program that can perform a letter frequency attack on any monoalphabetic substitution cipher without human intervention. Your software should produce possible plaintexts in rough order of likelihood. It would be good if your user interface allowed the user to specify “give me the top 10 possible plaintexts.”

import random, math

from collections import Counter

EN\_FREQ = {'E':12.7,'T':9.1,'A':8.2,'O':7.5,'I':7.0,'N':6.7,'S':6.3,'H':6.1,'R':6.0,'D':4.3,'L':4.0,'C':2.8,'U':2.8,'M':2.4,'W':2.3,'F':2.2,'G':2.0,'Y':2.0,'P':1.9,'B':1.5,'V':1.0,'K':0.8,'J':0.2,'X':0.2,'Q':0.1,'Z':0.1}

COMMON = ["the","and","to","of","a","in","that","is","was","he","for","it","as","with","his","on","be","at"]

def clean(s): return ''.join(c for c in s.upper() if c.isalpha() or c==' ')

def apply\_key(txt, key): return ''.join(key.get(c,c) for c in txt.upper())

def freq\_score(txt):

c=Counter(c for c in txt if c.isalpha()); n=sum(c.values()); sc=0

for k,v in c.items():

e=EN\_FREQ.get(k,0)/100\*n

sc -= (v-e)\*\*2/(e+1)

return sc

def word\_bonus(txt):

s=txt.lower(); b=0

for w in COMMON: b+=s.count(w)\*len(w)

return b

def fitness(t): return freq\_score(t)+3\*word\_bonus(t)

def rand\_key():

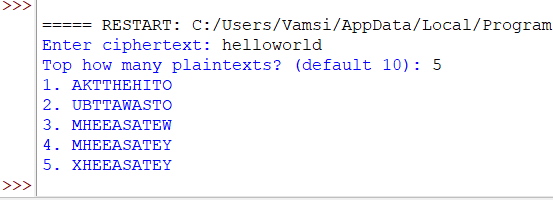
a=list("ABCDEFGHIJKLMNOPQRSTUVWXYZ"); b=a.copy(); random.shuffle(b)

return dict(zip(a,b))

def swap\_key(k):

k=k.copy(); a,b=random.sample(list(k.keys()),2)

k[a],k[b]=k[b],k[a]; return k



def attack(cipher, top=10, loops=5000):

key=rand\_key(); best=apply\_key(cipher,key); bests=f=fitness(best)

out=[(f,best)]

for \_ in range(loops):

k2=swap\_key(key); p2=apply\_key(cipher,k2); f2=fitness(p2)

if f2>f or random.random()<math.exp((f2-f)/max(1,loops\*0.001)):

key,f,best=k2,f2,p2

if f2>bests:

bests=f2; out.append((f2,best))

out.sort(reverse=True); seen=set(); res=[]

for s,t in out:

if t not in seen:

seen.add(t); res.append(t)

if len(res)>=top: break

return res

cipher=input("Enter ciphertext: ")

top\_n=int(input("Top how many plaintexts? (default 10): ") or 10)

results=attack(cipher,top\_n)

for i,r in enumerate(results,1):

print(f"{i}. {r}")]}")eview}")