INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

Projecto de Programação com Objectos
LEIC-A, LEIC-T, LERC, MEIC-A 2010/2011, 2º Ano, 1º Semestre

Gestor de uma Rede de Telemóveis

7 de Novembro de 2010

O projecto é obrigatório e contribui com um máximo de 8 valores (40%) para a nota final. Fraudes na execução do projecto terão como resultado a classificação de 0 (zero) valores e consequente eliminação do processo de avaliação em 2010/2011.

Datas importantes (todas as entregas até às 12:00): **2010/11/15** (entrega intermédia); **2010/12/02** (entrega final); **2010/12/13–2009/12/17** (teste prático).

Dúvidas que não possam ser esclarecidas presencialmente ou através do material no Fénix, devem ser colocadas através de correio electrónico (ver endereço no Fénix). Advertem-se os alunos contra a utilização de fontes de informação não oficialmente associadas ao corpo docente.

O objectivo do projecto é desenvolver um gestor de uma rede de telemóveis, denominado por "*grt*". O programa permite gerir vários serviços: registo de clientes e de telemóveis e controlo das comunicações efectuadas.

O grt disponibiliza aos seus gestores vários serviços, entre outros: (i) registar dados sobre os clientes; (ii) registar dados sobre os telemóveis pertencentes à rede; (iii) registar dados sobre as chamadas efectuadas; (iv) fazer pesquisas sobre as chamadas já efectuadas; (v) contabilizar o saldo associado a telemóveis.

1 Clientes, telemóveis, planos tarifários e chamadas

Os utilizadores do grt podem ter vários telemóveis, cada um deles associado a um número de telefone e um plano tarifário.

Nos pontos seguintes, descrevem-se as propriedades de cada entidade da aplicação.

1.1 Propriedades

Os clientes, telemóveis e chamadas possuem uma chave única, uma cadeia de caracteres para os clientes e um inteiro para os telemóveis e para as chamadas.

Cada cliente, para além da chave única, é identificado pelo nome (cadeia de caracteres) e número de identificação fiscal (inteiro).

Os telemóveis são identificados por um número (exactamente 6 dígitos) e estão associados a um cliente. Cada telemóvel pode estar ligado, no silêncio, ou desligado. Neste último caso, não é permitido iniciar ou receber chamadas. Quando o telefone está no silêncio pode iniciar qualquer tipo de chamada, mas só pode receber chamadas de texto. Cada telefone tem uma contabilidade própria, sendo sempre possível saber o saldo que lhe está associado.

Cada chamada é identificada por um inteiro, único em toda a rede, e possui ainda informação sobre o telemóvel que originou a chamada, o telemóvel de destino, a duração da chamada e o tipo de comunicação (voz, texto ou vídeo). A primeira chamada tem como identificador "1", devendo o identificador das chamadas subsequentes ser obtido por incremento unitário do identificador anterior.

Existem 3 tipos de comunicação entre telemóveis: texto (denominada SMS), voz (denominada VOZ) e vídeo (denominada MMS). O custo das chamadas não é uniforme (ver §1.4), dependendo do tipo de telefone e do cliente, entre outros factores. Também dependendo do cliente, é possível avisar quem o tentou contactar quando: (i) um telefone desligado é colocado no silêncio; (ii) um telefone desligado é ligado; ou (iii) um telefone no silêncio é ligado.

A aplicação desenvolvida deve ser suficientemente **flexível**, devendo permitir modelar: novos tipos de chamada, novos planos tarifários e novas políticas de avisos. Também deve permitir a gestão de várias redes de telemóvel. O impacto na aplicação deve ser mínimo.

1.2 Gestão de Clientes

O operador do grt pode realizar as seguintes operações, relativamente a clientes: (i) visualizar um cliente; (ii) registar um novo cliente; (iii) permitir o registo de contactos falhados; (iv) inibir o registo de contactos falhados; (v) calcular o saldo de um cliente.

