Ciências ULisboa Faculdade de Ciências da Universidade de Lisboa

DEPARTAMENTO·DE·INFORMÁTICA

Programação 1 (LTI), 2016/2017

Projeto

(este enunciado tem 13 páginas)

uberTranslation

0. Contexto

Os avanços científicos e tecnológicos ao longo da história da humanidade têm permitido eliminar limitações nas condições naturais de comunicação entre os seres humanos.

Por exemplo, com o advento da escrita há cerca de seis mil anos, tornou-se possível que os interlocutores comunicassem em linguagem natural de forma assíncrona, sem terem de estar na presença um do outro em simultâneo. Com a escrita pode dizer-se que se quebrou a barreira do tempo na utilização da linguagem.
Por sua vez, com o advento da imprensa mecânica, há cinco séculos atrás, começou a quebrar-se a barreira social no acesso à informação escrita. As publicações generalizaram-se e deixaram de estar acessíveis apenas para um muito pequeno grupo de leitores.

"Torre de Babel" de Pieter Bruegel, c. 1563, óleo sobre painel, 144x155, Museu de História da Arte, Viena, Áustria.
A destruição da torre de Babel é concomitante com a imposição aos seres humanos de diferentes idiomas ininteligíveis entre si, ambos

castigos infligidos por Deus de acordo com a narrativa bíblica em Génesis 11:1-9.

Há algumas décadas atrás um outro choque tecnológico para a linguagem natural teve lugar com o advento das telecomunicações. A barreira do espaço na utilização da linguagem foi quebrada, passando então a ser possível aos interlocutores comunicarem de forma síncrona apesar de não se encontrarem presentes no mesmo local.

Na comunicação em linguagem natural, resta porém ainda a imensa barreira da diversidade linguística. No mundo, existem cerca de 6 000 idiomas diferentes e apenas uma parte da população do planeta consegue dominar mais um ou dois outros idiomas para além da sua língua materna. A informação contida na esmagadora maioria dos documentos escritos por outros seres humanos nas suas línguas maternas mantém-se assim inacessível para cada um de nós.

A tradução de documentos é uma forma de mitigar esta barreira, mas requer um processo de tratamento dos textos que tem sido até agora lento e inacessivelmente caro.

Com a ajuda das novas tecnologias, a startup ReBaBel julga ter encontrado uma oportunidade de negócio neste desafio colocado à comunicação entre seres humanos pela barreira do multilinguismo. Recorrendo a técnicas de crowdsourcing quer tornar a tradução muito mais barata e mais rápida. Com uma plataforma online, vai receber pedidos de tradução de clientes que são de imediato reencaminhados para um conjunto de colaboradores que os devolvem executados. Ligados à internet, os colaboradores podem estar em qualquer ponto do mundo, trabalhar em part-time em acumulação com outra atividade profissional, e ser competentes nos mais diversos e raros pares de idiomas.

Um dos componentes desta plataforma é a uberTranslation, uma aplicação que permite otimizar a atribuição a tradutores de documentos a traduzir, maximizando a satisfação das condições de uns e de outros. Essa é a aplicação que vai ser desenvolvida no presente exercício pedagógico de programação.

1. Software a desenvolver

Objetivo

Com uma finalidade pedagógica, usando Python 2.7, neste projeto vai implementar o software `uberTranslation`. É um software que apoia o crowdsourcing de tarefas de tradução usado pela empresa ReBaBeL para gerir a atribuição de pedidos de tradução aos membros da sua rede de contribuidores na web.

Funcionalidade

O seu programa recebe uma listagem dos tradutores a qual caracteriza, num dado momento, cada um dos tradutores quantos a aspetos relevantes para a sua contribuição no atendimento de pedidos de tradução. O seu programa recebe também uma listagem dos pedidos de tradução que se encontram por atribuir a tradutores até esse mesmo dado momento.

Por um lado, o seu programa entrega um plano de execução desses pedidos de tradução pelos tradutores.

