Universidade Federal do Espírito Santo – Departamento de Informática
Estruturas de Dados I (INF09292)
1º Trabalho Prático
Período: 2012/1
Profª Patrícia Dockhorn Costa
Email: pdcosta@inf.ufes.br

Data de Entrega: 14/05/2012 ***Grupos de 2 pessoas***

Este trabalho tem como objetivo praticar o uso de tipos abstratos de dados e estruturas do tipo Lista.

Regras Importantes
- Não é tolerado plágio. Trabalhos copiados serão penalizados com zero.
- A data de entrega é inadiável. Para cada dia de atraso, é retirado um ponto da nota do trabalho.

Material a entregar
- Impresso: Documentação do trabalho, que deve conter:
  - Introdução: descrição do problema a ser resolvido e visão geral sobre o funcionamento do programa (em termos de módulos, arquivos, etc.).
  - Implementação: descrição da implementação do programa. Devem ser detalhadas as estruturas de dados utilizadas (de preferência **com diagramas ilustrativos**), o funcionamento das principais funções utilizadas incluindo pré e pós condições, o formato de entrada e saída de dados, bem como decisões tomadas relativas aos casos e detalhes de especificação que porventura estejam omissos no enunciado. Modularize o seu programa como discutido em sala de aula.
  - Conclusão: comentários gerais sobre o trabalho e as principais dificuldades encontradas em sua implementação.
  - Bibliografia: bibliografia utilizada para o desenvolvimento do trabalho, incluindo sites da Internet se for o caso.

- Por email (pdcosta@inf.ufes.br):
  - O assunto da mensagem deve ser ed201201:trab1:<nome1>:<nome2>
    - Por exemplo: ed201201:trab1:<joaosilva>:<mariacosta>
  - Documentação do trabalho (em formato PDF).
  - Todos os arquivos .c e .h criados (exigido código muito bem documentado!).
  - O makefile.
  - Favor nomear os arquivos da seguinte maneira: **containers.h, containers.c, navios.h, navios.c, portos.h, portos.c e simulaportos.c.**

Sistema Portuário

Um sistema portuário tem a função de gerenciar as atividades realizadas em um porto. Neste porto, containers podem chegar por via terrestre ou via marítma. Quando chegam por via terrestre, eles são adicionados ao conjunto de containers do porto, que ficam disponíveis para serem carregados em navios. Containers chegam por via marítma em navios, a serem descarregados no porto. À medida que chegam, os navios entram em uma fila de navios e esperam para serem ***processados***. Processar um navio significa *carregá-lo*, caso o navio esteja ***vazio***, ou ***esvaziá-lo***, caso o navio tenha algum container. Navios são carregados com os containers disponíveis no conjunto de containers do porto (cada container é designado para um determinado navio específico). Os containers retirados de um navio são imediatamente excluídos do

porto. Assim que os navios forem processados, eles são desconsiderados pelo sistema portuário. A figura a seguir ilustra o funcionamento do porto.

Figura 1 - Figura ilustrativa do porto "vitoria"

No porto mostrado na Figura 1 (porto "vitória"), há 3 navios na fila esperando para serem processados, um conjunto de containers disponíveis para serem carregados em navios e alguns caminhões e trens carregando o porto com containers. Esse porto pode ser implementado com um conjunto de listas encadeadas: basicamente, implementa-se uma lista de Navios, na qual cada célula contém o nome do navio (identificador único), bem como uma lista de containers que este navio eventualmente estiver transportando. Uma outra lista pode ser utilizada para armazenar o conjunto de containers do porto, na qual cada célula representa um container, que deve especificar o identificador único do container, e o identificador do navio para o qual ele será carregado. Isso pode ser observado na seguinte Figura 2.

