FACULDADE·DE·CIÊNCIAS UNIVERSIDADE·DE·LISBOA

DEPARTAMENTO·DE·INFORMÁTICA

Programação 1, 2014/2015

Projeto

(este enunciado tem 8 páginas)

AirConveyorBelts

1. Software a desenvolver

Cenário

Os aeroportos são grande infra-estruturas que requerem software dedicado para apoiar a gestão dos vários processos de trânsito de aeronaves, equipamentos, passageiros, bagagens, etc., que se vão desenrolando ao longo do dia. Um desses processos consiste na entrega das bagagens aos passageiros que chegam de viagem.

Embora não tenha um impacto crítico na segurança dos passageiros, é ainda assim um processo da maior importância para a qualidade do serviço que é percepcionada pelos utilizadores. Uma espera demasiado longa por uma mala frente a um tapete rolante pode arruinar a imagem da companhia tranportadora aérea, que porém até cumpriu com todos os demais requisitos do serviço (até bem mais exigentes tecnologicamente).

Objetivo

Com uma finalidade pedagógica, usando Python, neste projecto vai implementar o software AirConveyorBelts. É um software que apoia a gestão da entrega de bagagens na chegada dos voos a aeroportos de pequena dimensão.

Funcionalidade

O seu programa recebe uma listagem dos voos que vão chegar num certo período. Recebe também a indicação dos tapetes de bagagens que ainda estão ocupados no início desse período.

O seu programa entrega, por um lado, uma listagem dos tapetes rolantes no aeroporto e da sua utilização pelos voos chegados para entrega das respetivas bagagens. O objectivo é, para cada voo, minimizar o tempo de espera pela primeira mala a sair.

Por outro lado, entrega ainda informação que, entre outras coisas, permite saber o tempo médio de espera por passageiro nesse período.

Dados

- O aeroporto funciona das 5h00 às 23h00. Só tem uma pista. Não há dois voos a chegar ao mesmo tempo.
- Cada voo é caracterizado por um código específico, origem, hora de chegada, número de passageiros, e número de bagagens que traz no porão. Exemplo:

  `TP539, Berlin, 08:40, 175, 237`
- Uma listagem de voos que vão chegar diz respeito a um período de 2 horas.
- Estima-se em 30 minutos o tempo médio necessário para entregar todas as bagagens de um voo com 200 bagagens desde que o tapete entra em operação.
- Para se saber que tapetes estão disponíveis, e quando, há que consultar o relatório de operação do período anterior.
- Um relatório de operação dirá respeito também a um mesmo período de 2 horas, e contém um cabeçalho e três partes.
  - O cabeçalho indica o nome do aeroporto, o dia da operação (dd:mm:yyyy), e o período de 2 horas a que respeita.
  - Na primeira parte de um relatório de operação encontra-se a informação por tapetes: para cada tapete (do B1 ao Bn), os voos que serviu (identificados como acima), indicando para cada um desses voos o tempo (hh:mm) de início e fim de operação. Eis um exemplo, em que o tapete é o B2, o tapete entrou em operação às 14:45 e terminou essa operação às 16:35, para servir o voo TP104:

    `B2, 14:50, 16:35, (TP104, Recife, 14:00, 346, 498)`
  - Na segunda parte de um relatório de operação encontra-se a informação por voos: para cada voo, o tapete de bagagens por que foram entregues as bagagens desse voo aos passageiros, e o tempo médio de espera por uma bagagem desde a chegada. Eis um exemplo, em que o voo TP104 foi servido pelo tapete B2, e o tempo médio de espera de uma bagagem foi de 83 min:

    `(TP104, Recife, 14:00, 346, 498), B2, 83 min`

    Este tempo médio de espera de bagagem para um dado voo obtém-se adicionando o tempo do período de espera desde a chegada do voo até ao seu tapete entrar em operação com metade do tempo do período de operação do tapete para esse voo.

- Na terceira e última parte de um relatório de operação temos a informação global sobre o aeroporto no período em questão, com a indicação: do tempo médio de espera de uma bagagem (em minutos); e do tempo máximo de espera de uma bagagem (em minutos).
  Para um dado voo, o tempo máximo de espera de uma bagagem é o tempo que medeia entre a chegada do voo e a disponibiização da última bagagem. Para o aeroporto, é o maior dos tempos máximos de espera de todos os voos.
  Para um dado voo o tempo médio de espera de uma bagagem é obtido como foi indicado acima. Para o aeroporto, é a média dos tempos médios de todos voos ponderada pelos diferentes números de bagagens dos diferentes voos.

