é necessário alterar diretamente no código os valores

In [1]:
```
print("Insira a chave: ")
key = 3
print("Insira o texto: ")
texto = "A distinção crítica do SOA é que nenhum serviço pode nomear ou acessar dad
```

Valor da key

texto →

```
Insira a chave:
Insira o texto:
```

In [2]:
```
#Criação dos maps
alfabeto = "abcdefghijklmnopqrstuvwxyz"
n = 0
letra_p_numero = {}
numero_p_letra = {}
for letra in alfabeto:
    letra_p_numero[n] = letra
    numero_p_letra[letra] = n
    n+= 1
```

mapeando as letras e números

a = 1 | 1 = a
b = 2 | 2 = b
c = 3 | 3 = c
... | ...

In [3]:
```
#Percorrendo o texto e criptografando
texto_criptografado = ""
for letra in texto:
    letra = letra.lower()
    if letra in alfabeto:
        print(letra_p_numero[(numero_p_letra[letra] + key) % 26], end='')
        texto_criptografado += letra_p_numero[(numero_p_letra[letra] + key) % 26]
    else:
        print(letra, end='')
        texto_criptografado += letra
```

key = 3
texto = "A distinção ..."

print (letra_p_numero[(numero_p_letra[letra] + key) % 26], end='']

(a = 1 + 3) % 26 = 4

4 = d

O módulo é importante para que os valores não passem do limite "transbordem"

Resultado após criptografia com chave 3 ou 'c'

