MAC 2301 – Laboratório de Estrutura de Dados
Escola Politécnica – Primeiro Semestre de 20001
Primeira Prova – 8 de maio de 2001

Nome: ____

Assinatura: ____

Nº USP: ____________________ Turma: ____________________

Professor: ____

Instruções gerais

1. Esta prova é composta por cinco grupos de questões, sendo que cada questão vale dois pontos. Para o cômputo da nota serão consideradas no máximo uma questão de cada grupo.
2. Não destaque as folhas deste caderno.
3. Preencha o cabeçalho acima.
4. A prova pode ser feita a lápis. Cuidado com a legibilidade.
5. Verifique antes de começar a prova se o seu caderno de questões está completo.
6. Não é permitido o uso de folhas avulsas para rascunho.
7. Não é permitido a consulta a livros, apontamentos ou colegas.
8. Não é necessário apagar rascunhos no caderno de questões.
9. Dica: Como esta prova é curta, escolha a questão que lhe perecer mais fácil em cada grupo, não perca tempo resolvendo questões diferentes do mesmo grupo.

DURAÇÃO DA PROVA: 2 horas[1]

Questão	Valor	Nota
1	2,0	
2	2,0	
3	2,0	
4	2,0	
5	2,0	
Total	10	

[1] Se após duas horas mais da metade dos alunos ainda estiver na classe serão concedidos alguns minutos adicionais.

MAC-2301 — Estruturas de Dados

Primeira prova - 8/05/2001

1. Grupo 1 (Exercícios com pilha)

   (a) Uma cadeia de P pode ser especificada por ww^R, onde w é uma seqüência de a's e b's e w^R é o reverso de w, ou seja, w lido de trás para frente.

   $$P = \{\emptyset, aa, bb, aaaa, abba, bbbb, \ldots\}$$

   Faça um programa que, dada uma cadeia de caracteres x, determina se x pertence ou não a P, ou seja, se x é da forma ww^R.
   Exemplo: $abaaba \in P$ e $abbabba \notin P$.

   (b) Faça um programa que recebe uma sequência de parenteses, colchetes e chaves, e verifica se a sequências está balanceada.
   Exemplo: A sequência $([])\{\}$ está balanceada, a sequência $([)]$ não está.

   Para a resolução dos exercícios do grupo 1 você pode utilizar os seguintes protótipos (você não precisa escrever as funções, pode apenas usá-las):

```
/* Supõe-se o TipoPilha previamente definido */
void inicializaPilha(TipoPilha *p);
void empilha(TipoPilha *p, char ch);
void desempilha(TipoPilha *p, char *ch);
/* retorna 1 se a pilha esta vazia, 0 caso contrario */
int pilhaVazia(TipoPilha p);
/* retorna 1 e o caracter lido se a sequencia nao acabou,
   retorna 0 e o caracter '/0' caso contrario */
int leCaracter(char *ch);
```

2. Grupo 2 (Formas de alocação de pilhas em um vetor)

   - Como você faria para armazenar eficientemente duas pilhas em um vetor com MAX posições. Escreva em C as funções para a manipulação de uma das pilhas.

3. Grupo 3 (Listas ligadas)

   (a) Dado um apontador para um nó qualquer de uma lista ligada, insira um novo nó antes deste nó. Você deve utilizar o seguinte protótipo:
   `void insereAntes(apontNo p, int x);`.

   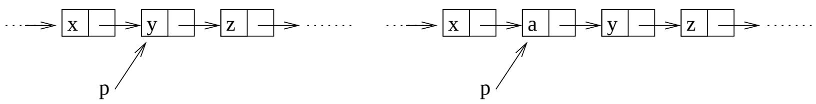

   Figura 1: Esquerda: Situação antes de `insereAntes(p,a);`. Direita: depois.

   (b) Dado um apontador para uma lista ligada de inteiros escreva uma função que elimina a ocorrência de números iguais na lista.

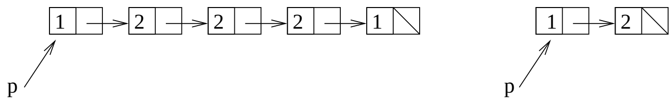

Figura 2: Esquerda: Situação inicial. Direita: Situação final.

Para a resolução dos exercícios do grupo 3 utilize a seguinte estrutura e protótipos para a lista ligada:

```
typedef struct _no {
           int info;
           struct _no *prox;
        } No, *apontNo;
void extraiNo(apontNo *p);
void devolveNo(apontNo p);
```

4. Grupo 4 (Manipulação de listas)

   (a) Dados dois apontadores para listas ligadas, escreva uma função que una estas listas em uma nova lista ligada. O que muda quando se utilizam listas circulares?

   (b) Dada uma lista circular duplamente ligada com cabeça de lista verifique eficientemente se a lista é palíndrome (o conteúdo do primeiro nó a esquerda do cabeça de lista é igual ao conteúdo do primeiro nó a direita, a assim por diante).

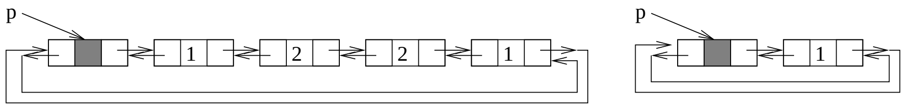

Figura 3: Exemplos de listas circulares palíndrome.

5. Grupo 5 (árvores)

   (a) Conhecendo os conceitos de árvores de busca, diga em que ordem, os elementos de 0 a 15 devem ser inseridos para gerar uma árvore de altura mínima. Explique.

   (b) Faça uma função que dado um apontador para a raiz de uma árvore conta a quantidade de nós folha (sem filhos) da mesma.

   Supõe-se que a árvore tem a seguinte estrutura:

```
typedef struct _no {
           int info;
           struct _no *esq, *dir;
        } no, *apontNo;
```