Existem três tipos de clientes: Normal (situação inicial, após o registo), Ouro e Platina. O custo das chamadas está indexado ao tipo de cliente (ver §1.4). Um cliente transita de tipo nas seguintes condições:

Normal → Ouro	o saldo (após realizar um pagamento) é superior a 500 cêntimos
Normal → Platina	—
Ouro → Normal	o saldo (após realizar uma chamada) é inferior a 0 cêntimos
Ouro → Platina	após realizar 5 chamadas consecutivas do tipo MMS (a contabilização da 5ª chamada ainda considera que o cliente é do tipo Ouro)
Platina → Ouro	após realizar 2 chamadas consecutivas do tipo SMS (a contabilização da 2ª chamada ainda considera que o cliente é do tipo Platina)
Platina → Normal	tentativa de realizar uma chamada MMS para um telefone 2G

Um cliente não pode ser removido, sendo sempre possível aceder a todo o seu historial.

1.3 Gestão de Telemóveis

O operador do grt pode realizar as seguintes operações, relativamente a telemóveis: (i) visualizar um telemóvel; (ii) registar um novo telemóvel; (iii) ligar, desligar e colocar no silêncio um telemóvel; (iv) adicionar e remover telefones da lista dos telefones amigos de um telemóvel; (v) proceder a um pagamento; (vi) calcular o saldo de um telemóvel.

Os telefones 2G só conseguem realizar chamadas SMS e de VOZ, não podendo nem iniciar nem receber chamadas MMS. Quando um telefone está no silêncio, não pode receber chamadas de VOZ ou MMS.

Para promover as comunicações, quando uma chamada não se efectua por o telefone chamado estar no silêncio ou desligado, regista-se a tentativa de contacto, para que, assim que seja possível a realização do contacto pretendido, se enviarem mensagens aos telefones chamadores. O registo da tentativa de contactos só tem lugar quando o cliente do telefone chamado tem activa a recepção de contactos falhados no instante em que se tentou efectuar a chamada (que não teve lugar por o telefone chamado o não permitir).

Um telemóvel não pode ser removido, sendo sempre possível aceder ao seu historial.

1.4 Planos tarifários

Os custos de uma chamada dependem do tipo de utilizador que inicia a chamada e do tipo da chamada: SMS, VOZ ou MMS.

Quando é efectuada uma chamada do tipo SMS, o custo ou é fixo ou depende do número (representado na tabela por N) de caracteres enviados:

	Cliente Normal	Cliente Ouro	Cliente Platina
N <50 caracteres	10 cêntimos	10 cêntimos	0 cêntimos
50 caracteres $\leq N$ <100 caracteres	16 cêntimos	10 cêntimos	4 cêntimos
$N \geq$100 caracteres	$2 * N$ cêntimos	$2 * N$ cêntimos	4 cêntimos

Quando é efectuada uma chamada do tipo VOZ ou MMS, o custo é proporcional ao tempo de conversação e quando se liga para um telefone registado como amigo é aplicado um desconto de 50%. O custo, em cêntimos por minuto, é o seguinte para telefones não registados como amigos:

	Cliente Normal	Cliente Ouro	Cliente Platina
VOZ	20 cêntimos	10 cêntimos	10 cêntimos
MMS	30 cêntimos	20 cêntimos	10 cêntimos

Por imposição das finanças, todos os cálculos envolvendo os custos das chamadas devem ser apresentados em cêntimos, não sendo possível fazer arredondamentos. Por exemplo, uma chamada de voz que dure 6 minutos tem um custo de 120 cêntimos para um cliente normal e 60 cêntimos para um cliente ouro ou platina.

1.5 Pesquisas

Deve ser possível efectuar pesquisas sob vários critérios e sobre as diferentes entidades geridas pelo grt: (i) chamadas efectuadas por um cliente; (ii) telemóveis que iniciaram chamadas para um determinado telemóvel; (iii) telemóveis que iniciaram uma chamada para um determinado cliente; (iv) telemóveis com um determinado plano tarifário; (v) clientes com pagamentos em atraso.

Deve ser possível introduzir novos métodos de pesquisa com um impacto mínimo na implementação desenvolvida.