d glvwlqçãr fuíwlfd gr vrd é txh qhqkxp vhuylçr srgh qrphdu rx dfhvvdu gdgrv gh txdo txhu rxwur vhuylçr; hoh srgh, dshqdv, idchu uhtxlvlçõhv sru gdgrv dwudyév gh dslv ha whuqdv. vh r gdgr txh hoh txhu qãr hvwá glvsrqíyho dwudyév gd dsl, hqwãr vlqwr pxlwr . shufhed txh r vrd qãr frlqflgh frp r prghor wudglflrqdo gh fdpdgdv gh vriwzduh, qr txdo fdgd fdpdgd pdlv hohydgd é frqvwuxígd, gluhwdphqwh, d sduwlu gdv fdpdgdv lqihul ruhv lphgldwdv, frpr hp xp vriwzduh hp vlor. r vrd hqyroyh pxlwdv idwldv dwudyév gh pxlwdv fdpdgdv h hvvdv idwldv vãr frqhfwdgdv hqwuh vl sdud irupdu xp vhuylçr. hqtxdq wr r vrd, jhudophqwh, vljqlilfd xp srxfr pdlv gh wudedokr frpsdudgr à frqvwuxçãr gh xp vhuylçr hp vlor, r uhwruqr é wuhphqgd- phqwh uhxwlolcáyho. rxwud ydqwdjhp gr vrd hvwá uhodflrqdgd àv dslv hasoíflwdv txh wruqdp rv whvwhv pdlv iáfhlv. halvwhp gxdv g hvydqwdjhqv dpsodphqwh dfhlwdv sdud vrd. hp sulphlur oxjdu, fdgd lqyrfdçãr gh xp vhu ylçr hqyroyh r fxvwr pdlru gh shufruuhu rv qíyhlv pdlv surixqgrv gd slokd gh surwrfr orv gh uhgh h, sruwdqwr, ká xp lpsdfwr qhjdwlyr qr ghvhpshqkr gr vrd. hp vhjxqgr oxj du, phvpr txh vhmd pxlwr suryáyho txh xp vlvwhpd hp vlorv yhqkd frpsohwdphqwh dedlar txdqgr ká xpd idokd, hqjhqkhlurv gh vriwzduh txh vh xwlolcdp gr vrd ghyhp olgdu wdpe ép frp r fdvr frpsolfdgr gh idokdv sdufldlv. dojxpdv sduwhv gr vlvwhpd srghp idokdu, hqtxdqwr txh rxwudv frqwlqxdp ixqflrqdqgr qrupdophqwh. vrd idc frp txh r sodqhmdphqw r gh frqildelolgdgh vhmd xp srxfr pdlv ghvdildgru. d hqruph ydqwdjhp gr vrd é d srvv lelolgdgh gh xvdu shtxhqrv vhuylçrv frqvwuxígrv frp vxfhvvr, hp sduwh srutxh srghprv xwlolcdu ghvhqyroylphqwr ájlo sdud frqvwuxí-orv h, srvwhulruphqwh, frpelqá-orv sdud frqvwuxlu vhuylçrv pdlruhv. lqiholcphqwh, vh r suhvlghqwh redpd wlyhvvh olgr hvwh fd síwxor d whpsr gh pdqgdu xp hpdlo, dr hvwlor ehcrv, drv frqwudwdqwhv gr dfd dqwhv gh oh vhu odqçdgr, d klvwóuld srghuld vh ohpeudu ghoh frpr xp suhvlghqwh pdlv ehp vxfhg lgr. sdud rv ixwxurv suhvlghqwhv hqwuh qrvvrv ohlwruhv: xp krphp suhyhqlgr ydoh sru grlv! doép gd qrvvd rslqlãr vreuh r udlov vhu whfqlfdphqwh vxshulru sdud ghvhqyroylp hqwr ájlo h r vddv, r uxeb h r udlov vãr dpsodphqwh xvdgrv. sru hahpsor, r uxeb dsdu hfh iuhtxhqwhphqwh hqwuh dv 10 olqjxdjhqv gh surjudpdçãr pdlv srsxoduhv. xp dsolfdwl yr vddv, pxlwr frqkhflgr, dvvrfldgr dr udlov é r wzlwwhu, txh frphçrx frpr xp dsolfd wlyr udlov hp 2006 h fuhvfhx gh 20.000 wzhhwv sru gld, hp 2007, sdud 200.000.000 wzh hwv hp 2011, shuírgr gxudqwh r txdo yáulrv rxwurv dufderxçrv vxevwlwxíudp sduwhv gr vlvwhpd. vh yrfê dlqgd qãr hvwá idploldulcdgr frp r uxeb rx r udlov, lvvr okh gá xpd fkdqfh sdud txh sudwltxh xpd kdelolgdgh gh hqjhqkduld gh vriwzduh lpsruwdqwh: xvh d ihuudphqwd fruuhwd sdud r wudedokr, phvpr txh lvvr vljqliltxh dsuhqghu xpd qryd ihuu dphqwd rx xpd qryd olqjxdjhp! gh idwr, xpd fdudfwhuívwlfd dwudwlyd gd frpxqlgdgh udl ov é txh vhxv frqwulexlqwhv phokrudp, fruultxhludphqwh, d surgxwlylgdgh dr lqyhqwduh p qrydv ihuudphqwdv sdud dxwrpdwlcdu wduhidv txh hudp dqwhulruphqwh ihlwdv gh irupd pdqxdo. uhsduh txh dwxdolcdçõhv iuhtxhqwhv gr vddv – ghylgr dr idwr gh whu dshqdv xp d úqlfd fósld gr vriwzduh – vh dolqkdp, shuihlwdphqwh, frp r flfor gh ylgd ájlo. frq vhtxhqwhphqwh, d dpdcrq, r hedb, r idfherrn, r jrrjoh, h rxwurv suryhgruhv vddv, ghs hqghp gr flfor gh ylgd ájlo, h frpsdqkldv gh vriwzduh wudglflrqdlv, frpr d plfurvriw , hvwãr, judgdwlydphqwh, xvdqgr péwrgrv ájhlv qr ghvhqyroylphqwr gh vhxv surgxwrv. r surfhvvr ájlo é xp hafhohqwh frpsdqkhlur sdud dv pxgdqçdv gh qdwxuhcd uáslgd gdv dso lfdçõhv vddv. dvvlp frpr rv urpdqflvwdv hvshudp txh vxdv fuldçõhv vhmdp olgdv r vxil flhqwh sdud vhuhp urwxodgdv frpr xp foávvlfr – txh sdud xp olyur vãr 100 dqrv! –, rv hqjhqkhlurv gh vriwzduh ghyhp hvshudu txh vxdv fuldçõhv wdpeép vhmdp gxudgrxudv. fod ur, r vriwzduh whp ydqwdjhqv vreuh rv olyurv srlv srghp vhu phokrudgrv dr orqjr gr w hpsr. gh idwr, xp vriwzduh gh orqjd ylgd suhvvxsõh txh rxwurv r pdqwhqkdp h r phokru hp, ghladqgr rv fuldgruhv gr fógljr ruljlqdo olyuh gh reuljdçõhv. lvvr qrv ohyd d do jxqv whuprv txh xvduhprv dr orqjr gr olyur. r whupr fógljr ohjdgr uhihuh-vh d xp vri wzduh txh, dshvdu gh vxd lgdgh, frqwlqxd vhqgr xvdgr srlv dwhqgh àv qhfhvvlgdghv grv folhqwhv. 60% grv fxvwrv gh pdqxwhqçãr gh vriwzduh vãr sdud dglflrqdu xpd qryd ixqfl rqdolgdgh dr vriwzduh ohjdgr, frqwud dshqdv 17% sdud uhvroyhu rv huurv gh surjudpdçã r; sruwdqwr, r vriwzduh ohjdgr é xp vriwzduh ehp vxfhglgr. r whupr “ohjdgr” whp xpd frqrwdçãr qhjdwlyd srlv lqglfd txh r fógljr srvvxl glilfxogdgh hp hyroxlu ghylgr à g hvhohjâqfld gh vhx surmhwr rx xvr gh xpd whfqrorjld dqwltxdgd. hp frqwudvwh frp r fó gljr ohjdgr, xwlolcdprv r whupr fógljr ehor sdud uhsuhvhqwdu xp fógljr gxudgrxur txh