Por outro lado, entrega ainda a listagem atualizada dos tradutores, após os pedidos de tradução terem sido distribuídos por eles.

Entrada

O programa recebe ficheiros com nomes e estruturas internas para arrumação de informação similares à dos seguintes exemplos fragmentários:

`translators23h55.txt`

```
Company:
ReBaBeL
Day:
07:11:2016
Time:
23:55
Translators:
Alonso Moreno, (english), (spanish), 3*, 1.120 , 3000, 50000, 23457,
12:12:2016
Hans Muller, (swedish; english), (german; italian), 2*, 0.954, 4500, 75000,
1256, 01:12:2016
Maria Sousa, (english; french), (portuguese), 1*, 0.600, 2500, 150000, 102522,
23:05:2017
...
Steven Smith, ...
```

tasks23h55.txt

```
Company:
ReBaBeL
Day:
07:11:2016
Time:
23:55
Tasks:
voyná i mir, russian, portuguese, 2*, 587287, price, editora leya
carta98765AB23X, portuguese, french, 1*, 4503, speed, ministério da cultura
menu20161110, catalan, chinese, 3*, 2455, quality, el bulli
osudy dobrého vojáka Švejka za světové války, czech, portuguese, 2*, 144238,
speed, bertrand
...
```

Saída

O programa produz dois ficheiros, um com a listagem dos tradutores atualizada e outro com a calendarização da execução dos pedidos de tradução. Com uma finalidade pedagógica, assume-se que as atualizações e a calendarizações são feitas em simultâneo e de dez em dez minutos.

O ficheiro com a listagem de tradutores atualizada tem uma estrutura interna similar ao ficheiro de entrada com a listagem dos tradutores. A diferença é que o cabeçalho é atualizado quanto à data e ao tempo (incrementado de 10 minutos em relação ao momento do ficheiro de entrada), e os dois últimos campos de cada tradutor, um com o volume de texto acumulado para traduzir e o outro com a data de entrega das suas traduções, são atualizados em função da calendarização feita dos pedidos de tradução.

O ficheiro com a calendarização da execução dos pedidos de tradução tem uma estrutura interna para arrumação de informação similar à do seguinte exemplo fragmentário:

schedule00h05.txt

```
Company:
ReBaBeL
Day:
08:11:2016
Time:
00:05
Schedule:
08:11:2016, Carlos Silva, 320, carta98765AB23X
09:11:2016, Núria Bel, 265, menu20161110
10:11:2016, Carlos Silva, 1203, carta2016-AR
...
14:09:2017, Vladislav Maraev, 42598, oyná i mir
```

Mais sobre especificação geral

- As diferentes listagens (tradutores, pedidos, calendarização) são guardadas em ficheiros .txt.

- Cada listagem começa com um cabeçalho que contém a indicação da companhia, do dia da operação, do tempo da operação e do âmbito do ficheiro (`Translators`, `Tasks` ou `Schedule`) como neste exemplo:

```
Company:
ReBaBeL
Day:
07:11:2016
Time:
00:55
Translators
```

- Cada ficheiro de entrada e de saída é nomeado de acordo com seguinte convenção: concatenação das strings que designam o âmbito do ficheiro, as horas, "h", os minutos, ".txt", em mínúsculas, como neste exemplo referente ao ficheiro com o cabeçalho do ponto anterior:

```
translators00h05.txt
```