2 Interacção com o Utilizador

Nesta secção é descrita a **funcionalidade máxima do gestor de uma rede de telemóveis**, descrevendo-se a interacção com o utilizador. No texto que se segue, o tipo de letra `fixo` indica um literal; o símbolo ␣ indica um espaço; e o tipo de letra *itálico* indica uma parte variável.

As excepções documentadas nesta secção são tratadas pela classe `pt.utl.ist.po.ui.Menu`. Todas as interacções com o utilizador (leitura e escrita) **têm** de ser realizadas através do objecto `UserInteraction.IO`, utilizando-se as mensagens descritas abaixo (todas as mensagens são produzidas através de chamadas a métodos). Uma lista completa das classes disponibilizadas pode ser obtida nas bibliotecas **po-uilib** e **po-prj10-strings** associadas a este enunciado.

A apresentação de listas (e.g., Clientes, Telemóveis, Chamadas, etc.) faz-se por ordem crescente da respectiva chave. Quando a chave é uma cadeia de caracteres, a ordem deve ser lexicográfica (UTF-8), não havendo distinção entre maiúsculas e minúsculas.

2.1 Menu Principal

As entradas do menu estão definidas em `grt.textui.main.MenuEntry`. Os métodos correspondentes às mensagens de diálogo estão definidos em `grt.textui.main.Message`.

Este menu permite realizar operações de leitura e escrita básicas e aceder a submenus que contêm a restante funcionalidade. A lista completa é a seguinte (omite-se a opção **Sair**, uma vez que é implementada automaticamente por todos os menus): **Criar**, **Abrir**, **Guardar**, **Guardar como...** e **Menu de Operação** (ver §2.2). As secções abaixo descrevem pormenorizadamente as acções associadas a estas opções.

2.1.1 Manipulação de Ficheiros

São definidas as operações básicas para manipulação de ficheiros: criação de nova informação, abertura de ficheiro com informação existente e salvaguarda da informação. Devem ser tratadas todas as excepções relativas à manipulação de ficheiros. É sempre possível trabalhar com o gestor do operador de telemóveis: se não for pré-carregada informação (através da linha de comando, por exemplo), quando o sistema inicia, existe automaticamente um operador vazio e anónimo. A funcionalidade de cada operação é a seguinte:

Criar – Criação de nova informação do gestor de um operador de telemóveis.

Abrir – Abertura e eventual utilização de um ficheiro existente (previamente guardado pelo gestor). O sistema pede o nome do ficheiro a abrir: caso não exista, o sistema limita-se a comunicar o erro (mensagem `fileNotFound()`).

Guardar – Salvaguarda das alterações desde a abertura do ficheiro associado ao gestor do operador de telemóveis. Caso não haja nenhum ficheiro associado, deve ser perguntado ao utilizador o nome do ficheiro a utilizar. Esta interacção realiza-se através do método `newSaveAs()`. Não é executada nenhuma acção se não existirem alterações desde a última salvaguarda.

Guardar como... – Esta opção permite associar/mudar a associação de um ficheiro aos dados do gestor (além de guardar os dados nesse ficheiro). Esta interacção utiliza o método `saveAs()`.

Apenas é possível trabalhar com um ficheiro de cada vez. Assim, sempre que se abandona um ficheiro com modificações não salvaguardadas, por exemplo, porque se cria outro, deve ser perguntado ao utilizador se deseja guardar a informação actual:

1. Se houver alterações, então deve-se perguntar ao utilizador se deseja guardar o ficheiro antes de sair: esta operação utiliza a mensagem `saveBeforeExit()` (a resposta é obtida invocando `readBoolean()`). Caso o utilizador responda de forma afirmativa, então deve-se guardar a informação relativa ao operador de telemóveis.
2. Se não existir nenhum ficheiro associado, então deve-se ainda perguntar ao utilizador qual o nome a utilizar para guardar a informação, através da mensagem `newSaveAs()`.

Note-se que este procedimento corresponde à opção Guardar.

2.2 Menu de Operação

As entradas do menu estão definidas em `grt.textui.grt.MenuEntry`. Os métodos correspondentes às mensagens de diálogo estão definidos em `grt.textui.grt.Message`.

