Primeiro Trabalho de Oficina de Computação
Primeiro semestre de 2013

Atenção:

Este trabalho é obrigatório e deverá ser entregue, impreterivelmente, até a semana de 13 a 17 de maio de 2013. A solução é individual e deverá ser arquivada no diretório "~<professor>/CI067/" onde "~<professor>" deve ser "~alexd" (para a TURMA C2) ou "~nicolui" (para a TURMA A). O nome do arquivo terá como prefixo o seu nome-de-usuário no sistema do laboratório e, como extensão, ".c" para indicar que seu conteúdo possui um programa em C.

Assim, por exemplo, se você é aluno da Turma C2 e o seu nome de usuário no sistema é grs12 então o nome do arquivo será grs12.c (dentro do diretório "~alexd/CI067/"). Não se esqueça de proteger completamente o arquivo criado, de maneira a permitir a leitura do mesmo apenas por você! Isso deve ser feito aplicando `chmod og-rwx grs12.c` antes de efetuar a cópia com a preservação das permissões (`cp -p grs12.c ~alexd/CI067/`).

Não se preocupe com as permissões do professor que irá corrigir o trabalho. A correção dos trabalhos será parcialmente automatizada, sendo assim, é importante que todos os arquivos com as soluções individuais estejam no diretório citado acima, dentro do prazo estipulado. O procedimento de entrega acima deverá ser feito em sala de aula, na presença do professor, nos dias estipulados no calendário de avaliações de cada turma. Não será permitida a entrega do arquivo por e-mail.

O Puzzle das Fichas:

O Puzzle das Fichas é composto de uma régua segmentada (tabuleiro) com casas e fichas que podem ser movidas entre as casas. A Figura 1 mostra uma das possíveis configurações de estado onde há 2 fichas brancas e 2 fichas pretas em um tabuleiro de 5 casas.

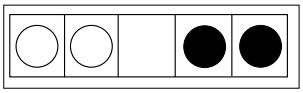

Figura 1: Um estado de configuração de um tabuleiro com 5 casas e 4 fichas.

As regras de composição e movimento são as seguintes:

1. A régua é constituída de n casas;
2. Há fichas brancas e pretas que, somadas, totalizam $n - 1$ fichas;
3. Em uma dada configuração, as fichas podem ocupar qualquer casa da régua, ficando sempre uma casa vazia;
4. É possível movimentar as fichas na régua de duas maneiras:
   - Fazer com que uma ficha deslize para a casa vazia, se a casa vazia for adjacente à ficha;
   - Fazer com que uma ficha salte para uma casa vazia, se o salto ocorrer por cima de uma única ficha.
5. Para resolver qualquer problema no Puzzle das Fichas, o objetivo é calcular e imprimir um único plano (mesmo que haja vários!) para transformar uma configuração inicial (dada por meio da entrada padrão) em uma configuração final (também dada como entrada) usando apenas as duas regras de movimento;
6. o plano não pode conter nenhuma repetição de configurações intermediárias entre os estados inicial e final das fichas;
7. O plano não pode conter nenhum movimento depois que todas as fichas esiverem no estado final.

A Figura 2 mostra todas as 4 configurações de estado diretamente atiníveis (adjacentes) a partir da aplicação das duas regras de movimento.

Fazer um programa em linguagem C para ler duas sequências de caracteres do teclado, as quais definem, respectivamente, as configurações dos estados inicial e final das fichas. Ambas as sequências terminam com o caractere "." (ponto, para indicar fim). Antes do "." cada sequência contém uma quantidade arbitrária de caracteres "b" e "p". Além desses, também deve aparecer obrigatoriamente um, e somente um, caractere espaço em branco antes do "." de terminação da leitura. Em total acordo com as regras acima, cada um dos caracteres lidos denota o seguinte:

"b": ficha branca;

"p": ficha preta;