Entrada

O programa recebe um ficheiro com uma estrutura similar à do seguinte exemplo:

```
Airport: Neverland
Number of belts: 3
Day: 06:11:2014
Period: from 14:00 to 16:00
Arrivals:
KLM75, Amsterdam, 14:35, 60, 50
AF111, Paris, 14:20, 50, 64
LH333, Frankfurt, 14:10, 112, 203
KLM71, Madrid, 14:55, 120, 100
TAP103, Salvador, 15:20, 174, 210
LH123, Berlin, 15:10, 115, 210
```

Num ficheiro de entrada, os voos podem não estar ordenados por tempo crescente de chegada.
O programa recebe também um ficheiro com o relatório de operação do período imediatamente anterior, como indicado a seguir.

Saída

O programa produz um relatório de operação com uma estrutura similar à do seguinte exemplo:

```
Airport: Neverland
Number of belts: 3
Day: 06:11:2014
Period: from 14:00 to 16:00
Belts:
B1, 14:20, 14:29, (AF111, Paris, 14:20, 50, 64)
B1, 14:35, 14:42, (KLM75, Amsterdam, 14:35, 60, 50)
B1, 15:10, 15:41, (LH123, Berlin, 15:10, 115, 210)
B2, 15:20, 15:51, (TAP103, Salvador, 15:20, 174, 210)
B3, 14:10, 14:40, (LH333, Frankfurt, 14:10, 112, 203)
B3, 14:55, 15:10, (KLM71, Madrid, 14:55, 120, 100)
Flights:
(LH333, Frankfurt, 14:10, 112, 203), B3, 15 min
(AF111, Paris, 14:20, 50, 64), B1, 4 min
(KLM75, Amsterdam, 14:35, 60, 50), B1, 3 min
(KLM71, Madrid, 14:55, 120, 100), B3, 7 min
(LH123, Berlin, 15:10, 115, 210), B1, 15 min

(TAP103, Salvador, 15:20, 174, 210), B2, 15 min
Global:
Average delivery time = 12 min
Longest delivery time = 31 min
```

Os tapetes são listados por ordem crescente do seu identificador, do `B1` ao `Bn`.

Se algum tapete, por exemplo o B2, não for usado, deve surgir a menção:

```
B2, not_used
```

Neste relatório de operação, os voos são listados por ordem crescente da hora da sua chegada.

Estrutura da aplicação

A aplicação AirConveyorBelts é composto pelo programa `conveyorBelts.py` e pelos seguintes módulos a que este recorre:

```
readInput.py
sort.py
assignBelts.py
printReport.py
```

Estes módulos devem incluir, entre possivelmente outras que entender necessárias ou convenientes, as seguintes funções:

`readInput.py`

```
def arrivalsFile(file_name):
 """Reads part of an input file with the arrivals into a list of flights.
 Requires: file_name, for arrivals, is a text file with the structure indicated in
 the quizz
 Ensures: list of tuples, each corresponding to one flight"""


def flightLine(line):
 """Convert a line with a flight identification into a standard format.
 Requires: line is a string whose content correspond to one line/flight as
      indicated in the quizz (flight code, origin, arrival time, nb passengers, nb
 lugages), as in the example "TP539, Berlin, 08:40, 175, 237"
 Ensures: a tuple where each element corresponds to each element of the flight as
      indicated above"""


def headerArrivalsFile(file_name)
 """Reads the header of an input file with the arrivals into a string.
 Requires: file_name, for arrivals, is a text file with the structure indicated in
 the quizz
 Ensures: string representing the header of the file_name, including line breaks"""


def operationReport(file_report):
 """Reads part of an input file with an operation report into a list of belts with
 the hours at which they are available to be used.
 Requires: file_report, for an operation report, is a text file with the structure
 indicated in the quizz
 Ensures: list of pairs, where the first component is the number of the belt and
 the second component the next time this belt will be available within the current
 period of operation"""
```

sort.py

```
def flights(arrivals):
 """Sorted list of flights according to their increasing hour of arrival.
 Requires: arrivals is a list of tuples, each corresponding to one flight
 Ensures: list of tuples, each corresponding to one flight, sorted according to
 their increasing arrival time."""


def belts(belts):
 """Sorted list of belts according to their increasing hour of availability.
 Requires: belts is a list of pairs, where the first component is the number of
 the belt and the second component the next time this belt will be available
 within the current period of operation
 Ensures: list of pairs, each corresponding to one belt, sorted according to their
 increasing hour of availability."""
```

assignBelts.py

```
def belts2flights(belts, flights):
 """Belts assigned to flights.
 Requires: belts is a list of pairs, where the first component is the number of
 the belt and the second component the next time this belt will be available,
 sorted according to their increasing hour of availability, and flights is a list
 of tuples, as described in the quizz, each corresponding to one flight, sorted
 according to their increasing arrival time.
 Ensures: list of lists whose first element is a belt, the second the time when it
 is available, and the third a flight (represented as a tuple)"""


def beltsTimes2flights(beltsflights):
 """Time of start and end of operation of belts assigned to flights, ensuring
 minimal time of wait for the passengers of each flight.
 Requires: beltsflights is a list of lists whose first element is a belt, the
 second the time when it is available, and the third a flight (represented as a
 tuple)
 Ensures: list of lists whose first element is a belt, the second the time when it
 starts into operation, the third when it stops, and the fourth the flight
 (represented as a tuple) delivering its lugages via that belt"""
```