- Na **listagem de tradutores**, a seguir ao cabeçalho, cada linha corresponde a um tradutor (cujos respetivos elementos informativos estão separados por vírgulas) estando a listagem ordenada de cima para baixo por ordem alfabética do nome do tradutor.
  Cada tradutor é caracterizado por nome (e.g. `Hans Muller`), línguas das quais pode traduzir (e.g. `(swedish; english)`), línguas para as quais pode traduzir (e.g. `(german; italian)`), qualidade do seu trabalho (viz. `1*`, `2*`, ou `3*`), tarifa que cobra em euros por palavra (e.g. `0.954`), o ritmo de tradução, em número de palavras que traduz por dia (e.g. `4500`), volume máximo de texto acumulado que aceita para traduzir, em número de palavras (e.g. `75000`), volume de texto que de momento tem acumulado para traduzir, em número de palavras (e.g. `1256`), e última data de entrega das traduções que esteja a realizar (e.g. `01:12:2016`) (esta data será igual à data do cabeçalho se o tradutor não tiver nada entre mãos), como ilustrado no seguinte exemplo:

```
Hans Muller, (swedish; english), (german; italian), 2*, 0.954,
4500, 75000, 1256, 01:12:2016
```

  O cabeçalho indica a data e a hora da última atualização da listagem dos tradutores. Com uma finalidade pedagógica, assume-se que as atualizações são feitas de dez em dez minutos.

- Na **listagem de pedidos de tradução**, a seguir ao cabeçalho, cada linha corresponde a um pedido de tradução (cujos elementos informativos estão separados por vírgulas) estando a listagem ordenada pela ordem de chegada dos pedidos.

Cada pedido de tradução é caracterizado por um identificador (e.g `oyná i mir`), pelo idioma em que o texto se encontra escrito (e.g. `russian`), o idioma para que deve ser traduzido (e.g. `portuguese`), o nível mínimo de qualidade do tradutor (e.g. `2*`), o número de palavras a traduzir (e.g. `587287`), o critério de prioritização (viz. `price`, `speed`, ou `quality`), e a entidade que fez o pedido (e.g. `editora leya`), como ilustrado no seguinte exemplo:

```
oyná i mir, russian, portuguese, 2*, 587287, price, editora leya
```

O cabeçalho indica a data e a hora desde que têm estado a ser registados os pedidos, que devem ser idênticas à data e hora da última atualização do ficheiro dos tradutores.

- Na **calendarização dos pedidos de tradução**, a seguir ao cabeçalho, cada linha corresponde a um pedido calendarizado (cujos elementos informativos estão separados por vírgulas) estando a listagem ordenada por ordem crescente do momento de conclusão da tarefa.
  Cada pedido calendarizado é caracterizado pela data da entrega (e.g. `14:09:2017`), pelo nome do tradutor (e.g. `Vladislav Maraev`), pelo custo da execução da tradução, em euros (e.g. `42598`), e pelo identificador do texto a traduzir (e.g. oyná i mir), como ilustrado no seguinte exemplo:

  ```
  14:09:2017, Vladislav Maraev, 42598, oyná i mir
  ```

  O cabeçalho é similar aos dos ficheiros de entrada, atualizado quanto à data e ao tempo (incrementado de 10 minutos em relação ao momento dos ficheiros de entrada).
  Os pedidos de tradução devem ir sendo atribuídos a tradutores de acordo com a sua ordem de chegada, seguindo a ordem no ficheiro de entrada com a listagem de pedidos.
  No ficheiro de saída com a calendarização dos pedidos atribuídos, este são ordenados pela data crescente de concretização, do início para o fim do ficheiro.
  A **data de entrega** é determinada pelo volume de palavras a traduzir e pelo ritmo de tradução do tradutor, contada a partir da data da próxima entrega desse tradutor.
  O **tradutor** a quem é atribuída a tradução é aquele que melhor permite satisfazer o critério de prioritização indicado no pedido, desde que essa atribuição não ultrapasse o volume máximo de texto acumulado que esse tradutor aceita para traduzir.
  A um mesmo tradutor será atribuído mais de um pedido de tradução se após a atribuição de pedidos anteriores, esse tradutor é o que melhor satisfaz as condições do pedido em causa.
  - Se o critério for `quality`, deve ser atribuído um tradutor com nível `3*` (havendo vários, atribui-se o que estiver disponível mais cedo; havendo vários disponíveis nesse mesmo primeiro momento possível, atribui-se o que cobra uma tarifa mais baixa).

