```
 1 import os
 2
 3 import numpy as np
 4 from PIL import Image
 5 import torch
 6 import string
 7 from f0701_662965513 import LSTM
 8 import random
 9
10 device = torch.device("cpu")
11 model = LSTM().to(device)
12 model.load_state_dict(torch.load('
   f0702_662965513_Bartoletti.ZZZ'))
13 model.eval()
14
15 alpha_dict = {k: ord(k)-96 for k in string.
   ascii_lowercase}
16 alpha_dict['\n'] = 0
17 alpha_dict_rev = {n:k for k,n in zip(alpha_dict.keys
   (),alpha_dict.values())}
18
19
20 #Generation of 20 names with letter specified in
   input
21 lettera = input('Inserisci la lettera iniziale: ')
22
23 list_names=[]
24 for _ in range(20):
25     initial_l = np.zeros((27))
26     initial_l[alpha_dict[lettera]] = 1
27     initial_cod = [initial_l.tolist()]  # list
28     feed_mod = torch.as_tensor(initial_cod).reshape(1
   , 1, 27).type(torch.FloatTensor)
29
30     name=[]
31     name.append(lettera)
32     current_letter = initial_cod
33     i=0
34     for i in range(11):
35         prob = model(feed_mod).detach().numpy()[0][-1
   ]
```

```
36          prob_letters = model(feed_mod).detach().numpy
   ()[0][-1].argsort()[::-1]
37          letter_index = random.choices(prob_letters[0:
   3],weights=[prob[prob_letters[0]],prob[prob_letters[1
   ]],prob[prob_letters[2]]])
38          letter = alpha_dict_rev[letter_index[0]]
39          st = np.zeros((27))
40          st[letter_index] = 1
41          if letter != '\n':
42              name.append(letter)
43              st = st.tolist()
44              current_letter.append(st)
45              feed_mod = torch.as_tensor(current_letter
   ).reshape(1, i+2, 27).type(torch.FloatTensor)
46          else:
47              break
48
49      list_names.append(name)
50  for el in list_names:
51      str=""
52      print(str.join(el))
53
```