Segundo Trabalho de Implementação

Algoritmos e Estrutura de Dados II

O objetivo deste trabalho é aplicar a técnica de *backtracking*, reduzindo o espaço de busca da solução de um problema. Dessa forma, um algoritmo mais eficiente do que uma simples 'força bruta' deve ser implementado.

Quebrador de Senhas

Considere que um sistema permite fazer indefinidas tentativas de login em sequência. Deve ser implementado um algoritmo para quebrar a senha desse sistema. As seguintes restrições sobre a senha foram implementadas no sistema:

1. Apenas letras minúsculas (códigos da tabela ASCII do 97 ao 122 em decimal) e números (códigos da tabela ASCII do 48 ao 57 em decimal) são permitidos
2. A senha deve conter exatamente 6 caracteres
3. Repetição de caracteres não é permitida
4. A senha deve conter pelo menos 2 letras
5. A senha deve conter pelo menos 2 números

Utilizando dessas informações, um algoritmo utilizando backtracking deve ser capaz de quebrar a senha de forma mais eficiente do que testar todas as permutações possíveis de letras minúsculas e números.

Detalhes de Implementação

Serão fornecidos arquivos de código fonte de dois tipos:

1. Arquivos de código fonte "esqueleto" que devem ser preenchidos com sua implementação,
2. Arquivos de header, exemplo ou auxiliares.

O arquivo *quebrador-senha.c* é o único arquivo que deve ser preenchido e faz parte da avaliação. Duas funções devem ser implementadas:

- *quebrador_senha_backtracking*: A implementação do algoritmo de quebrar senhas usando a técnica de backtracking.
- *quebrador_senha_exaustivo*: A implementação do algoritmo usando força bruta, onde uma busca exaustiva por todas as possibilidades deve ser efetuada. Contudo, essa busca levará em conta apenas as letras minúsculas e números. Outros caracteres **não** devem ser contados para a busca.

Funções auxiliares dentro do arquivo *quebrador-senha.c* podem ser criadas. Uma dica é criar funções auxiliares que implementarão de fato o algoritmo recursivo, tais funções podem receber parâmetros de entradas com variáveis/estruturas que facilitam a implementação.

Os cabeçalhos das funções, os arquivos de header (.h) e o arquivo biblioteca.c **não devem** ser modificados, pois essas funções serão usadas na bateria de testes que o programa será submetido.

O arquivo *teste.c* é um exemplo de bateria de testes que será executada. É altamente recomendado fazer seu programa funcionar para essa bateria de testes, mas sabendo que a bateria de teste que o programa será submetido pode ser trocada.

O trabalho deve ser feito de forma que possa ser compilado e executado nos computadores do laboratório do Departamento de Informática.

O que você deve entregar

Apenas um arquivo deve ser entregue:

- *quebrador_senha.c*

Esse arquivo deve ser compactado num arquivo .tar.gz (detalhes abaixo).

Forma de Entrega

O trabalho pode ser feito em grupos de até dois alunos.

Os arquivos devem ser empacotados em um arquivo grr1-grr2.tar.gz, onde grr1-grr2 é uma string com os GRR's dos integrantes da equipe. Ao descompactar este arquivo deverá ser criado um diretório de nome grr1-grr2 que conterá todos os demais arquivos.

Este arquivo deve ser enviado como anexo por e-mail ao endereço do professor com o assunto "CI056-trab2" (exatamente).

Data de Entrega

O trabalho pode ser entregue até às 23h59m da data estipulada no site da disciplina. Entregas feitas após esse prazo **não serão** avaliadas (recebem nota zero).

Critério de Avaliação

Os critérios de avaliação são os seguintes:

- O trabalho deve ser entregue no formato que foi especificado, assim como explicado na Seção "Forma de Entrega".
- O trabalho deve compilar e executar sem problemas, inclusive sendo compilado com Makefile e podendo ser testado com o arquivo teste.c.
- O trabalho está correto, ou seja, as implementações usando backtracking e busca exaustiva devem quebrar a senha definida e estão implementados de acordo com o especificado.
- Clareza do código.