O **Menu de Operação** permite aceder aos menus relativos a aspectos específicos da informação do gestor de uma rede de telemóveis: **Gestão de Clientes**, **Gestão de Telemóveis**, **Consultas** e **Ver Saldo**. As secções abaixo descrevem pormenorizadamente as acções associadas a estas opções.

2.3 Menu de Gestão de Clientes

As entradas do menu estão definidas em `grt.textui.cliente.MenuEntry`. Os métodos correspondentes às mensagens de diálogo estão definidos em `grt.textui.cliente.Message`.

Sempre que o sistema pedir a cadeia de caracteres correspondente ao identificador de um cliente (`clientKeyReq()`) (cadeia de caracteres) e o identificador não existir, deve ser lançada a excepção `grt.textui.client.UnknownClientKeyException` (excepto no processo de registo).

2.3.1 Visualizar [visualizar todos os clientes]

O formato de apresentação de cada cliente é o seguinte, em que *númeroTelefones* é o número de telemóveis activos associados ao cliente e *saldo* é o somatório do saldo de cada um desses telemóveis (um *long* representando os cêntimos):

`CLIENTE` | *idCliente* | *nomeCliente* | *NIF* | *tipoCliente* | *estadoAtendedor* | *númeroTelefones* | *saldo*

Os valores para o campo *tipoCliente* são `NORMAL` (`clientNormal()`), `OURO` (`clientGold()`) e `PLATINA` (`clientPlatine()`), e os valores para o campo *estadoAtendedor* são `ACTIVO` (`atendedorActivo()`) e `INACTIVO` (`atendedorInactivo()`). Se um cliente ainda não tiver telemóvel, a impressão termina em: `|0|0` (i.e., número de telefones é 0 (zero) e o saldo também).

2.3.2 Registar [registar novo cliente]

O sistema pede o identificador que ficará associado ao cliente (se já existir um cliente com o mesmo identificador, deve ser apresentada a mensagem `duplicateClient()`, não se realizando qualquer acção). De seguida, pede o nome do cliente (`clientNameReq()`) e o número de identificação fiscal (`clientNifReq()`). Após o registo, o cliente fica no estado Normal e com o registo de contactos falhados activo.

2.3.3 Activar [activar recepção de contactos falhados]

O sistema pede o identificador do cliente. Se o registo de contactos falhados já estava activo, o cliente não é alterado e é enviada a mensagem `alreadyMessageActive()`.

2.3.4 Desactivar [desactivar recepção de contactos falhados]

O sistema pede o identificador do cliente. Se o registo de contactos falhados já estava inactivo, o cliente não é alterado e é enviada a mensagem `alreadyMessageInactive()`.

2.3.5 Calcular saldo [calcular o saldo referente a todos os telemóveis de um cliente]

O sistema pede o identificador do cliente, devendo ser apresentado de imediato o seu saldo (um *long* representando os cêntimos correspondentes ao somatório do saldo de todos os telemóveis desse cliente) para com o operador de telemóveis.

2.4 Menu de Gestão de Telemóveis

As entradas do menu estão definidas em `grt.textui.mobile.MenuEntry`. Os métodos correspondentes às mensagens de diálogo estão definidos em `grt.textui.mobile.Message` (excepto no processo de registo).

Sempre que for pedido o identificador de um telemóvel (`numeroReq()`) e o identificador não existir, deve ser lançada a excepção `grt.textui.mobile.UnknownKeyException`.

2.4.1 Visualizar [visualizar todos os telemóveis]

O formato de apresentação é o seguinte:

`TELEMOVEL` | *número* | *cliente* | *tipoTelemóvel* | *estado* | *saldo* | *telemóveisAmigos*

Os valores para o campo *tipoTelemóvel* são `2G` (`mobile2G()`) e `3G` (`mobile3G()`). Os valores para o campo *estado* são `LIGADO` (`mobileOn()`), `SILENCIO` (`mobileSilence()`) e `DESLIGADO` (`mobileOff()`). Os valores para o campo *telemóveisAmigos* são os identificadores dos telemóveis registados como amigos (números), separados por vírgulas (“,”). Os identificadores de *telemóveisAmigos* devem ser apresentados por ordem crescente dos números desses telemóveis.