Para os restantes critérios, viz. `price` e `speed`, se o nível pretendido for *k**, então atribui-se um tradutor cujo nível seja *k** ou maior.
- Se o critério for `price`, atribui-se um tradutor que cobrar a tarifa mais baixa (havendo vários, atribui-se o que tiver mais baixo nível de desempenho; havendo vários disponíveis com o mesmo nível de desempenho, atribui-se o que estiver disponível mais cedo).
- Se o critério for `speed`, atribui-se um tradutor que estiver disponível mais cedo (havendo vários, atribui-se o que tiver mais baixo nível de desempenho aceitável; havendo vários disponíveis com o mesmo nível de desempenho, atribui-se o que cobrar a tarifa mais baixa).
Subsistindo eventualmente ainda empates na observância dos critérios de prioritização e suas regras de desempate, a tarefa é atribuída por ordem alfabética crescente dos nomes dos tradutores empatados quantos aos demais critérios relevantes.
Caso não haja nenhum tradutor que satisfaça as condições, deve ser colocada uma linha para o documento a traduzir com a data do cabeçalho e o nome do documento e as palvavras reservadas `not-assigned` e `not-applicable` como neste exemplo:

```
07:11:2016, not-assigned, not-applicable, oyná i mir
```

O **custo** da execução da tradução (arredondado às unidades) é determinado pelo volume de palavras a traduzir e pela tarifa cobrada pelo tradutor.

- Assume-se que a **atualização** da listagem dos tradutores e a **calendarização** das tarefas de tradução são feitas em simultâneo, de dez em dez minutos.

Especificação em pormenor

A especificação em pormenor do programa é feita através da especificação das suas funções, de acordo com as convenções adotadas no curso. As especificações nos esqueletos dos módulos fornecidos juntamente com este enuciado não se destinam a ser exemplificativas: têm de ser seguidas.

Programação por contrato tem de ser a abordagem seguida.

Estrutura da aplicação

A aplicação `uberTranslation` é composta pelo programa `rebabel.py` e pelos seguintes módulos a que este recorre:

```
constants.py
dateTime.py
readingFromFiles.py
scheduling.py
writingToFiles.py
```

Estes módulos devem incluir, entre possivelmente outras que entender necessárias ou convenientes, as funções apresentadas nos esqueletos e stubs disponibilizados em associação com o presente enunciado. O código desses módulos e funções tem de ser completado e pode ter de ser corrigido. As especificações fornecidas têm de ser respeitadas e as restantes têm de ser completadas.

O programa `rebabel.py` por sua vez contem uma função cuja chamada assegura o funcionamento da aplicação.

Exceção

Deve ser lançada a exceção:

```
Input file error: scope or time inconsistency between name and header in file
<name of file>.
```

quando num ficheiro de input se verificar inconsistência entre o seu nome e o seu cabeçalho quanto ao tempo (e.g. `01h35`) e/ou ao âmbito (`Translators`, `Tasks` ou `Schedule`). Deve ser lançada uma exceção por cada ficheiro em que ocorrer algum deste tipo de inconsistência.

Em vista de conter o projeto dentro dos seus limites pedagógicos, as pré-condições sobre a restante estrutura interna dos ficheiros de input, respeitante ao formato de arrumação da informação, exemplificado acima, não devem ser verificadas (assumimos que os ficheiros vêm todos bem estruturados).

Linguagem

A linguagem do input e output do software para utilizadores humanos é o inglês.

A linguagem da documentação, especificação, nomeação de funções, variáveis, comentários no código etc é também o inglês.

Executar o software

O software é executado através da seguinte instrução na linha de comandos:

```
python rebabel.py inputFile1 inputFile2
```

`inputFile1` é um ficheiro com a listagem dos **tradutores**, `inputFile2` é um ficheiro com a listagem dos **pedidos de tradução**, indicado na linha de comando por esta ordem.

