Instituto Federal da Paraíba – Campus Cajazeiras

Segurança de Dados – 2019.1

Equipe: Eduardo Lucas Crispim de Castro, Isleimar de Souza Oliveira, José David Emanoell Feitoza Braga

Atividade colaborativa – criação de uma *wordlist*

1. Descrição da atividade

Considerando os conjuntos A={0,1,2,3,4,5,6,7}, B={a,e,i,o,u} e C={b,c,d,f,g,h,j,l,m,n,p,q,r,s,t,v,x,z,k,y}, crie um algoritmo que crie palavras, sem repetição, formadas por:

- 3 pares de [consoantes(C) + vogais (B)];
- 2 números (A);

Exemplos: bacana01, gelada77, pikaxu12, pepino32…

Além disso, responda as seguintes perguntas:

1. Quantas palavras são possíveis de serem criadas?
2. Como produzir aleatoriedade (*shuffle*) de uma lista ordenada?

Restrição: Limitar a *wordlist* a 10 mil palavras, aleatórias e não repetidas.

2. Respostas

1. Quantas palavras são possíveis de serem criadas?

R: Através da análise combinatória e das condições descritas no ponto 1 deste documento, percebemos que temos 20 consoantes, 5 vogais e 8 números e que a regra de composição deve ser uma consoante seguida de uma vogal, logo temos 20 * 5 = 100. O problema descreve 3 pares de consoantes seguidos de vogal, logo 100^3 = 1.000.000, por fim, os dois últimos dígitos devem ser números, então 1.000.000 * 8 * 8 = 64.000.000 de possibilidades de palavras.

2. Como produzir aleatoriedade (*shuffle*) de uma lista ordenada?

R: aleatoriedade imprime a quebra de ordem e padrões determinísticos das mesmas, implicando na imprevisibilidade de

elementos, em se tratando de recursos computacionais e linguagens de programação, as linguagens e seus módulos proveem soluções prontas, expressas em funções ou métodos, para a geração de números aleatórios onde o intervalo que os define é determinado pelo programador (ou usuário, dependendo do domínio de aplicação) e os números gerados aleatoriamente podem ser utilizados como índices em listas ordenadas para recuperação de dados armazenados nestas listas.

3. O algoritmo

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon May  6 15:39:31 2019

@author: isleimar
"""

import random

#Lista de caracteres possíveis
consoantes=['b','c','d','f','g','h','j','l','m','n','p','q','r','s','t','v','x','z','k','y']
vogais=['a','e','i','o','u']
limAfa=[len(consoantes)-1,len(vogais) -1]
limNum=[7,7]
wordlist = []

#Configuração da palavra
palavra=[[0,0],[0,0],[0,0],[0,0]]
limites=[limAfa,limAfa,limAfa,limNum]

#Função para montar a saída da sílaba
def saidaSilaba(s):
    return consoantes[s[0]] + vogais[s[1]]

#Função para montar a saída da palavra
def saidaPalavra(p):
    tamPalavra = len(palavra)-1
    r = ''
    for j in range(tamPalavra):
        r = r + saidaSilaba(p[j])
    r = r +  str(p[tamPalavra][0]) + str(p[tamPalavra][1])
    return r

def aletSilaba(m,n):
    m[0]= random.randint(0,n[0])
    m[1]= random.randint(0,n[1])
```

```
def aleatorio():
    tamPalavra = len(palavra)
    for j in range(tamPalavra):
        aletSilaba(palavra[j],limites[j])

def existe(p):
    for j in wordlist:
        if j == p:
            print(p)
            return True
    return False

cont = True
while len(wordlist) < 10000:
    aleatorio()
    s = saidaPalavra(palavra)
    while (existe(s)):
        aleatorio()
        s = saidaPalavra(palavra)

    wordlist.append(s)

arquivo = open('wordlist.txt','w')
for j in wordlist:
    arquivo.writelines(j+'\n')
arquivo.close
print('Fim')
```

Este algoritmo cria um arquivo .txt e forma as palavras nas configurações propostas primeiramente gerando as 3 sílabas e em seguida os 2 números. Após gerar a palavra, verifica-se a sua existência no arquivo, caso já exista, a mesma é descartada e é gerada uma nova palavra, caso contrário, a palavra é escrita normalmente no arquivo. As palavras são escritas por linha no arquivo .txt até que se atinja um total de 10.000 linhas.

4. Exemplo

Para evitar a inserção de 10.000 linhas neste documento, é mais cômodo verificar o arquivo “wordlist.txt” localizado na raiz do projeto.