2.4.2 Registar [registar novo telemóvel]

O sistema pede o número identificador do telemóvel. Se o número introduzido não contiver exactamente 6 dígitos, o sistema volta a pedir o número identificador. Se já existir um telemóvel com o mesmo número identificador, deve ser apresentada a mensagem `duplicateMobile()`, não se realizando qualquer acção.

De seguida, o sistema pede o tipo de telemóvel (`typeReq()`). A resposta deve ser `2G` (i.e., `mobile2G()`) (é registado um telemóvel da geração 2G), ou `3G` (i.e., `mobile3G()`) (é registado um telemóvel da geração 3G). Se a resposta não corresponder a nenhum dos dois valores, a pergunta `typeReq()` é repetida até se obter uma resposta válida. Finalmente, pede o identificador do cliente a que ficará associado. Se não existir um cliente com o identificador introduzido, deve ser lançada a excepção adequada (ver §2.3) e não se regista o telemóvel.

Quando um telemóvel é registado, fica no estado ligado, com saldo zero e com uma lista vazia de telemóveis amigos.

2.4.3 Gestão do telemóvel NNNNNN [entrar no menu de gestão do telemóvel com o número NNNNNN]

O sistema pede o número identificador do telemóvel antes de mostrar o menu correspondente a esse telemóvel (ver §2.7).

2.5 Menu de Consultas

As entradas do menu estão definidas em `grt.textui.lookup.MenuEntry`.

As mensagens estão definidas em `grt.textui.lookup.Message`.

Sempre que for feita uma consulta e nenhuma entidade satisfazer as condições associadas ao pedido, nada deve ser impresso.

2.5.1 Consultar todas as chamadas [informação relativa a chamadas]

O formato de apresentação é o seguinte:

`CHAMADA` | *idChamada* | *telChamador* | *telChamado* | *tipo* | *duração* | *custo* | *estado*

Os possíveis valores para o campo *tipo* são: `VOZ` (`voiceMessage()`), `TEXTO` (`textMessage()`), `MMS` (`videoMessage()`).

Os possíveis valores para o campo *estado* são:

- `EFECTIVA` (`done()`), a chamada teve lugar;
- `SEM-MENSAGEM` (`noMessage()`), a chamada não teve lugar e não ficou registada no chamador a tentativa de comunicação;
- `COM-MENSAGEM` (`message()`), a chamada não teve lugar mas ficou registada no chamador a tentativa de comunicação.

Quando o tipo de mensagem é `SEM-MENSAGEM` ou `COM-MENSAGEM`, tanto a *duração* como o *custo* são sempre 0 (zero).

2.5.2 Consultar chamadas de um cliente [informação relativa a chamadas]

O sistema pede o identificador do cliente (§2.3), apresentando as chamadas (§2.5.1) originadas por esse cliente.

2.5.3 Consultar chamadas para um cliente [informação relativa a chamadas]

O sistema pede o identificador do cliente (§2.3), apresentando as chamadas (§2.5.1) recebidas por esse cliente.

2.5.4 Consultar clientes sem dívidas [informação relativa a clientes]

O sistema apresenta os clientes (§2.3.1) sem dívidas.

2.5.5 Consultar clientes com dívidas [informação relativa a clientes]

O sistema apresenta os clientes (§2.3.1), ordenados por ordem decrescente das dívidas, e, caso tenham a mesma dívida, por ordem crescente do seu identificador. Um cliente tem dívidas se tem pelo menos um telefone com saldo negativo, sendo a sua dívida total a soma de todos os saldos negativos.

2.5.6 Consultar telemóveis sem actividade [informação relativa a telemóveis]

O sistema apresenta todos os telemóveis (ver §2.4.1) que ainda não efectuaram nem receberam qualquer chamada.