Os ficheiros de saída produzidos são escritos na mesma diretoria onde se encontram os ficheiros de input. Um, com a calendarização, tem o nome `scheduleXXhYY.txt`, e o outro, com a listagem atualizada dos tradutores, o nome de `translatorsXXhYY`, em que XXhYY deve representar a hora que resulta de acrescentar 10 minutos ao tempo e data indicados nos ficheiros de entrada.

Dicas

Para ordenação de coleções, sugere-se a utilização do método `sort` ou da função `sorted` da biblioteca padrão do Python. Um pequeno manual encontra-se aqui:
https://wiki.python.org/moin/HowTo/Sorting

Para obter os nomes dos ficheiros a partir da instrução de arranque do programa na linha de comandos acima indicada, sugere-se a utilização da varíavel `argv` do módulo `sys`.
Especificação e explicação encontram-se aqui:
https://docs.python.org/2/library/sys.html
https://www.tutorialspoint.com/python/python_command_line_arguments.htm

2. Desenvolvimento do software

Grupos

O projeto tem de ser realizado por grupos de exatamente 2 alunos. Cada estudante ERASMUS deve fazer grupo com um estudante não-ERASMUS. Os grupos podem conter alunos de diferentes turmas. Os grupos registam-se no site da disciplina.

A única FORMA DE REGISTO de grupos é através do site da disciplina, em:
https://moodle.ciencias.ulisboa.pt/course/view.php?id=5

Elementos fornecidos aos alunos

Para a elaboração da componente de avaliação respeitante ao projeto, são fornecidos os seguintes elementos, que se encontram no site da disciplina:

- presente enunciado
- esqueleto dos módulos com especificações das funções
- exemplos com os ficheiros de input e correspondentes ficheiros de output

Máximas

Os estudantes a realizar o presente projeto são tipicamente programadores principiantes. Têm toda a vantagem em observar as seguintes máximas, que ainda não tiveram oportunidade de descobrir por si próprios:

1. "já"
positivo: começar a resolver o projeto agora, no momento em que este enunciado foi publicado
negativo: esperar até alguns dias antes do prazo de entrega para começar leva ao desastre

2. "passo a passo"
positivo: ir fazendo e testando pequenas partes do código progressivamente
negativo: esperar para testar até haver uma primeira versão total ou completa leva ao desastre

3. "desbloquear rápido"
positivo: falar com os docentes (e colegas) para esclarecer dúvidas e desbloquear impasses logo que estes surgem
negativo: esperar por futuro rasgo solitário de inspiração súbita leva ao desastre

Apoio para a resolução do projeto

Continuam ao dispor os meios de apoio pedagógico para os alunos desta disciplina, que se encontram disponíveis desde o início do curso, e que podem e devem ser usados para apoio à resolução do presente projecto. Relembra-se que são os seguintes:

- contato com os docentes ao **final das aulas** ao longo do semestre
- horários de **atendimento** presencial, individual e personalizado, aos alunos ao longo da semana
  https://moodle.ciencias.ulisboa.pt/course/view.php?id=5
- **forum** da disciplina, com acesso por todos os estudantes
  https://moodle.ciencias.ulisboa.pt/mod/forum/view.php?id=22
- espaço de **notícias** da disciplina
  https://moodle.ciencias.ulisboa.pt/mod/forum/view.php?id=23

Dada a natureza da tarefa a concretizar e o contexto do código em que eventuais dificuldades surgem, esclarecimentos sobre a resolução do projeto devem ser obtidos através destes meios de apoio, não sendo atendíveis através de mensagens de email para os docentes.

3. A componente de avaliação

Elementos a entregar pelos alunos para avaliação

Uma pasta com o ficheiro com relatório de implementação e com os ficheiros de código desenvolvidos, incluindo os seguinte sete ficheiros (e outros se for o caso):

```
constants.py
dateTime.py
readingFromFiles.py
rebabel.py
relGrupoN.pdf
scheduling.py
writingToFiles.py
```

A pasta deve ter o nome `uberTransGroupN`, em que `N` é o número do grupo, atribuído no processo de inscrição do grupo. Por exemplo, para o grupo de alunos que recebeu o número 786, a pasta deve ter o nome `uberTransGroup786`.

