Universidade do Estado do Amazonas
Escola Superior de Tecnologia
Data: 10 de Setembro de 2019
Professora: Elloá B. Guedes
Disciplina: Fundamentos Teóricos da Computação
Monitor: Diego Lucena de Medeiros

PROJETO PRÁTICO I
TRANSAÇÕES DE TRANSPORTE INTERGALÁTICO

1 Conhecendo a plataforma Run.Codes

A disciplina de *Fundamentos Teóricos da Computação* possui projetos práticos em sua avaliação, os quais possuem **caráter individual** e **obrigatório** e que serão executados por intermédio da plataforma *Run.Codes*.

O primeiro passo a ser realizado é o cadastro na plataforma. Acesse o site run.codes e, utilizando o seu **nome completo** e **e-mail institucional**. Após esta etapa, procure a disciplina **Fundamentos Teóricos da Computação** (ESTECP006) e cadastre-se na sua turma utilizando o código **QJB2**. Todos os alunos devem obrigatoriamente se cadastrarem até o dia **11/09/2019**. O não cadastro, o cadastro em turma incorreta ou a não realização dos exercícios nos prazos estabelecidos culminará em nota zero.

Para quem não conhece, o run.codes é uma plataforma automatizada para testes de entrada e saída. Cada aluno submete o seu código escrito na linguagem Python e este código é submetido a um conjunto de testes, previamente escritos e cadastrados pela professora e monitor. A nota do aluno é individual e obtida de maneira automática, correspondendo ao percentual de acertos nos testes. Por exemplo, se há 15 casos de testes cadastrados e o aluno acertou 12, a nota obtida é 8.

A partir do momento em que o exercício inicia até o momento do seu encerramento, o aluno pode submeter o código para a plataforma quantas vezes quiser, sem que isso afete a nota final. Por exemplo, se um aluno submeteu 10 vezes e acertou 12/15 casos de teste, a nota será a mesma de um outro aluno que acertou 12/15 mas que submeteu 300 vezes.

Uma outra regra a ser considerada na plataforma é que a última versão do código é sempre a que será considerada. Imagine que o aluno João Última Hora está enlouquecidamente programando o exercício e já tem 14/15 casos corretos mas, nos segundos finais do prazo limite submete alterações em seu código e cai para 8/15 acertos. Se o sistema encerrar, a versão 8/15 será a considerada para avaliação. Se João Última Hora tivesse aproveitado melhor o tempo e começado a resolver o exercício desde o momento em queficou disponível na plataforma, não teria passado esse sufoco e ficado com uma nota final tão ruim. Todos podemos aprender com o drama de João Última Hora e evitar tais problemas.

Algumas dicas finais para você ter um bom desempenho nos problemas práticos são:

1. Considere que seu programa recebe uma entrada de cada vez;

2. Efetue testes em seu programa antes de submetê-lo ao run.codes. É uma forma simples de conhecer como seu programa se comporta e uma oportunidade de acertar mais testes logo de primeira;

3. Aproveite o tempo;

4. Há boatos de que você não deve deixar seu computador saber quando você está com pressa!

2 Apresentação do Problema

Este projeto prático trata de uma situação hipotética e de caráter fictício, mas os conceitos trabalhados são úteis em vários contextos cotidianos e auxiliam na fixação de conceitos da disciplina. Leia as instruções a seguir com atenção para um bom desenvolvimento do projeto prático.

Considerando a melhoria no transporte intergalático, uma *startup* unicórnio de ex-alunos da UEA resolveu lançar um aplicativo de naves particulares para locomoção pelas galáxias conhecidas. Cada viagem espacial produz um *log* com dados que a identificam e a tornam válida, mas os *hackers* interplanetários também tentam lucrar indevidamente com o sistema de transporte, enviando transações falsas para tentar que sejam faturadas. Sua tarefa consiste em construir expressões regulares que ajudem na identificação das viagens autênticas, que são compostas dos seguintes elementos:

1. **Origem**. É uma galáxia dentre as galáxias conhecidas e com infraestrutura interplanetária adequada para o transporte, sendo uma opção dentre as listadas a seguir: Via Láctea, Andrômeda, Triângulo, Redemoinho, Sombreiro, Girassol, Grande Nuvem de Magalhães, Pequena Nuvem de Magalhães, Compasso, Anã de Fênix, Messier 87, Messier 32, Leo I, Messier 110. O nome da galáxia de origem é denotado em maiúsculas ou minúsculas e sem acentos ou caracteres especiais;

2. **Destino**. Uma das galáxias dentre as listadas anteriormente, desde que seja diferente da galáxia de origem. As mesmas regras de validação devem ser obedecidas;

