EXERCÍCIO 1:
TESTE AUTOMÁTICO DE TRABALHOS

RAUL H.C. LOPES

1. Introdução

O objetivo deste exercício consiste em escrever uma conjunto programas em *bash* para compilar e testar um conjunto de trabalhos práticos. Assuma que cada trabalho está inicialmente arquivo como anexo de uma mensagem. O objetivo consiste em extrair o anexo, ativar a compilação e execução de testes e produzir relatório referentes à compilação e execução de tetses.

O ponto de partida do trabalho é um *mailbox* contendo mensagens, cujos anexos são trabalhos. Além disso, você recebe um conjunto de programas que realizam de forma semi-automática a tarefa de extrair os trabalhos das mensagens, compilar os trabalhos e gerar *log*s de compilação. Sua tarefa consiste em:

- entender os programas que geram *log*s compilação;
- escrever um conjuntos de programas que automatizem o processo de teste e produção de logs de execução;
- junto os programas de compilação e teste em um programa único;
- escrever um pequeno relatório, descrevendo seus programas e como usá-los.

O trabalhos a compilar e testar são trabalhos da disciplina de *Técnicas de Busca e Ordenação*, do curso de *Ciência da Computação* da **UFES**. Veja a especificação do trabalho.

2. Sua tarefa inicial

Inicialmente você tem à sua disposição um *tarball* contendo programas em *bash* para extrair os trabalhos de um *mailbox*, compilá-los e gerar *log*s de compilação. Suas tarefas:

- Entender esses programas.
- Alterá-los para que eles, o *mailbox* dos trabalhos a compilar, e o destino de *log*s e programas compilados possam estar em locais arbitrários, definidos pela secretária que irá usar seu programa.
- Escrever um programa que receberá três argumentos:

– A localização do *mailbox*;
– Identificação do trabalho a compilar.
– A localização dos programas e *log*s que serão gerados.

O nome deste programa será

start.compile

Ele acionará todos os passos do processo de compilação.

Note que a estrutura dos diretórios gerados pelo processo de compilacão deve ser mantida. Além disso, o *mailbox* que você receberá estará em um diretório com nome

submit

Dentro dele você encontrará diretórios cada um tendo como nome a identificação do trabalho a ser corrigido. Cada um destes diretórios conterá, por sua vez, três diretórios com nomes: **cur**, **new**, **tmp**.

O diretório de nome **new** conterá os trabalhos a corrigir. No diretório **new** existirá exatamente um arquivo, de nome arbitrário para cada trabalho a corrigir. Cada um desses arquivos é uma mensagem, contendo o trabalho em anexo.

3. A tarefa de testar os programas submetidos

O objetivo final desta seção consiste em escrever um programa em *bash*, de nome **runtests** que será executado com uma linha de comando como:

```
./runtest groups groups.test.logs groups.reports int
```

onde os parâmetros de **runtest** são:

- Primeiro parâmetro, no exemplo **groups**: diretório onde estão os programas compilados de cada grupo. Cada grupo tem seu próprio diretório e dentro deles existem quatro programas compilados: **tsp.frp, tsp.nn, tsp.greedy, tsp.pqgreedy.**
- Segundo parâmetro, no exemplo **groups.test**: diretório que conterá os resultados de testes de cada grupo. Novamente pode ser conveniente armazenar os resultados de cada grupo em um diretório separado e, dentro do diretório do grupo, os resultados, separados por diretório, de cada um dos programas acima. Veja um exemplo no *tarball*.
- Terceiro parâmetro, no exemplo, **groups.reports**: diretório com quatro arquivos de relatórios com nomes
  - **tsp.frp.tex**, contendo *log*s de execução para o programa **tsp.frp**.
  - **tsp.nn.tex**
  - **tsp.greedy.tex**
  - **tsp.pqgreedy.tex**

Cada um desses arquivos é uma tabela com linhas, contendo resultados de testes. Por exemplo a linha

```
tsp.frp & C1k.1 & 1 & ok & 43889849.26 & 0.11
```

Ela indica:

- Esta linha é resultado da execução do programa **tsp.frp**.
- A instância usada foi **C1k.1**.
- O grupo que escreveu o programa é identificado como **1**, estando os *log*s de execução no diretório
  groups.test.logs/1
- **ok** indica que o atour produzido está correto.
- **43889849.26** é o peso do *tour*.
- **0.11** é o tempo de execução.

Veja exemplo no *tarball*.

Os seguintes programas estão contidos no *tarball*:

- **tourlen**: dado um *tour*, avalia seu peso.
- **tourok**: dado um *tour*, avalia se ele está correto.
- **fixed.timed.bench.pl**: executa um programa de cálculo de um *tour* por um tempo determinado.

Leia arquivo **README** que detalha como executar tais programas.

Faça download do seguinte *tarball*. Ele contém um *mailbox* com uma mensagem, contendo uma submissão de solução para o trabalho prático da *Técnicas de Busca e Ordenação*. Use-o para depurar seus programas.

4. A avaliação

Este trabalho vale três pontos, sendo dois pontos para a seção 2 e um ponto para a seção 3.

5. A submissão

Submeta seus scripts com um *tarball* por e-mail. Sua mensagem terá a seguinte estrutura:

- **Subject:** linux.1
- Anexo: o *tarball* do arquivo.

Gere o *tarball*, usando o comando *tar*. Por exemplo, estando no diretório dos seus scripts, use:

```
tar -jcf ~/1.tar.bz2 *
```

Esse comando gerará em seu *home* o arquivo *1.tar.bz2* com os seus scripts.

Submeta até dia 6/04/2005.