A pasta tem de ser submetida zipada, com o nome `uberTransGroup786.zip`.

Cada um dos ficheiros de código, por sua vez, tem de conter nas primeiras linhas, como comentários, informação sobre o número do grupo e número e nome completo de cada membro do grupo que trabalhou no projeto, como exemplificado a seguir:

```
#2016-2017 Programacao 1 (LTI)
#Grupo 786
#51123 Maria Francisca Dias
#51456 Miguel Fernando Silva
```

Ficheiros de código sem algum destes elementos não serão avaliados.

Relatório de implementação

O relatório de implementação não deve ultrapassar duas páginas, tem o nome `relGrupoN.pdf` (em que N é o número do grupo) e tem de estar no formato .pdf (relatórios noutros formatos serão ignorados). Tem de ser estruturado de acordo com as seguintes **secções**:

1. Número do grupo
2. Número e nome completo de cada membro do grupo
3. Indicação detalhada do que cada membro do grupo fez para a resolução do projeto
4. Indicação de funções extra implementadas (se aplicável) e do seu funcionamento
5. Indicação das funcionalidades que ficaram por implementar (se aplicável)
6. Indicação de erros conhecidos (se aplicável)

O relatório pode ser escrito em português ou em inglês.

Dimensões em avaliação

Os projetos serão avaliados de acordo com as seguintes dimensões e ponderações:

A. 1 se está completo e funciona sem gerar erros ao compilar e correr sobre exemplos (1, 2 e 3) fornecidos,
   0 caso contrário
B. Correcção semântica (funciona como especificado no enunciado), 60%
C. Correcção pragmática (organizado como indicado no enunciado, estruturas de dados e abordagens algorítmicas ponderadas e práticas de programação apropriadas), 20%
D. Documentação (especificação, comentários q.b.), 10%
E. Legibilidade (nomeação perspícua, arrumação e formatação do código), 5%
F. Relatório de implementação, 5%

A classificação é encontrada através da fórmula A ***** (B + C + D + E + F)

Integridade académica

Como futuro profissional, espera-se de si uma atitude irrepreensível, em termos éticos e deontológicos. Tenha pois o maior cuidado em respeitar e fazer respeitar a lei da criminalidade informática.

Alunos detetados em situação de fraude ou plágio parcial ou total - plagiadores e plagiados, com ou sem a intervenção de intermediários - em alguma componente de avaliação ficam liminarmente com esta prova cancelada e serão alvo de processo disciplinar, o que levará a um registo dessa incidência no processo de aluno. Não queira ter de mostrar o seu diploma a um futuro empregador com uma incidência dessas registada.

Pode e deve haver entreajuda entre alunos, através da discussão de métodos e algoritmos aplicáveis. É porém da exclusiva responsabilidade de cada grupo tomar medidas para proteger o seu código de ser plagiado.

No processo de avaliação será usado software de apoio na detecção de plágio que compara a resposta de cada grupo com cada uma das respostas dos outros grupos.

Forma e data de entrega

Para submeterem a solução do vosso grupo a avaliação, **entregam um FICHEIRO .zip**, que resulta de se comprimir a pasta com os ficheiros de código desenvolvidos e o relatório (por exemplo, `uberTransGroup786.zip`).

A **única FORMA DE ENTREGA é a através do site da disciplina, em**:
https://moodle.ciencias.ulisboa.pt/course/view.php?id=5
Qualquer entrega noutra forma não será considerada para avaliação.

Para ser avaliada, a vossa solução deve ser submetida até ao **PRAZO de segunda-feira, 19 de Dezembro de 2016, 23h00 (hora de Lisboa)**.

Qualquer entrega ou resubmissão depois deste prazo não será considerada para avaliação.