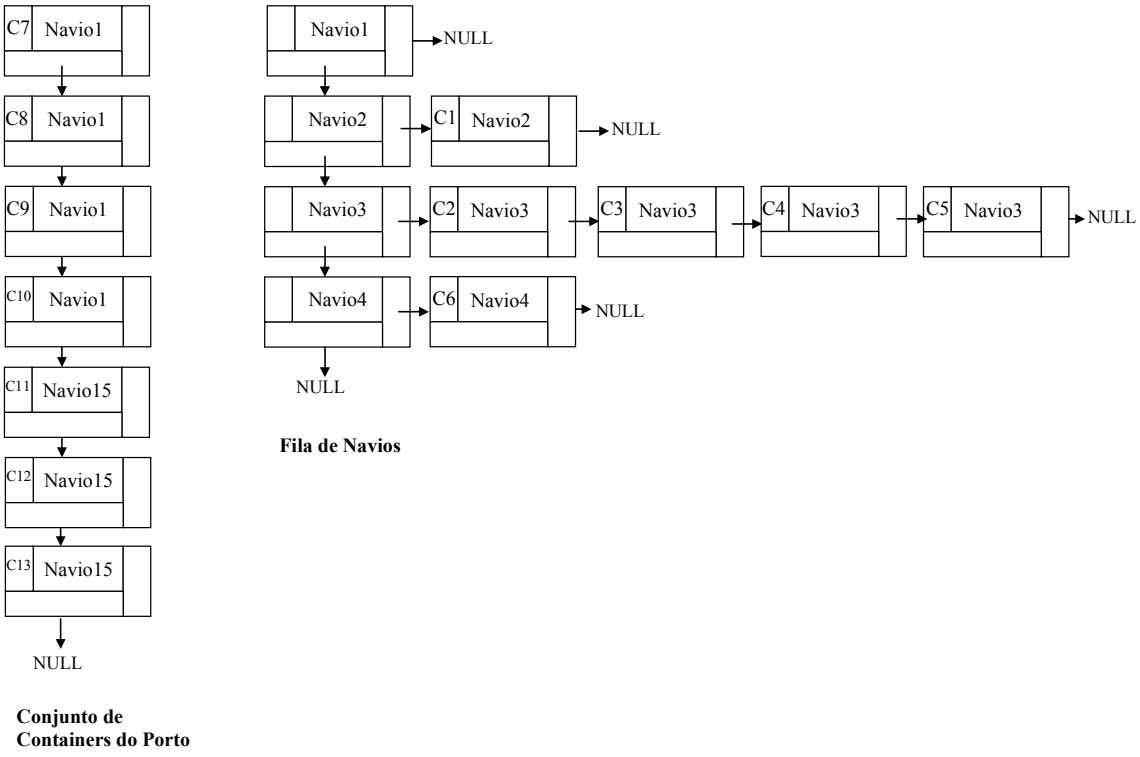
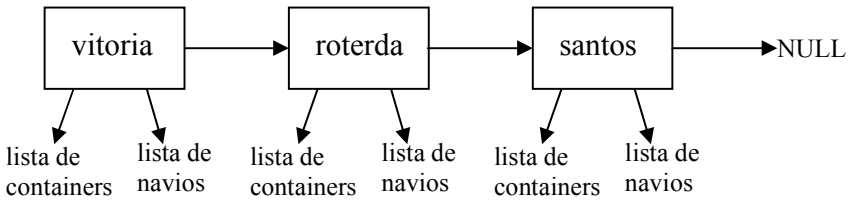


Figura 2 - Estruturas de dados do porto "vitória"

Como pode ser visto na Figura 2, na lista de containers há 4 containers (08, 09, 10 e 11) destinados para o navio 1 (identificador único "navio1"). Desta forma, quando o navio 1 for processado, estes containers serão carregados neste navio. Os outros containers se referem aos navios 15 e 16, que ainda não chegaram ao porto. Quando for a vez do navio 2, o container c2 será descarregado no navio (célula retirada) e este navio sairá da lista de navios, ou seja, será excluído da lista de navios do porto.

O sistema portuário a ser implementado pelo trabalho é formado por uma lista de portos, cada porto contendo uma lista de navios e uma lista de containers (como na figura Figura 2). A figura a seguir ilustra uma lista de portos que implementa o sistema portuário.



Nesse trabalho, você deverá implementar essa estrutura. No site da disciplina pode ser encontrada a especificação do TAD (portos.h), que define os tipos opacos e operações deste TAD. A implementação dos TADs de navios e containers é responsabilidade dos alunos.