3. **Dados do Passageiro**. Um número de identificação pessoal aplicado tanto para terráqueos quanto para outros indivíduos, organizado segundo três grupos de dígitos: AAA-GG-SSSS. Algumas regras se aplicam:
   a) Nenhum dos grupos podem ser composto inteiramente por zeros. Por exemplo, 000-12-1234, 123-00-1234 e 123-12-000 são exemplos de identificadores inválidos;
   b) Não podem iniciar com 666 ou com números no intervalo de 900 a 999;
   c) Não podem ter todos os dígitos iguais;
   d) Não podem ser iguais aos números 078-85-1120 e 219-09-9999, pois estes foram reservados para uso em comerciais de divulgação;

4. ***Timestamp***. Uma marca temporal do início da corrida, da forma: DD/MM/AAAA HH:NN, em que D,M,A,H,N denotam dígitos de 0 a 9 que compõem uma data e hora válidos;

5. **Tipo de Espaçonave**. Uma opção dentre as listadas a seguir, seguindo a categorização intergalática proposta em Star Trek: Nebula Class A, Nebula Class B, Intrepid Class, Niagaria Class, Wells Class, Holoship, Raven, Peregrine, Danube Class.

6. **Valor da Corrida**. Denotado segundo uma criptomoeda seguida do valor. No caso de Bitcoints, denotado por BTC seguido de um número no intervalo $]0, 1[$ separado por ponto e com três casas decimais. No caso de Ethereum, denotado por ETH seguido de um valor inteiro de até três dígitos. Por último, no caso de Litecoin, denotado por LTC seguido de um número de dois dígitos inteiro e dois dígitos decimais, separados por ponto.

7. **Código de Segurança**. Uma sequência de 12 símbolos composta de 3 letras maiúsculas, 4 dígitos, 2 caracteres especiais ($,@,%,(,*) e 3 letras minúsculas, dispostos em qualquer ordem.

8. **Hash md5**. Um número de 32 dígitos na base hexadecimal, em que os números de 10 a 15 nesta base são denotados em letra minúscula.

Uma viagem é considerada válida se todos os elementos acima descritos encontram-se presentes e respeitam as regras especificadas, estando dispostos na ordem em que foram apresentados e separados por vírgulas. Havendo espaços em branco em componentes em que estes não são significativos, estes devem ser ignorados na validação, ou seja, não devem interferir negativamente.

A entrada consiste em várias viagens, uma por linha, e sua tarefa consiste em validar cada linha, informado quando a mesma é válida (saída `True`) e quando é inválida (saída `False`). A entrada encerra quando a palavra `END` for encontrada. Ao final, você deve informar as somas acumuladas em cada moeda, na ordem: Bitcoin, Ethereum e Litecoin, usando duas casas decimais para denotar cada valor.

Para resolver o problema em questão, você deve utilizar a linguagem de programação Python 3 e obrigatoriamente fazer uso de expressões regulares. Soluções que não fizerem uso de expressões regulares serão anuladas. Os exemplos a seguir auxiliam a ilustrar entradas e saídas para o problema considerado.

3 Exemplos de Entradas e Saídas

Entrada	**Saída**

Via Lactea,ANDROMEDA,458-12-8541,09/05/2052 21:45,Nebula Class A, BTC 0.251,K$FfXa20@z18,e7680cee0538ac00aff721c5c96e3d4b
Andromedaz,Via Outra Lactea,458-00-8541,31/02/2052 66:99,Nebula Class X, XYZ 3.251,K$fX20@z18,e7680xyz0538ac00aff721c5c96e3d4b
END
	`True`
	`False`
	`BTC 0.25`
	`ETH 0.00`
	`LTC 0.00`

4 Observações Importantes

- Lembre-se, a entrada de dados é feita via `input` e a saída via `print`;

- Atenha-se exatamente ao padrão de entrada e saída fornecidos nos exemplos. Qualquer mensagem adicional na entrada ou na saída de dados pode culminar em incorretude;
- Cuidado ao copiar caracteres do PDF! Eles podem estar com codificação incorreta. Atente-se ao enunciado;
- Cada caso de teste pode possuir quantidades de transações diferentes;
- Na construção do seu programa você deve usar apenas os conceitos aprendidos em sala de aula. Respostas que utilizem bibliotecas prontas não serão consideradas;
- Em caso de plágio, todos os envolvidos receberão nota zero!
- Na execução do seu programa no *run.codes*, existem casos de testes que vão além dos exemplos mostrados a seguir. Esses casos de teste não serão revelados. Pense em exemplos de entradas e saídas que podem acontecer e melhore o seu código para capturá-las.

5 Prazos Importantes

- **Início**. 10/09/2019 às 17h20min (horário do servidor)
- **Encerramento**. 18/09/2019 às 23h55min (horário do servidor)

6 Links Úteis

- https://developers.google.com/edu/python/regular-expressions
- https://www.debuggex.com/cheatsheet/regex/python