

Data: 14 de novembro de 2013**Horário limite:** 15:40

Resolva as questões abaixo identificando-as claramente na folha de respostas. Mantenha o silêncio na sala (mantendo desligado aparelhos eletrônicos). A interpretação das questões faz parte da prova.

(10 pontos) 1) O que é algoritmo?**(6 pontos)** 2) Diferencie Lista, Fila e Pilha.**(8 pontos)** 3) Mostre a situação da Pilha P. Inicialmente vazia, após a execução de cada uma das operações a seguir:

```

1 Empilha ( 'a', &P );
2 Empilha ( 'b', &P );
3 Empilha ( 'c', &P );
4 Empilha ( Topo (&P), &P );
5 Empilha ( Desempilha (&P), &P );
6 Desempilha (&P );
7 Empilha ( 'e', &P );
8 Desempilha (&P );

```

(20 pontos) 4) Suponha que queremos decidir se uma dada sequência de parênteses, colchetes e chaves estão bem-formadas (ou seja, parênteses, colchetes e chaves são fechados na ordem inversa àquela em que foram abertos). Por exemplo, a primeira e a segunda das sequências abaixo estão bem-formadas enquanto a terceira e a quarta não estão.**Primeira cadeia** - So when I die (the [first] I will see in (heaven) is a score list).**Segunda cadeia** - ([((([[]) () (())]))]).**Terceira cadeia** - Help(I[m being held prisoner in a fortune cookie factory]).**Quarta cadeia** - Half Moon tonight (At least it is better than no Moon at all].

Suponha que uma frase qualquer que possua sequência de letras, pontos, espaços, parênteses, colchetes e chaves está armazenada em uma cadeia de caracteres (string) s. Como é hábito em C, o último caractere da cadeia é o caractere nulo.

Escreva a função `int bem_formada (char *string)` que retorna 1 se a string está bem formada, senão retorna 0.**(16 pontos)** 5) Mostre a árvore de chamadas recursivas (pilha de recursão) e o resultado do algoritmo abaixo para a entrada inicial $X(4)$

```

1 int X( int n) {
2     if (n == 1 || n == 2) return n;
3     else return X( n-1) + n * X( n-2);
4 }

```

(40 pontos) 6) Jogando Cartas Fora

Dada uma pilha de n cartas enumeradas de 1 até n com a carta 1 no topo e a carta n na base. A seguinte operação é realizada enquanto tiver 2 ou mais cartas na pilha:

- Jogue fora a carta do topo e mova a próxima carta (a que ficou no topo) para a base da pilha.

Sua tarefa é encontrar a sequência de cartas descartadas e a última carta remanescente.

Cada linha de entrada (com exceção da última) contém um número $n \leq 50$. A última linha contém 0 e não deve ser processada. Cada número de entrada produz duas linhas de saída. A primeira linha apresenta a sequência de cartas descartadas e a segunda linha apresenta a carta remanescente.

Entrada

A entrada consiste em um número indeterminado de linhas contendo cada uma um valor de 1 até 50. A última linha contém o valor 0.

Saída

Para cada caso de teste, imprima duas linhas. A primeira linha apresenta a sequência de cartas descartadas, cada uma delas separadas por uma vírgula e um espaço. A segunda linha apresenta o número da carta que restou. Nenhuma linha tem espaços extras no início ou no final. Veja exemplo para conferir o formato esperado.

Exemplo de Entrada

7
19
10
6
0

Saída para o exemplo

Discarded cards: 1, 3, 5, 7, 4, 2
Remaining card: 6
Discarded cards: 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 4, 8, 12, 16, 2, 10, 18, 14
Remaining card: 6
Discarded cards: 1, 3, 5, 7, 9, 2, 6, 10, 8
Remaining card: 4
Discarded cards: 1, 3, 5, 2, 6
Remaining card: 4