O Programa Testador (simulaportos.c)

O programa testador deverá ser capaz de ler as instruções do arquivo texto de entrada e realizar as devidas operações no TAD portos. Para isto, o programa testador precisa manter uma lista de portos. O seu programa (**simulaportos**) deverá ler os dados de entrada a partir de um arquivo, cujo nome é passado como parâmetro na linha de comando (faz parte do trabalho descobrir como manipular arquivos e strings em C). Exemplo de execução do programa a partir da linha de comando:

```
simulaportos entrada.txt
```

O arquivo de entrada é basicamente uma lista de comandos (um por linha) em formato texto. O último comando é a palavra FIM, que indica o final do arquivo. O formato a ser usado é exemplificado abaixo:

Exemplo de arquivo de entrada

```
CRIAPORTO vitoria
CRIAPORTO roterda
CRIAPORTO santos
INCLUICONTAINERPORTO vitoria c1 navio1
INCLUICONTAINERPORTO vitoria c2 navio1
INCLUICONTAINERPORTO vitoria c3 navio1
INCLUICONTAINERPORTO vitoria c4 navio2
INCLUICONTAINERPORTO vitoria c5 navio3
INCLUICONTAINERPORTO vitoria c6 navio2
INCLUICONTAINERPORTO roterda c7 navio1
INCLUICONTAINERPORTO roterda c8 navio2
INCLUICONTAINERPORTO roterda c9 navio3
INCLUICONTAINERPORTO roterda c10 navio4
INCLUICONTAINERPORTO santos c11 navio4
```

```
INCLUICONTAINERPORTO santos c12 navio4
INCLUICONTAINERPORTO santos c13 navio5
INCLUICONTAINERPORTO santos c14 navio6
EXCLUICONTAINERPORTO santos c14
IMPRIMECONTAINERSPORTO santos
IMPRIMECONTAINERSPORTO vitoria
IMPRIMECONTAINERSPORTO roterda
CRIANAVIO navio1 vitoria
CRIANAVIO navio2 vitoria
CRIANAVIO navio3 roterda
CRIANAVIO navio4 santos
CRIANAVIO navio5 santos
CRIANAVIO navio6 santos
CRIANAVIO navio7 roterda
CRIANAVIO navio8 santos
CRIANAVIO navio9 vitoria
IMPRIMENAVIOSPORTO vitoria
IMPRIMENAVIOSPORTO santos
IMPRIMENAVIOSPORTO roterda
TRANSPORTA vitoria roterda
TRANSPORTA roterda vitoria
TRANSPORTA santos vitoria
TRANSPORTA vitoria santos
INCLUICONTAINERPORTO vitoria c15 navio8
INCLUICONTAINERPORTO santos c16 navio8
INCLUICONTAINERPORTO santos c17 navio8
TRANSPORTA vitoria santos
TRANSPORTA vitoria santos
TRANSPORTAPRIORITARIO vitoria santos
TRANSPORTA santos vitoria
TRANSPORTAPRIORITARIO roterda vitoria
TRANSPORTA santos roterda
IMPRIMEPORTO vitoria
IMPRIMEPORTO roterda
IMPRIMEPORTO santos
EXCLUIPORTO vitoria
EXCLUIPORTO santos
EXCLUIPORTO roterda
FIM
```

O seu programa deve checar consistência de dados do arquivo de entrada, como por exemplo: não incluir portos com os mesmos nomes, containeirs com os mesmos identificadores, etc. Para qualquer comando do arquivo de entrada, o teste de consistência deve ser realizado.

Os comandos de impressão de dados (IMPRIMECONTAINERSPORTO, IMPRIMENAVIOSPORTO e IMPRIMEPORTO), imprimem os dados com o formato específico.

Os comandos CRIAPORTO, INCLUICONTAINERPORTO, EXCLUICONTAINERPORTO, CRIANAVIO e EXCLUIPORTO possuem uma correspondência direta com as funções do tad portos. Portanto, para interpretá-los, vide documentação do arquivo portos.h.

O comando (TRANSPORTA porto1 porto2) faz com que seja processado o primeiro navio da fila de navios do porto1 e que este chegue no porto2 e entre na última posição da fila de navios.