2.5.7 Consultar telemóveis com saldo positivo [informação relativa a telemóveis]

O sistema apresenta todos os telemóveis (ver §2.4.1) que têm saldo positivo, ou seja, superior a zero cêntimos.

2.6 Ver Saldo Global

O sistema apresenta, de imediato, o somatório dos saldos de todos os clientes registados no gestor da rede de telemóveis (ver §2.3.5). O valor apresentado deve ser um *long* representando cêntimos.

2.7 Menu de Gestão do Telemóvel NNNNNN

As entradas do menu estão definidas em `grt.textui.oneMobile.MenuEntry`, sendo o título deste menu obtido pela concatenação da cadeia de caracteres `grt.textui.oneMobile.MenuEntry.TITLE` com o número do telemóvel a gerir. Os métodos correspondentes às mensagens de diálogo estão definidos em `grt.textui.oneMobile.Message`.

2.7.1 Ligar telemóvel [ligar o telemóvel]

Se o telefone indicado já estiver ligado, envia a mensagem `alreadyOn()`. Se for necessário enviar mensagens aos telefones que tentaram o contacto enquanto o telefone não permaneceu ligado, usa-se a mensagem `isAvailable(caller)`. Estas mensagens devem ser ordenadas pelos números dos telefones que tentaram o contacto e só deve ser enviada uma mensagem a cada telefone que tentou o contacto, mesmo que tenha havido mais do que uma tentativa de contacto.

2.7.2 Colocar telemóvel no silêncio [Coloca o telemóvel no silêncio]

Se o telefone indicado já estiver no silêncio, envia a mensagem `alreadySilent()`. Se estava anteriormente desligado e se for necessário enviar mensagens aos telefones que tentaram o contacto por SMS enquanto o telefone permaneceu desligado, usa-se a mensagem `isAvailableForSMS(caller)`. Estas mensagens devem ser ordenadas pelos números dos telefones que tentaram o contacto e só deve ser enviada uma mensagem a cada telefone que tentou o contacto, mesmo que tenha havido mais do que uma tentativa de contacto.

2.7.3 Desligar telemóvel [Desligar o telemóvel]

Se o telefone indicado já estiver desligado, envia a mensagem `alreadyOff()`.

2.7.4 Adicionar amigo [Adicionar telemóvel à lista dos amigos]

O sistema pede o identificador do telefone a adicionar à lista dos amigos. Se o telefone indicado já fizer parte da lista dos telefones amigos, não faz nada.

2.7.5 Retirar amigo [Retirar telemóvel da lista dos amigos]

O sistema pede o identificador do telefone a retirar da lista dos telefones amigos. Se o telefone indicado não fizer parte da lista dos telefones amigos, não faz nada.

2.7.6 Pagamento [proceder a um pagamento]

O sistema pede o valor a pagar através da mensagem `paymentValue()`. O valor indicado é em cêntimos (sempre um inteiro).

2.7.7 Consultar saldo [consultar o saldo de um telemóvel]

O sistema deve apresentar de imediato o saldo, em cêntimos, associado ao telefone.

2.7.8 Estabelecer ligação [efectuar uma chamada]

Se o telefone a partir do qual se pretende efectuar a ligação estiver desligado apresenta a mensagem `thisMobileIsOff()` e não faz mais nada.

O sistema pede o identificador do número de telefone a contactar e se o número pretendido não existir, apresenta a mensagem `noMobile()` (neste caso, excepcionalmente, não é lançada nenhuma excepção). Se o número existir, pede o tipo de chamada pretendido (`typeReq()`) (a resposta deve ser a cadeia de caracteres: `VOZ` (`voiceMessage()`), `SMS` (`textMessage()`) ou `MMS` (`videoMessage()`)). Se a resposta não corresponder a nenhum dos três valores, a pergunta `typeReq()` é repetida até se obter uma resposta válida.

Sempre que uma chamada MMS é feita de um telemóvel 2G, a excepção `grt.textui.oneMobile.MMSfrom2GException` é lançada. Se o telefone de origem for 3G mas tentar realizar uma chamada MMS para um telemóvel 2G, é lançada a excepção `grt.textui.oneMobile.MMSto2GException`. Quando é lançada uma excepção, não se regista a tentativa de comunicação (o contador de chamadas não deve ser incrementado) e não são feitas mais perguntas.

