TÉCNICO LISBOA

Fundamentos da Programação

Primeiro Projecto

26 de Setembro de 2014

Números de cartões de crédito

Os números dos cartões de crédito partilham um esquema de numeração comum de acordo com a norma ISO / IEC 7812. O esquema mais utilizado para os números de cartões de crédito tem entre 12 e 19 dígitos com a seguinte constituição:

- Os seis dígitos iniciais identificam o emissor do cartão de crédito (conhecido por "Issuer Identification Number"ou IIN);
- Este dígitos são seguidos por um número variável de dígitos correspondente ao identificador da conta associada ao cartão;
- Um dígito final de verificação calculado usando o algoritmo Luhn.

O primeiro dígito do número de um cartão de crédito (conhecido por "Major Industry Identifier"ou MII) representa a categoria da entidade que emitiu o cartão. O MII define as seguintes categorias do emissor:

1: Companhias aéreas;

2: Companhias aéreas e outras tarefas futuras da indústria;

3: Viagens e entretenimento e bancário / financeiro;

4: Serviços bancários e financeiros;

5: Serviços bancários e financeiros;

6: Merchandising e bancário / financeiro;

7: Petróleo e outras atribuições futuras da indústria;

8: Saúde, telecomunicações e outras atribuições futuras da indústria;

9: Atribuição nacional.

Por exemplo, o American Express e o Diners Club International estão na categoria de viagens e entretenimento; o VISA e o MasterCard são do sector bancário e financeiro.

O IIN (incluindo o dígito inicial MII) identifica a instituição que emitiu o cartão. A tabela 1 apresenta alguns valores do IIN.

Rede emissora	Abreviatura	Dígitos iniciais do IIN	Número de dígitos
American Express	AE	34 e 37	15
Diners Club International	DCI	309, 36, 38 e 39	14
Discover Card	DC	65	16
Maestro	M	5018, 5020, 5038	13 ou 19
Master Card	MC	50 a 54 e 19	16
Visa Electron	VE	4026, 426, 4405, 4508	16
Visa	V	4024, 4532, 4556	13 ou 16

Tabela 1: Composição do IIN.

O resto do número é atribuído pelo emissor. A maioria dos números de cartões de crédito é validada utilizando o algoritmo de Luhn:[1]

1. Retire o último dígito do número. O último dígito é o que serve para verificação;
2. Inverta o número;
3. Multiplique os dígitos em posições ímpares (1, 3, 5, etc.) por 2 e subtraia 9 a todos os números resultantes que sejam superiores a 9;
4. Adicione todos os números, incluindo o número de verificação;
5. O resto da divisão da soma por 10 dever ser 0.

Exemplo:

Passo																
Número original	4	5	5	6	7	3	7	5	8	6	8	9	9	8	5	5
Remova o último dígito	4	5	5	6	7	3	7	5	8	6	8	9	9	8	5	
Inverta os dígitos	5	8	9	9	8	6	8	5	7	3	7	6	5	5	4	
Duplica ímpares	10	8	18	9	16	6	16	5	14	3	14	6	10	5	8	
Subtrai 9	1	8	9	9	7	6	7	5	5	3	5	6	1	5	8	
Adicione números	90															
Resto da divisão por 10	0															

1 Trabalho a realizar

Pretende-se com este trabalho a escrita de um programa em Python que efectua dois tipos de operações: (1) verifica a correcção do número de um cartão de crédito e (2) gera números correctos para cartões de crédito. Para isso:

1. Deverá escrever um programa que recebe um número correspondente a um cartão de crédito, determina se este é um número correcto e, se o for, indica a rede emissora e a categoria da entidade que emitiu o cartão (baseado na tabela 1).