O comando (TRANSPORTAPRIORITARIO porto1 porto2) faz com que seja processado o primeiro navio da fila de navios do porto1 e que este chegue no porto2 e entre na primeira posição da fila de navios.

Considerando o arquivo de entrada dado acima, espera-se o seguinte no arquivo de saída:

Arquivo de saída para o arquivo entrada.txt

```
Containers do Porto santos: C11(navio4), C12(navio4), C13(navio5)

Containers do Porto vitoria: C1(navio1), C2(navio1), C3(navio1),
c4(navio2), c5(navio3), c6(navio2)

Containers do Porto roterda: C7(navio1), C8(navio2), C9(navio3),
c10(navio4)

Navios do Porto vitoria
Navio: navio1
Containers:
Navio: navio2
Containers:
Navio: navio9
Containers:

Navios do Porto santos
Navio: navio4
Containers:
Navio: navio5
Containers:
Navio: navio6
Containers:
Navio: navio8
Containers:

Navios do Porto roterda
Navio: navio3
Containers:
Navio: navio7
Containers:

Navios do Porto vitoria
Navio: navio7
Containers:
Navio: navio4
Containers:

Containers do Porto vitoria: C5(navio3)

Navios do Porto roterda
Navio: navio1
```

```
Containers: c1, c2, c3
Navio: navio5
Containers: c13

Containers do Porto roterda: C7(navio1), c8(navio2), c10(navio4)

Navios do Porto santos
Navio: navio6
Containers:
Navio: navio8
Containers:
Navio: navio2
Containers: c4, c6
Navio: navio9
Containers:
Navio: navio3
Containers:

Containers do Porto santos: C16(navio8), c17(navio8)
```

Como mostrado no arquivo de saída exemplo, existe uma padronização a ser seguida pelas funções de impressão:

Imprime containers de um porto

O comando IMPRIMECONTAINERSPORTO, imprime os containers no seguinte formato:

"Containers do Porto " + nome-porto + ": " + id-container + "(" + id-navio + "), " + id-container + "(" + id-navio + ") ...

Por exemplo, o comando *IMPRIMECONTAINERSPORTO vitoria* gerou a seguinte saída:

```
Containers do Porto vitoria: C1(navio1), C2(navio1), C3(navio1), c4(navio2), c5(navio3), c6(navio2)
```

Imprime os navios de um porto

O comando IMPRIMENAVIOSPORTO, imprime os navios do porto no seguinte formato:

"Navios do Porto " + nome-porto + \n
"Navio: " + id-navio + \n
"Containers: " + id-container + ", " + id-container + … + \n
"Navio: " + id-navio + \n
"Containers: " + id-container + ", " + id-container + … + \n
…

Por exemplo, o comando *IMPRIMENAVIOSPORTO Roterda* gerou a seguinte saída:

```
Navios do Porto roterda
Navio: navio1
```

```
Containers: c1, c2, c3
Navio: navio5
Containers: c13
```

Imprime os navios e containers de um porto

O comando IMPRIMEPORTO, imprime os navios e os containers do porto, no seguinte formato:

"Navios do Porto " + nome-porto + \n
"Navio: " + id-navio + \n
"Containers: " + id-container + ", " + id-container + … + \n
"Navio: " + id-navio + \n
"Containers: " + id-container + ", " + id-container + … + \n
…
"Containers do Porto " + nome-porto + ": " + id-container + "(" + id-navio + "), " + id-container + "(" + id-navio + ") ...

Por exemplo, o comando IMPRIMEPORTO *Santos* gerou a seguinte saída:

```
Navios do Porto santos
Navio: navio6
Containers:
Navio: navio8
Containers:
Navio: navio2
Containers: c4, c6
Navio: navio9
Containers:
Navio: navio3
Containers:

Containers do Porto santos: C16(navio8), c17(navio8)
```

Mensagens de erro

Use a seguinte padronização para mensagens de erro:

"Erro: porto x já existente"
"Erro: navio x já existente no porto x"
"Erro: container x já existente no porto x"
"Erro: não foi encontrado porto x"
"Erro: não foi encontrado container x no porto x"
etc...

BOM TRABALHO!