Quando o telefone de destino está desligado, o grt imprime a mensagem `mobileIsOff()`, quando está no silêncio e a chamada pretendida é de VOZ ou MMS, imprime a mensagem `mobileIsSilent()`.

Se e só se a comunicação for possível, pede a duração da chamada (`timeReq()`) (número inteiro) ou o número de caracteres (`charReq()`) e imprime a mensagem `costOfMessage(long)` com o custo da ligação estabelecida.

3 Inicialização e Persistência

Os dados de um grt (clientes e telemóveis) podem ser definidos inicialmente a partir de um ficheiro de texto, segundo o formato abaixo. Sugere-se a utilização do método `split` da classe `String`. Não há entradas mal-formadas e assume-se que os identificadores referidos numa entrada já foram previamente descritos. Exemplo de ficheiro `data.txt`:

```
CLIENTE|cli001|Manuel␣Pinheiro|103443|NORMAL
CLIENTE|cli002|Pedro␣Pinheiro|103447|NORMAL
CLIENTE|Cli201|Ludgero␣Oliveira|103440|NORMAL
CLIENTE|cli␣Es|Maria␣Eucalipto|103441|OURO
CLIENTE|01|Oliveira␣Preto|103547|NORMAL
CLIENTE|cli003|Pedro␣Oliveira|103449|PLATINA
TELEMOVEL|969001|cli001|2G|LIGADO|-900
TELEMOVEL|969003|cli002|2G|LIGADO|0
TELEMOVEL|969002|cli002|3G|SILENCIO|330
TELEMOVEL|969007|cli␣Es|3G|LIGADO|700
TELEMOVEL|969008|cli003|2G|DESLIGADO|2500
TELEMOVEL|969009|cli003|2G|DESLIGADO|1200
TELEMOVEL|969010|cli003|2G|LIGADO|-2100
TELEMOVEL|969006|cli003|3G|LIGADO|4800
TELEMOVEL|969005|cli003|3G|LIGADO|3400
TELEMOVEL|969004|cli003|2G|LIGADO|1000
AMIGO|969001|969008,969009,969004
AMIGO|969004|969001
AMIGO|969003|969008
```

Execuções subsequentes do sistema não utilizam o ficheiro inicial: os dados deverão ser mantidos persistentemente usando a serialização do Java.

4 Considerações

4.1 Introdução de Modificações

O projecto deve ser concebido de forma a possibilitar futuras extensões ou alterações de funcionalidade com um impacto mínimo no código pré-existente. Assim, deverá ser simples (i) gerir vários operadores de redes de telemóvel; (ii) adicionar novos tipos de chamadas; (iii) adicionar novas classes de clientes; (iv) adicionar novas políticas de facturação: e (iv) adicionar novas formas de consulta.

4.2 Excepções

Note-se que, além das excepções descritas, é possível a definição de outras. As novas excepções não devem, no entanto, substituir as fornecidas nos casos descritos por este enunciado.

4.3 Execução

Usando os ficheiros `data.txt`, `input.dat` e `output.dat` (a disponibilizar) é possível verificar automaticamente se o programa produz o resultado esperado para um exemplo. Supondo que os ficheiros `input.dat` e `output.dat` estão no directório onde é dado o comando de execução e que a localização do pacote `grt` é possível com a definição apropriada da variável `CLASSPATH` (ou da opção equivalente `-cp`), se for dado o comando abaixo, deve ser produzido um ficheiro de saída `output1.dat`, que deve ser igual ao ficheiro `output.dat` (o ponto de entrada da aplicação a desenvolver é o método `main` da classe `grt.Grt`):

```
java -DImport=data.txt -Din=input.dat -Dout=output1.dat grt.Grt
```

Os ficheiros de saída (esperada e obtida) devem ser iguais em caso de sucesso. A comparação pode ser feita com o comando:

```
diff -b output.dat output1.dat
```

Este comando não deve produzir qualquer resultado quando os ficheiros são iguais. Note-se, contudo, que este teste não garante o correcto funcionamento do código desenvolvido, apenas verificando algumas funcionalidades.