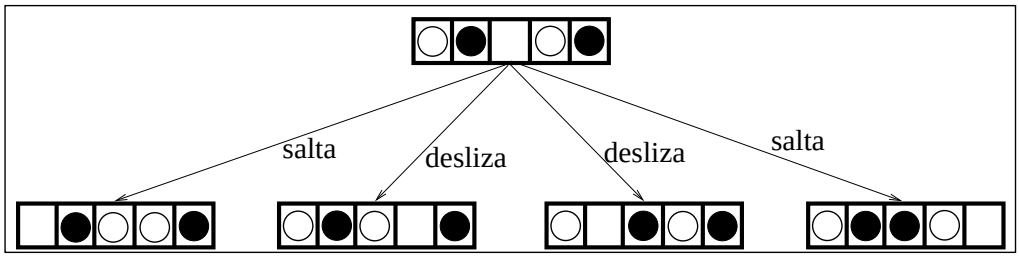

Figura 2: Estados edjacentes de um tabuleiro com 5 casas e 4 fichas.

“ ”: casa vazia (espaço em branco).

Ou seja, sem contar o “.”, a quantidade de caracteres lidos determina a quantidade de casas da régua. Depois da leitura, o programa deve gerar o plano e imprimí-lo na tela do computador. O formato de saída do plano deve ser dividido em linhas, onde cada linha apresenta o tipo de movimento (salta ou desliza) seguido da configuração adjacente à da linha anterior. Para ilustrar a execução do seu programa, veja o seguinte exemplo:

```
./grs12
Entre com o estado inicial: bb pp.
Entre com o estado final: bp pb.
desliza: _b bpp_
salta:   _bpb p_
desliza: _bp bp_
desliza: _b pbp_
desliza: _ bpbp_
salta:   _pb bp_
desliza: _p bbp_
salta:   _pbb p_
desliza: _pbbp _
salta:   _pb pb_
desliza: _p bpb_
desliza: _ pbpb_
salta:   _bp pb_
```

É importante você pesquisar sobre o conceito de retroação (`backtracking` em inglês), o qual pode ser implementado por meio de uma função recursiva. Atenção, pois a retroação é fundamentalmente diferente da retroalimentação. A título de ilustração, a Figura 3 mostra, para um exemplo diferente, o fragmento das ramificações que um processo retroativo sistemático de localização pode realizar para resolver automaticamente o problema de montagem de um plano usando as duas regras de movimento das fichas.

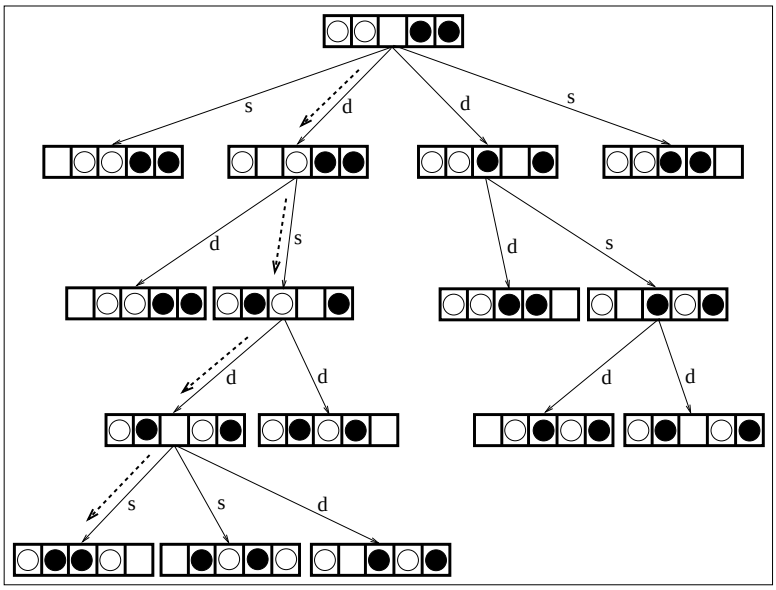

Figura 3: Fragmento das ramificações do planejamento de um problema.

Não esqueça de elaborar o seu código de forma modular e abstrata. Isso é importante. Bom trabalho!