Universidade do Estado do Amazonas
Escola Superior de Tecnologia
Data: 13 de Agosto de 2020
Professora: Elloá B. Guedes
Disciplina: Fundamentos Teóricos da Computação

Atividade Avaliativa 1.2
Filtrando Spams e Hams

1 Conhecendo a plataforma Run.Codes

A disciplina de *Fundamentos Teóricos da Computação* possui atividades práticas em sua avaliação, as quais devem ser desenvolvidas em **duplas**, possuem caráter **obrigatório** e que serão executadas 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** (exemplo: "Elloá Barreto Guedes" é adequado, enquanto "elloa guedes' não o é) e **e-mail institucional**. Após esta etapa, procure a disciplina **Fundamentos Teóricos da Computação** (ESTECP006) e cadastre-se na sua turma utilizando o código **RQH2**. Todos os alunos devem obrigatoriamente se cadastrarem até o dia **15/08/2020**. 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 dupla submete o código escrito na linguagem Python e este é submetido a um conjunto de testes de entrada e saída, previamente escritos e cadastrados pela professora. A nota do dupla é obtida de maneira automática, correspondendo ao percentual de acertos nos testes. Por exemplo, se há 15 casos de testes cadastrados e a dupla acertou 12, a nota obtida é 8. As submissões podem ser feitas por um componente da dupla ou por ambos, em que este último caso é preferível. Havendo diferença na nota dos componentes da dupla, será considerada a maior nota dentre eles.

A partir do momento em que o exercício inicia até o momento do seu encerramento, a dupla 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 que

ficou disponível na plataforma, não teria passado esse sufoco e ficado com uma nota final tão ruim. Aliás, e o colega de dupla de João Última Hora? Nem se sabe dele. Todos podemos aprender com o drama de João Última Hora e evitar tais problemas.

Algumas dicas finais para alcançar um bom desempenho nos problemas práticos:

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

O fluxo de e-mails têm crescido gradativamente nos últimos anos, pois caracteriza-se como uma forma prática, barata e rápida para as pessoas se comunicarem. Tentando explorar estes benefícios, porém, empresas mal intencionadas e golpistas têm produzido volumes massivos de e-mails indesejados, os tão conhecidos `spams`. Um bom filtro de e-mails deve ser capaz de separar bem os `hams` (e-mails autênticos) dos `spams` (e-mails falsos, enviados automaticamente, de maneira massiva e com conteúdo malicioso).

Vamos explorar uma das primeiras técnicas para filtragem de e-mails: o casamento de padrões! De acordo com esta técnica, a existência de certos padrões nos e-mails dá mais evidências de que este seja um `spam`. Considerando o contexto da disciplina, o casamento de padrões deverá ser feito com o uso das nossas queridas *expressões regulares*!

O seu programa irá receber uma string contendo o nome de um arquivo a ser aberto no diretório local. Este arquivo corresponde ao conteúdo de um e-mail. Ao abrir o arquivo para fins de leitura, você deve investigar os seguintes padrões:

1. **Begin Message**: denotado na primeira linha do arquivo, da seguinte forma: `-----beginmessage-----` (5 hífens antes e 5 hífens depois, sem espaços);
2. **Remetente**: Sob a forma `from: <email>;` (após a palavra `from:` há um único espaço em branco);
3. **Endereços de E-mails**: Seguindo um formato de e-mail simplificado, em que o início do endereço é feito com uma letra, há um arroba, e pelo menos um ponto após o arroba. São exemplos de e-mails válidos: ebgcosta@uea.edu.br, elloa.uea@gmail.com, c4rl0s@teste.com.au.;

4. **Destinatário**: Sob a forma `to: <email>;` (após a palavra `to:` há um único espaço em branco). O e-mail obedece ao padrão previamente apresentado;

5. **IP do autor**: Reflete o IP do dispositivo utilizado pelo autor para enviar o e-mail. É um número de 32 bits dividido em 4 octetos de bytes representados no formato decimal. São exemplos de IPs válidos: 192.168.1.2, 127.0.0.1, 255.255.255.255;

6. **Time stamp**: O e-mail é acompanhado de uma estampa de tempo com a data e hora do seu envio. Esta estampa é da forma `AAAA.MM.DD HH:MM:SS`.

7. **Separador**: Um separador da forma `----------` (23 hífens) que marca o fim do cabeçalho e o início do corpo da mensagem;

8. **Mensagem**: Um conjunto de palavras e números que compõem o objeto da comunicação entre remetente e destinatário. O corpo da mensagem deve ser processado em busca de padrões específicos, a citar:
   - Endereços de e-mail no corpo da mensagem;
   - Tags específicas HTML(`head,body,img,alt,href`) quer sejam de abertura ou fechamento, que podem dar indícios de vínculo a conteúdo externo malicioso;
   - Palavras suspeitas, em particular: `milionario`, `emprestimo`, `loteria`, `banco`, `heranca`, `seguidor` e `desconto`.
   - Palavras que tenham comprimento maior que 11 e que possuam mais consoantes que vogais;
   - Mais de 15 sinais de pontuação (`;`, `,` e `.`) no corpo inteiro da mensagem.

9. **End Message**: denotado após a mensagem, da seguinte forma: `-----endmessage-----` (5 hífens antes e 5 hífens depois, sem espaços);

Um e-mail é considerado `ham` quando todos os seus elementos estão caracterizados de maneira adequada, na ordem especificada, como apresentado anteriormente. Por conseguinte, será considerado `spam` qualquer arquivo cujo conteúdo viole pelo menos uma das regras acima. Desta maneira, a saída do seu programa deve ser uma impressão na tela da classificação final: `spam` ou `ham`.

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.

3 Exemplos de Entradas e Saídas

Entrada
`email1.txt`
Saída
`ham`
Conteúdo do arquivo1.txt

`-----beginmessage-----`
`from:  ebgcosta@uea.edu.br;`
`to:  monitoriaFTC@gmail.com;`
`194.138.1.125`
`2020.08.13 11:13:51`
`----------`
`Monitores,`
 	
`bom dia!`
 	
`preciso dos casos de teste hoje a tarde.`
 	
`Atenciosamente,`
`Elloa`
`-----endmessage-----`

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;
- A cada execução do programa será fornecida apenas uma entrada, cujo resultado deve ser exibido ao final do processamento;
- 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**. 13/08/2020 às 14h40min (horário do servidor)
- **Encerramento**. 20/08/2020 às 23h55min (horário do servidor)

6 Links Úteis

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