Note-se que o projecto apenas será considerado se existir no repositório CVS associado à disciplina (ver Fénix), não sendo admissível qualquer outra forma de entrega. A estrutura do código fonte do projecto é a que está descrita no Fénix. Chama-se a atenção especialmente para a organização do código nas várias bibliotecas.

Não serão consideradas quaisquer alterações aos ficheiros de apoio disponibilizados: eventuais versões dessas alterações serão automaticamente substituídas durante a avaliação da funcionalidade do código entregue.

4.4 Erratas

Correcções a que este enunciado esteja sujeito serão devidamente anunciadas.

5 Funcionalidades Necessárias para Cada Entrega

A verificação da funcionalidade far-se-á unicamente através da interface textual, tanto na entrega intermédia como na entrega final.

A **entrega intermédia** pressupõe a definição de um diagrama UML (classes) com as classes da solução final, a correcta execução do processamento inicial da informação da aplicação de gestão de um operador de telemóveis a partir do ficheiro de entrada textual, e a funcionalidade de consulta de clientes com dívidas e consulta de telemóveis com saldo positivo. Todos os menus devem estar implementados (isto não implica que todos os comandos estejam completos). Deve ser implementada uma classe de teste que verifica se a listagem de clientes funciona correctamente.

A **entrega final** pressupõe a correcta execução de toda a funcionalidade e a apresentação da versão final do diagrama UML (classes).

5.1 Aspectos avaliados

Avaliação intercalar: funcionalidade, correcta aplicação dos conceitos de programação com objectos (máximo: 5 valores).

Correcta aplicação de padrões de desenho (máximo: 4 valores);

Correcta aplicação de conceitos de programação com objectos (máximo: 6 valores);

Correcto funcionamento do programa (máximo: 5 valores, a atribuir automaticamente através da execução de uma bateria de testes). É essencial que o código compile correctamente com as classes disponibilizadas.

Se algum teste falhar, devido a não ter seguido a ordem na inserção de dados especificada no ponto "funcionalidade máxima do gestor de telemóveis" §2, será atribuída a cotação de 0 (zero) valores nesse teste.

5.2 Penalizações

Relativamente à qualidade dos comentários que acompanham classes e métodos alterados ou introduzidos para a resolução deste exercício de programação: só é considerado o material correspondente à codificação da funcionalidade exigida para a entrega intercalar (máximo: 2 valores).

A não utilização da classe `pt.utl.ist.po.ui.UserInteraction` (interacção com o utilizador) conduz a uma classificação de 0 (zero) valores.

O uso de `System.exit()`, `Runtime.getRuntime().exit()` ou qualquer outro tipo de `exit()`, bem como o tratamento indiscriminado de excepções, e.g. uso de `catch(Exception e)` (ou hierarquicamente superiores) conduz a uma classificação de 0 (zero) valores.

A entrega de código não compilável conduz a uma classificação de 0 (zero) valores, na avaliação da funcionalidade do módulo correspondente. Não serão aceites quaisquer justificações.

O não cumprimento das convenções de codificação em Java implica uma penalização de 25%.

Não são aceites entregas fora de prazo: trabalhos não entregues têm uma classificação de 0 (zero) valores. A indisponibilidade temporária do repositório, desde que inferior a 24 horas, não justifica atrasos na submissão de um trabalho. Sugere-se a verificação, com antecedência, dos requisitos exigidos pelo processo de submissão.

A entrega de um projecto pressupõe o compromisso de honra que o trabalho correspondente foi realizado pelos alunos referenciados nos ficheiros submetidos para avaliação. A quebra deste compromisso, ou seja, a tentativa de um grupo se apropriar de trabalho realizado por colegas, tem como consequência a reprovação de todos os alunos envolvidos (incluindo os que possibilitaram a ocorrência) neste ano lectivo.