printReport.py

```
def operationReport(inputFile, beltsflights, outputFile):
 """The file with the operation report.
 Requires: inputFile, for arrivals, is a text file with the structure indicated in
 the  quizz, beltsflights is list of lists whose first element is a belt, the
 second the time when it starts into operation, the third when it stops, and the
 fourth the flight (represented as a tuple) delivering its lugages via that belt,
 and outputFile is a text file where the operation report will be written
 Ensures: text file with the operation report as described in the quizz"""
```

Sugestões

Para a implementação do módulo `sort`, sugere-se a utilização da função `sorted` da biblioteca do Python.

Linguagem

A linguagem do input e output do software para utilizadores humanos é o inglês. A linguagem da documentação, especificação, nomeação de funções, variáveis, etc é também o inglês.

Executar o software

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

```
python conveyorBelts.py inputFile1 inputFile2 outputFile
```

2. A componente de avaliação

Grupos

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

A ÚNICA forma de registo de grupos é através do site da disciplina, em:
https://mocho.di.fc.ul.pt/course/view.php?id=35

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:

- o presente enunciado
- um exemplo com:
  - os seus dois ficheiros de input
  - o respectivo ficheiro de output

Apoio disponível para os alunos

Continuam disponíveis 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:

- contacto com os docentes ao **final das aulas** ao longo do semestre

- **5 horários de atendimento, individual e personalizado, aos alunos** ao longo da semana
  https://mocho.di.fc.ul.pt/file.php/35/descricaoProg1_20142015.pdf

- **fórum** da disciplina para facilitar a entreajuda dos alunos
  https://mocho.di.fc.ul.pt/mod/forum/view.php?id=5327

- espaço de conversa (**chat**) para assuntos da disciplina
  https://mocho.di.fc.ul.pt/mod/chat/view.php?id=13125

- espaço de **notícias** da disciplina
  https://mocho.di.fc.ul.pt/mod/forum/view.php?id=12900

N.B.: ***esclarecimentos devem ser obtidos através destes meios de apoio; não são esclarecidas dúvidas através de email.***

Elementos a entregar pelos alunos para avaliação

Uma pasta com o conjunto dos ficheiros de código desenvolvidos, que deve conter os ficheiros de código desenvolvido, e o ficheiro com relatório de implementação, incluindo os seguinte seis ficheiros:

```
conveyorBelts.py
readInput.py
sort.py
assignBelts.py
printReport.py
relGrupoN.pdf
```

A pasta deve ter o nome `airConveyorBeltsGroupN`, em que `N` é o número do grupo. Por exemplo, para o grupo de alunos com o número 13, a pasta deve ter o nome `airConveyorBeltsGroup13`.

Cada um dos ficheiros de código, por sua vez, tem de conter nas primeiras linhas, como comentários, os seguintes elementos:

- Número do grupo
- Número de nome completo de cada membro do grupo que trabalhou no projeto

N.B.: ***ficheiros sem algum destes elementos não serão avaliados.***

Forma e data de entrega

Entregam um ficheiro .zip, que resulta de se comprimir a pasta com os ficheiros de código desenvolvidos (por exemplo, `airConveyorBeltsGroup13.zip`).

A **ÚNICA forma de entrega é a através do site da disciplina**, em:
https://mocho.di.fc.ul.pt/course/view.php?id=35

Para ser avaliado, o resultado deve ser submetido até ao **PRAZO de sexta-feira, 12 de Dezembro de 2014, 23h00 (hora de Lisboa)**.

N.B.: ***qualquer entrega noutra forma ou depois deste prazo não será avaliada.***

Dimensões de avaliação

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

A. Relatório de implementação, 15%
B. Documentação (especificação, comentários q.b.), 5%
C. Legibilidade (apresentação, arrumação e formatação do código), 15%
D. Correcção sintáctica (compilação), 1 se não tem erros, 0 caso contrário
E. Correcção semântica (funciona como especificado no enunciado), 40%
F. Correcção pragmática (organizado como indicado no enunciado), 25%

O relatório de implementação não deve ultrapassar uma página, 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 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
6. Indicação de erros conhecidos (se aplicável)

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

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

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.

A nível académico, alunos detetados em situação de fraude ou plágio, tanto plagiadores como plagiados, qualquer que tenha sido o processo que levou ao plágio, em alguma prova ficam reprovados à disciplina 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.