Projecto

Laboratórios de Algoritmia I
Laboratórios de Informática II

2014/2015

Pretende-se criar uma aplicação na linguagem de programação C que corra no sistema operativo Linux (disponibilizado numa máquina virtual para quem precisar) que resolva o puzzle da Batalha Naval.

Definição do problema

Todos nós já jogámos à batalha naval. Cada jogador coloca vários barcos numa grelha e depois vai alternando para tentar descobrir onde o adversário colocou os seus barcos. No puzzle temos só um jogador que deve descobrir onde estão todos os barcos mediante dois tipos de informação:

- Sabendo o que está em certas posições da grelha: água, segmentos de barcos;
- Sabendo o nº de segmentos em cada linha ou coluna.

Leia o documento chamado *Battleships Solving Guide* para perceber melhor o que é um destes puzzles e como se resolve.

Formato do ficheiro

Para representar o puzzle em texto utilizaremos o seguinte formato:

- Uma linha com dois números separados por espaços que representam o nº de linhas e de colunas do puzzle;
- Duas linhas contendo números separados por espaços contendo a primeiro linha o nº de segmentos em cada linha do puzzle e a segunda o nº de segmentos em cada coluna;
- Para cada linha do puzzle utiliza-se uma string com os seguintes caracteres:
  - `.` Valor não determinado (i.e., vazio);
  - `~` Água;
  - `o` Marca de ocupado (mas sem saber que tipo de peça é);
  - `0` Submarino;
  - `<` Parte esquerda de um barco;

`>` Parte direita de um barco;

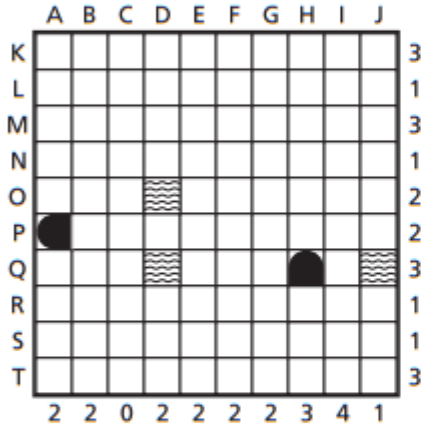
`#` Parte central de um barco;

`^` Parte de cima de um barco;

`v` Parte de baixo de um barco;

Assim, para representar o puzzle na figura utilizar-se-ia:

```
10 10
3 1 3 1 2 2 3 1 1 3
2 2 0 2 2 2 2 3 4 1
..........
..........
..........
..........
...~......
<.........
...~...^.~
..........
..........
..........
```



Tarefas

A aplicação deverá ler comandos do *standard input* que permitam realizar várias tarefas:

c Ler o tabuleiro a partir do *standard input*

m Mostrar o tabuleiro no *standard output*

l <ficheiro> Ler o tabuleiro a partir do ficheiro

e <ficheiro> Escrever o tabuleiro no ficheiro

h <num> Colocar o estado de todas as grelhas da linha nº num que ainda não estão determinadas como sendo água

v <num> Colocar o estado de todas as grelhas da coluna nº num que ainda não estão determinadas como sendo água

p<char> <l> <c> Colocar o caractere <char> na linha *l* e coluna *c*

E<num> Aplicar a estratégia nº `num`

V Verificar a solução

D Desfazer o último comando

R Resolver o puzzle

G <linhas> <colunas> Gerar um puzzle com um dado nº de linhas e colunas com solução única

q Sair do programa

Segue-se um exemplo de utilização da linguagem:

```
c
10 10
3 1 3 1 2 2 3 1 1 3
2 2 0 2 2 2 2 3 4 1
..........
..........
..........
..........
...~......
<.........
...~...^.~
..........
..........
..........
v 3
p> 6 2
h 6
p~ 5 1
p~ 5 2
p~ 7 1
p~ 7 2
p~ 7 7
p~ 7 9
p~ 8 7
po 8 8
p~ 8 9
po 2 9
po 3 9
po 4 9
h 2
h 4
p~ 3 8
p~ 3 10
m
```

Que deveria imprimir o seguinte:

```
..~....... 3
~~~~~~~o~ 1
..~....~o~ 3
~~~~~~~o~ 1
~~~~...... 2
<>~~~~~~~ 2
~~~~..~^~~ 3
..~...~o~. 1
..~....... 1
..~....... 3
2202222341
```

Análise do Código Gerado pelo Compilador

1. Vá buscar através do `git` o código que está no ficheiro `comp.c` na pasta da Análise de código;
2. Compile o código com o GCC versão 4.8.2 no sistema operativo Linux e seguidamente use o `gdb` e escreva o comando

   ```
   gdb executável
   ```

3. Vá buscar o código gerado pelo compilador para a função escrevendo o seguinte no prompt do gdb:

   ```
   disassemble contar_segs
   ```

4. Guardo o código que obteve num ficheiro de texto;
5. Crie a tabela de alocação de registos;
6. Corra o programa e coloque um breakpoint na função;
7. Identifique a área de memória associada à variável `tab`, descubra quanto espaço ocupa (e explique porquê) e faça um esquema da organização dessa área de memória;
8. Indique como é feita a indexação da matriz e mostre quais são as linhas do código *assembly* que lhe correspondem;
9. Identifique que instruções em *assembly* correspondem a cada instrução em C em particular no que diz respeito às estruturas de controlo (i.e., os ciclos);
10. Entregue um ficheiro com o resultado chamado `analise.pdf` contendo o resultado da sua análise (nomeadamente os pontos 4, 5, 7, 8 e 9) juntamente com o resto do trabalho colocando este ficheiro na raiz (ao mesmo nível do ficheiro `identificacao` e das pastas `code` e `doc`).