   O seu programa deve ser invocado através da função `verifica_cc` que recebe como argumento um inteiro correspondente a um possível número de um cartão

[1] Patenteado em 1960 por Hans P. Luhn (U.S. Patent 2,950,048, *Computer for Verifying Numbers*).

de crédito e que devolve um tuplo contendo a categoria e a rede do cartão se o número corresponder a um cartão de crédito ou a cadeia de caracteres `'cartao invalido'` em caso contrário. Por exemplo:

```
>>> verifica_cc(38153682601755)
('Viagens e entretenimento e bancario', 'Diners Club International')
>>> verifica_cc(4532728243332223)
('Servicos bancarios e financeiros', 'Visa')
>>> verifica_cc(4556223160581)
'cartao invalido'
>>> verifica_cc(5485060128076517)
('Servicos bancarios e financeiros', 'Master Card')
>>> verifica_cc(6554615813812884)
'cartao invalido'
```

2. Deverá escrever um programa que recebe como argumento uma cadeia de caracteres correspondente à abreviatura de uma entidade emissora do tipo "serviços bancários e financeiros" e que gera um número de cartão de crédito válido emitido por essa entidade baseado na tabela 1.

   O número da conta criado pela entidade emissora deverá ser gerado aleatoriamente utilizando a função a função `random()`, localizada na biblioteca `random`, que gera números aleatórios no intervalo $[0, 1[$. Se para uma dada rede emissora existitem múltiplos números do IIN ou diferentes valores para o tamanho do número do cartão de crédito, o seu programa deve escolher um deles de modo aleatório.

   O seu programa deve ser invocado através da função `gera_num_cc` de um argumento do tipo cadeia de caracteres correspondente à abreviatura de uma rede emissora (ver tabela 1). Por exemplo:

```
>>> gera_num_cc('MC')
1953081084203475
>>> gera_num_cc('DC')
6567518003213645
>>> gera_num_cc('V')
4532440306227148
>>> gera_num_cc('V')
4556245018079
```

2 Sugestão de passos a seguir

Tendo em atenção as operações que vai ser necessário efetuar sobre números de cartões de crédito, estes números devem ser convertidos em cadeias de carateres. Para tal deve usar a função embutida `str`; por exemplo, `str(5364)` devolve a cadeia `'5364'`.

2.1 Verificação do número de um cartão de crédito

Para verificar a validade de um número de um cartão de crédito, terá de verificar os seguintes aspectos:

1. O número tem que satisfazer o algoritmo de Luhn;
2. O prefixo do número, isto é, os seus primeiros dígitos, deve ser um dos indicados na Tabela 1. Por exemplo, o número `8764532910871` não é válido porque não existe nenhum IIN que comece por `8`.
3. O comprimento do número deve ser um dos possíveis para o IIN do número. Esta informação também está representada na Tabela 1. Por exemplo, o número `402687645329` não é válido porque os números que começam pelo prefixo `4026` têm de ter 16 dígitos.

Algoritmo de Luhn

Para verificar se o número satisfaz o algoritmo de Luhn, começe por escrever a função `calc_soma`, que recebe uma cadeia de carateres, representando um número de cartão, sem o último dígito. A função devolve a soma ponderada dos dígitos de `n`, calculada de acordo com o algoritmo de Luhn. Por exemplo,

```
>>> calc_soma('3248')
    18
```

Como precisa de calcular uma soma e não pode somar cadeias de carateres, use a função `eval`; esta função, quando recebe uma cadeia formada apenas por dígitos, devolve o inteiro correspondente. Por exemplo, `eval('8')` devolve o inteiro `8`.

Usando a função `calc_soma`, escreva a função `luhn_verifica`, que recebe uma cadeia de carateres, representando um número de cartão e que devolve *verdadeiro*, se o número passar o algoritmo de Luhn, e *falso* em caso contrário. Esta função não verifica se o número corresponde a alguma rede emissora. Por exemplo,

```
>>> luhn_verifica('4556245018079')
True
>>> luhn_verifica('4556245018072')
False
```

Prefixo do número

Da análise da Tabela 1, concluímos que os prefixos dos IINs que identificam as várias redes emissoras podem ter comprimentos diferentes; concluímos também que algumas redes têm um prefixo único (por exemplo o *Discover Card*), e outras têm vários prefixos possíveis (por exemplo o *Visa Electron*).

