**Universidade do Estado do Amazonas**
**Escola Superior de Tecnologia**
**Data**: 26 de agosto de 2016
**Disciplina**: Fundamentos Teóricos da Computação
**Professores**: Elloá B. Guedes
**Aluno**:

Projeto Prático 1
Transações com BitCoins

# 1 Inscrição no Run.Codes

Você deve usar a plataforma Run.Codes para realizar as tarefas descritas neste projeto. Caso não tenha cadastro, primeiramente o faça em `https://run.codes/`. Utilize o seu e-mail institucional. Após a realização do cadastro, procure a turma ESTECP006 Fundamentos da Engenharia da Computação II - Fundamentos Teóricos da Computação. O código para matrícula nesta turma é H12M.

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 de programação determinada pelo professor da disciplina e este é submetido a um conjunto de testes previamente cadastrado pelo professor. A nota do aluno corresponde ao percentual de acertos nestes testes.

Algumas observações importantes. O aluno pode submeter quantas vezes quiser e efetuar diversos testes. A nota final não sofre influência do número de submissões, apenas do número de acertos ao final. Considere que seu programa recebe uma entrada de cada vez. Efetue testes em seu próprio computador antes de submeter ao run.codes, é uma forma mais simples de detectar o que pode estar havendo de errado com o seu código. Aproveite o tempo! Você tem até a data limite para submeter a versão final do seu código.

Do ponto de vista do professor, o run.codes é uma ferramenta excelente para detecção de plágio, evitando a cópia de resposta entre os estudantes.

# 2 Apresentação

Nas linguagens de programação, as *expressões regulares* fornecem uma maneira automatizada de verificar se determinadas entradas estão escritas de acordo com um padrão ou regra de formação. Seu objetivo neste projeto é resolver o problema a seguir utilizando expressões regulares na linguagem Python. Soluções que não utilizem expressões regulares serão desconsideradas.

Neste projeto você vai validar transações feitas com a criptomoeda BitCoin num cenário particular. Um usuário pode trocar BitCoins com um outro usuário ou pode enviar BitCoins a uma instituição financeira. Para tanto, deve fornecer o seu CPF e o CPF do destinatário, caso seja outro usuário, ou então o CNPJ, caso seja uma instituição financeira. Todos os usuários cadastrados possuem um login, que também deve ser informado na transação, bem como a senha,

que possui regras de construção específicas para garantir a segurança. Cada transação pode ser composta por um valor em BitCoins e mais um valor em uma determinada moeda. Além disso, toda transação possui um hash md5 que deve ter sua integridade checada. A seguir, alguns detalhes sobre os elementos que compõem uma transação deste tipo.

- **Login do usuário**. Sempre iniciado com um caractere minúsculo, podendo ser seguido de outros caracteres. Caracteres especiais não são permitidos, tais como $, #, espaço, nem dígitos etc. Não há limite de caracteres para o login. São exemplos de logins válidos: elloa, elloaGuedes, umLoginMuitoMuitoGrande, etc.

- **CPF do usuário**. Como estamos considerando apenas espectadores brasileiros, o CPF é utilizado para identificar o comprador. Este dado segue o padrão de CPFs amplamente conhecido. Exemplo: 123.456.789-01. Além de checar se a entrada está no formato de um CPF, deve-se adicionalmente aplicar o algoritmo para checar a validade do CPF fornecido, disponível em `http://www.macoratti.net/alg_cpf.htm`. Um CPF válido é aquele que passa no padrão e também no teste de validade;

- **Modalidade de transação**. Se há transação for para outro usuário, a modalidade da transação é `user`, se for para uma instituição financeira, a modalidade da transação é a letra `bank`.

- **Destinatário da transação**. Se a transação for do tipo `user`, segue-se um login e um CPF. Se a transação for do tipo `bank`, segue-se um banco e o seu CNPJ.

  - **Tipos de banco**. Os tipos de banco permitidos são apenas: brasil, caixa, itau, bradesco, safra e outro;
  - **CNPJ**. O Cadastro Nacional de Pessoa Jurídica identifica a instituição financeira. É composto por 14 dígitos, sendo da forma `XX.XXX.XXX/YYYY-ZZ` e há uma regra de formação que deve ser verificada para assegurar sua integridade, que pode ser vista em `http://www.geradorcnpj.com/algoritmo_do_cnpj.htm`

- **Valor de BitCoins**. O valor de BitCoins a ser transferido começa com $B$$, segue-se um espaço e depois há um número real positivo com duas casas decimais separadas da parte inteira por um ponto. São exemplos de valores válidos: $B$$132.00, $B$$7347264926.01, etc.

- **Complemento da transação**. Nem toda transação possui um complemento. Porém, quando houver, será um valor em reais apresentado de maneira similar ao valor de BitCoins, exceto que a moeda será representada por $R$$.

- **Hash da transação**. A cada transação está associado o resultado de um hash md5. Você deve checar se o hash da transação é válido, sendo composto por 32 dígitos formados por números de 0 a 9 ou caracteres de $a$ até $f$ em minúsculo neste cenário.

Os elementos que compõem uma transação são apresentados em uma única linha conforme a ordem especificada, separados por espaços em branco. Uma transação é válida quando todos os seus elementos são caracterizados de maneira adequada, na ordem especificada, como apresentado anteriormente. Não precisa fazer checagens se há saldo na conta do usuário ou afins, assuma que isso irá compor outro módulo deste sistema e que uma outra pessoa está responsável por implementar isto. De maneira similar, não preocupe-se em saber o que gerou o hash md5, apenas verifique se o hash que aparece na entrada é válido. A equipe de criptografia está empenhada nestas outras preocupações.

De maneira resumida, a entrada do seu problema é uma string contendo uma transação. A saída é a palavra "True" quando a transação é válida e "False" em caso contrário.

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 Links Úteis

- `https://developers.google.com/edu/python/regular-expressions`
- `http://goo.gl/SWwe4R`
- `https://www.debuggex.com/cheatsheet/regex/python`
- `http://www.miraclesalad.com/webtools/md5.php`
- `https://pt.wikipedia.org/wiki/Cadastro_Nacional_da_Pessoa_Jur%C3%ADdica#Algoritmo_de_Valida.C3.A7.C3.A3o`
- `http://www.4devs.com.br/gerador_de_cnpj`

# 4 Exemplos de Entrada e Saída

- **Entrada**: fadaDoDente 335.132.065-50 user criancaJanelinha 516.257.862-20 B$ 2.00 R$ 2.00 698dc19d489c4e4db73e28a713eab07b ⇒ **Saída**: True
- **Entrada**: grampo 491.207.790-97 bank gringotes 33.189.734/0001-10 B$ 1.000.000,00 9584ec80754574635f35e63e9262f95a **Saída**: False

# 5 Prazos Importantes

- Apresentação da atividade: 26/08
- Cadastro da atividade no Run.Codes: 26/08
- Data limite de entrega: 02/09, 23h59min no horário do servidor