Calendarização e Entrega

Etapa	Data de Entrega
1ª etapa	05-04-2015
Defesa da 1ª etapa	06-04-2015 a 10-04-2015
2ª etapa	17-05-2015
Defesa da 2ª etapa	18-05-2015 a 22-05-2015
3ª etapa	31-05-2015
Defesa da 3ª etapa	01-06-2015 a 05-06-2015

A defesa de cada etapa é **presencial** e deverá ser feita por **todos** os elementos na semana correspondente à defesa dessa etapa e no turno prático correspondente. Se algum elemento ou grupo não defender, terá **zero** nessa etapa. A entrega de uma etapa poderá ser feita na etapa seguinte (isto só é válido para a 1ª e 2ª etapas) mas a avaliação levará uma penalização de 25%.

Eis o que deverá ser entreguem em cada etapa:

1. Os comandos **c**, *m*, **h**, *v*, **p** e **q**;
2. Os comandos **l**, **e**, **V**, **E1**, **E2**, **E3** e **D**;
3. Os comandos **G**, **R**, estratégias avançadas e análise do código gerado.

Para além disso também se avaliam os seguintes pontos na terceira etapa:

- Não ter avisos ou erros quando o código é compilado com as seguintes opções: `gcc -ansi -Wall -Wextra -pedantic -O2` do gcc;
- Legibilidade do código;
- Documentação do código;
- Relatório do projeto que explique as opções tomadas;
- Gestão do projeto.

Gestão de Projecto

Os grupos de trabalho são compostos por 3 elementos e terão necessáriamente de ser compostos por pessoas do mesmo turno prático. Nos casos em que o número de elementos no turno não seja divisível por 3 aceitam-se 2 grupos de 2 elementos se o resto da divisão do número por 3 for 1 e 1 grupo de 2 elementos se o resto der 2.

Pretende-se que os grupos de trabalho façam a gestão do projecto, utilizando para esse efeito o `Redmine`, evidenciando desse modo a sua capacidade de organização. Para o efeito será utilizada uma ferramenta de gestão de projectos que permitirá ao grupo planear o desenvolvimento do projecto, definindo subtarefas, fazer a sua atribuição aos elementos da equipa e acompanhar a sua implementação. Deverão também utilizar todas as restantes funcionalidades da ferramenta (e.g. documentação, wiki, etc.). A utilização correcta do sistema de gestão de projecto é obrigatório para a avaliação.

Não serão avaliadas as tarefas que não tenham sido correctamente introduzidas e contabilizadas no sistema.

Material a entregar em cada etapa

- Código fonte.
- Documentação gerada automáticamente pelo `Doxygen`;
- Relatório de desempenho do grupo na execução das diversas tarefas utilizando funcionalidades da ferramenta de gestão de projecto (descrição das tarefas incluindo tempo total dispendido e tempo por cada pessoa envolvida);

Critérios obrigatórios

Os seguintes critérios tem que ser cumpridos ou a entrega não é válida:

- O programa tem que compilar sem erros com as opções `-Wall -Wextra -pedantic -ansi -O2` e funcionar na máquina virtual disponibilizada;
- O programa tem que ler os comandos do `stdin`.

Entrega

Deverá ser colocado na opção "Files"do redmine ("Ficheiros"para os alunos que usam a versão portuguesa) um arquivo compactado com o comando `tar` do qual constem os seguintes ficheiros e pastas:

identificacao ficheiro com a identificação dos alunos (nome completo e número);

analise.pdf ficheiro com o relatório da análise do código gerado pelo compilador;

code pasta com o código fonte;

doc com a documentação `html`[1] gerada pelo `doxygen`.

A pasta deverá ter o nome PLg<nº do grupo>-et<nº da etapa>.tar.bz2
Nos casos em que o número do grupo seja só um algarismo este deverá ser precedido de um zero. Exemplos:

- PLg02-et1.tar.bz2 grupo 2 a entregar a etapa 1
- PLg11-et2.tar.bz2 grupo 11 a entregar a etapa 2

Para se usar o comando `tar` da forma correcta

1. Abre-se uma consola no Linux
2. Usando o comando cd vai-se para a directoria que contém as pastas code e doc
3. Escreve-se o comando `tar jcf PLg11-et2.tar.bz2 identificacao code doc`

Se não cumprirem alguns destes requisitos, o trabalho não será considerado entregue.

[1] i.e., deve conter dentro (e não em subpastas) o ficheiro index.htm ou index.html gerado pelo Doxygen assim como os restants ficheiros nnecessários