Comece por escrever a função `comeca_por`, que recebe duas cadeias de carateres, `cad1` e `cad2`. A função devolve *verdadeiro* se `cad1` começar por `cad2`. Em caso contrário devolve *falso*. Por exemplo,

```
>>> comeca_por('12345678', '123')
    True
>>> comeca_por('12345678', '23')
    False
>>> comeca_por('123', '12345678')
    False
```

Use esta função para escrever a função `comeca_por_um`, que recebe uma cadeia de carateres, `cad`, e um tuplo de cadeias de cadeias de caracteres, `t_cads` e que devolve *verdadeiro* apenas se `cad` começar por um dos elementos do tuplo `t_cads`. Por exemplo,

```
>>> comeca_por_um('36238462919584', ('309',' 36', '38', '39'))
    True
>>> comeca_por_um('36238462919584', ('34', '37'))
    False
```

Rede emissora

Escreva agora a função `valida_iin`, que recebe uma cadeia de carateres, representando um número de cartão. A função devolve a cadeia de carateres correspondente à rede emissora do cartão, se esta existir. Em caso contrário, devolve a cadeia de carateres vazia. Esta função apenas verifica os dígitos iniciais e o comprimento da cadeia. Por exemplo,

```
>>> valida_iin('4508654345231273')
    'Visa Electron'
>>> valida_iin('45086')
    ''
```

Note que o número do primeiro exemplo não passa o algoritmo de Luhn, e, consequentemente, não é um número válido. Essa verificação não é no entanto feita pela função `valida_iin`.

MII

Para terminar, escreva a função `categoria`, que recebe uma cadeia de carateres, e devolve uma cadeia correspondente à categoria da entidade correspondente ao primeiro caráter da cadeia. Deverá decidir o que fazer se a categoria não estiber correcta. Por exemplo,

```
>>> categoria('193')
    'Companhias aereas'
>>> categoria('1')
    'Companhias aereas'
```

Note a ausência de carateres acentuados nos exemplos acima. No seu programa não deve usar carateres acentuados, nem mesmo nos comentários.

E agora é simples ... basta usar as funções que definiu para escrever a função `verifica_cc`.

2.2 Geração do número de um cartão de crédito

Para gerar um número de cartão de crédito de determinada rede, deverão em primeiro lugar ser escolhidos uma rede emissora e um comprimento, com base na Tabela 1. Por exemplo, se a rede for a "Maestro", poderá ser escolhido como prefixo 5018, 5020 ou 5038, e como comprimento 13 ou 19.

Em seguida deverão ser acrescentados (ao prefixo escolhido) dígitos entre 0 e 9 (gerados aleatoriamente), até atingir um comprimento igual ao comprimento escolhido menos um. Se no exemplo anterior tivesse sido escolhido o prefixo '5020' e o comprimento 13, poderia ser gerado o número 5020726819227, cujo comprimento é 12.

Para terminar, deverá ser acrescentado o dígito de verificação. Para gerar este dígito, use em primeiro lugar a função `calc_soma` aplicada à cadeia gerada, e obtenha o resto da divisão da soma por 10, $resto$. Se $resto = 0$, então o dígito de verificação deverá ser 0. Senão, o dígito de verificação deverá ser calculado pelo seu programa. Por exemplo, `calc_soma('30229065652764')` devolve 53 e, consequentemente, o dígito de verificação deverá ser 7. Assim, um número de cartão completo, gerado aleatoriamente, poderia ser 302290656527647.

