis formas de dar troco e verificar qual é a que utiliza menos moeda. Escreva uma função recursiva que, dado um vetor com as moedas existentes em determinado sistema monetário e a quantia necessária, tente compor aquela quantia somente utilizando o menor número de moedas.

9. Considere o trecho de programa abaixo. Depois de executado, quais são os valores associados aos itens de (a) a (g). Suponha que os endereços das variáveis u e v são 1000 e 1004 respectivamente.

```
int v, u;
int pv, pu;
v = 45;
pv = &v;
*pv = v + 1;
u = *pv + 1;
pu = &u;
```

   (a) &v
   (b) pv
   (c) *pv
   (d) u
   (e) &u
   (f) pu
   (g) *pu

10. Considere o trecho de programa abaixo. Depois de executado, quais são os valores associados aos itens de (a) a (i). Suponha que os endereços das variáveis a, b e c são 1000, 1004 e 1008 respectivamente.

```
float a, b;
float c, *pa, *pb;
a = 0.001;
b = 0.003;
pa = &a;
*pa = 2 * a;
pb = &b;
c = 3 * (*pa + *pb);
```

(a) &a
(b) &b
(c) &c
(d) pa
(e) *pa
(f) &(*pa)
(g) pb
(h) &(*pb)
(i) c

11. Dado p um ponteiro qualquer, qual a diferença entre `p++`, `(*p)++` e `*(p++)` ?

12. Escreva duas funções, `le_dados_do_teclado()` e `escreve_dados_na_tela()`. A função `le_dados_do_teclado` lê um conjunto de n inteiros da entrada padrão e escreve em um arquivo texto. A função `escreve_dados_na_tela()` lê um conjunto de inteiros do arquivo e escreve na saída padrão. O número de inteiros lidos da entrada padrão deve ser informado pelo usuário, mas não deve ser gravado no arquivo. O arquivo deve ser gravado em modo texto

    Dica: Utilize as funções `fscanf`, `fprintf` e `feof`.

13. Altere o exercício anterior para que ele leia e escreva os inteiros em arquivos binários, ao invés de arquivos texto.

    Dica: Utilize as funções `fread`, `fwrite` e `feof`.

14. Escreva uma função que, dado um arquivo binário composto de vários registros do tipo `Registro`, de tamanho `sizeof(Registro)`, faça a leitura de todos os registros desse tipo para um vetor `v`, do tipo `Registro`, passado como paramêtro e que possui um tamanho grande o suficiente para que caibam todos os registros existentes naquele arquivo.

15. Escreva uma função que, dado um arquivo binário composto de vários registros do tipo `Registro`, de tamanho `sizeof(Registro)`, e um inteiro n passado como parâmetro, faça a leitura do n-ésimo registro existente no arquivo binário, ou retorne um registro vazio caso essa posição não exista no arquivo.

    Dica: Utilize as função `fseek`

16. Escreva uma função que leia n inteiros da entrada padrão e armazene-os em uma lista ligada simples de inteiros.

17. Escreva uma função que escreva todo o conteúdo de uma lista ligada na saída padrão.

18. Escreva um programa que ordene os elementos de uma lista ligada. Utilize o método de ordenação que você considerar mais conveniente.

    Dica: Quicksort não é uma boa escolha.

19. ⋆ **Pilhas:** Pilhas são listas de elementos onde o elemento removido é sempre aquele que foi inserido mais recentemente. Escreva duas funções que manipulam pilhas

    - `int insere_pilha(int n)`: insere o elemento n no topo da pilha, retornando 1, ou retorna 0 se a pilha estiver cheia.
    - `int remove_pilha()`, Remove o elemento do topo da pilha, ou retorna -1 se a pilha estiver vazia.

    Você deve fazer duas implementações: uma utilizando um vetor para armazenar os dados da pilha (utilize um inteiro para armazenar a posição do topo da pilha) e outra utilizando listas ligadas (utilize um ponteiro para armazenar o topo da pilha. Como seriam realizadas as operações de inserção e remoção?)

20. ⋆ **Lista ligada de vetores**. Embora pareça eficiente, alocar um elemento de memória para cada inteiro pode tornar um programa extremamente lento, devido a forma como o acesso a memória é feito internamente no computador.

    Uma solução é criar listas ligadas de vetores. Quando precisamos inserir um elemento, verificamos se o vetor do topo da lista ligada ainda possui espaço. Se possuir, inserimos o elemento nesse espaço, caso contrário, alocamos um novo nó na memória e inserimos o novo elemento no início do vetor alocado.

    A remoção utiliza um procedimento análogo. Para remover um elemento, removemos o elemento do vetor. Se ele ficar vazio, desalocamos o vetor e rearranjamos os ponteiros de forma a criar novamente a lista ligada. Marcamos as posições não utilizadas com -1, para diferenciar de posições que contenham valores reais.

    (a) Escreva duas funções de manipulação de listas ligadas: `insere_elemento`, que insere o elemento n na primeira posição vaga da lista ligada de vetores, e `remove_elemento`, que remove o primeiro elemento da lista ligada cujo valor seja igual a n, retornando 1, ou 0 caso esse elemento não exista na lista ligada.

    (b) Escreva a função `compacta_lista`, que pega uma lista ligada de vetores e remove todos os espaços não utilizados, substituindo por elementos válidos, e reduzindo o espaço total que a lista ocupa.

    (c) Escreva a função `ordena_lista`, que pega uma lista ligada de vetores previamente compactada e ordena os seus elementos. Utilize o algoritmo de ordenação que achar mais conveniente.

        Dica: Mergesort pode ser uma boa opção