Escreva a função `digito_verificacao` que recebe uma cadeia de carateres, representando um número de cartão, sem o último dígito, e devolve o dígito de verificação que lhe deverá ser acrescentado, de forma a obter um número de cartão válido. Tenha em atenção que o dígito devolvido deve ser também uma cadeia de carateres. Por exemplo, `digito_verificacao('30229065652764')` devolve a cadeia `'7'`, e não o inteiro `7`.

Utilize agora as funções acima descritas para escrever a função `gera_num_cc()`.

2.3 Nota

É importante que as suas funções sigam escrupulosamente as especificações dadas, nomeadamente no que diz respeito ao tipo de dados que recebem e que devolvem.

Ao escrever cada função, teste-a com diferentes argumentos até se convencer que esta está correcta. Após uma função escrita e testada deverá escrever as funções seguintes.

3 Classificação

A avaliação da execução será feita através de um sistema automático para entrega de projectos designado Mooshak. Existem vários testes configurados no sistema. O tempo de execução de cada teste está limitado, bem como a memória utilizada. Só poderá efectuar uma nova submissão pelo menos 15 minutos depois da submissão anterior. Só são permitidas 10 submissões em simultâneo no sistema, pelo que uma submissão poderá ser recusada se este limite for excedido. Nesse caso tente mais tarde.

Os testes considerados para efeitos de avaliação podem incluir ou não os exemplos disponibilizados no enunciado, além de um conjunto de testes adicionais. O facto de um projecto completar com sucesso os exemplos fornecidos no enunciado não implica, pois, que esse projecto esteja totalmente correcto, pois o conjunto de exemplos fornecido não é exaustivo. É da responsabilidade de cada grupo garantir que o código produzido está correcto.

Não será disponibilizado qualquer tipo de informação sobre os casos de teste utilizados pelo sistema de avaliação automática. Os ficheiros de teste usados na avaliação do projecto serão disponibilizados na página da disciplina após a data de entrega.

A nota do projecto será baseada nos seguintes aspectos:

1. Execução correcta (70%).

   Esta parte da avaliação é feita recorrendo a um programa de avaliação automática que sugere uma nota face aos vários aspectos considerados.

2. Facilidade de leitura, nomeadamente abstracção procedimental, nomes bem escolhidos, qualidade (e não quantidade) dos comentários e tamanho das funções (25%).

3. Estilo de programação (5%).

4 Condições de realização e prazos

A entrega do 1º projecto será efectuada exclusivamente por via electrónica. Deverá submeter o seu projecto através do sistema Mooshak, até às **23:59** do dia **24 de Outubro de 2014**. Projectos em atraso não serão aceites seja qual for o pretexto.

Deverá submeter um único ficheiro com extensão .py contendo todo o código do seu projecto. O ficheiro de código deve conter em comentário, na primeira linha, os números e os nomes dos alunos do grupo, bem como o número do grupo.

No seu ficheiro de código não devem ser utilizados caracteres acentuados ou qualquer carácter que não pertença à tabela ASCII. Isto inclui comentários e cadeias de caracteres. Programas que não cumpram este requisito serão penalizados em três valores.

Duas semanas antes do prazo, serão publicadas na página da cadeira as instruções necessárias para a submissão do código no Mooshak. Apenas a partir dessa altura será possível a submissão por via electrónica. Nessa altura serão também fornecidas a cada um as necessárias credenciais de acesso. Até ao prazo de entrega poderá efectuar o número de entregas que desejar, sendo utilizada para efeitos de avaliação a última entrega efectuada. Deverá portanto verificar cuidadosamente que a última entrega realizada corresponde à versão do projecto que pretende que seja avaliada. Não serão abertas excepções.

Pode ou não haver uma discussão oral do trabalho e/ou uma demonstração do funcionamento do programa (será decidido caso a caso).

Projectos iguais, ou muito semelhantes, serão penalizados com a reprovação na disciplina. O corpo docente da cadeira será o único juiz do que se considera ou não copiar num projecto.