vhmd iáflo gh hyroxlu. r slru fdvr qãr é r fógljr ohjdgr pdv vlp fógljr gh fxuwd gxu dçãr lqhvshudgd, txh é euhyhphqwh ghvfduwdgr srlv qãr dwhqgh àv qhfhvvlgdghv grv fol hqwhv. ydprv ghvwdfdu dojxqv hahpsorv txh ohydp dr fógljr ehor frp r ífrqh gd prqd o lvd. gd phvpd irupd, luhprv ghvwdfdu r whawr txh ohyd dr fógljr ohjdgr dwudyév gr íf rqh gr áedfr txh, dshvdu gh gxudgrxur, é xp glvsrvlwlyr gh fáofxor txh qãr whp pxgdg r pxlwr. qrv suóalprv fdsíwxorv, prvwuduhprv hahpsorv wdqwr gh fógljr ohjdgr txdqwr gh fógljr ehor txh, hvshudprv, luãr lqvsluá-or d idchu ghvljqv txh vhmdp pdlv vlpsoh v gh vhuhp hyroxígrv.

In [4]:
```
#Descriptografando por meio da força bruta.
from langdetect import detect

for i in range(0, 26, 1):
    texto_teste = ""
    for j in range(100):
        letra = texto_criptografado[j].lower()
        if letra in alfabeto:
            texto_teste += letra_p_numero[(numero_p_letra[letra] - i + 25) % 26]
        else:
            texto_teste += letra
    idioma = detect(texto_teste)
    if idioma == "pt":
        print(alfabeto[i])
        for letra in texto_criptografado:
            if letra in alfabeto:
                print(letra_p_numero[(numero_p_letra[letra] - i + 25) % 26], end =
            else:
                print(letra, end = '')
        break
```

biblioteca que descobre a língua do texto

26 letras

realizei a análise apenas dos 100 primeiros characteres

verificando o idioma dos 100 characteres,

processo inverso da criptografia, somo-se 25 para prevenir números negativos.

se o idioma == português será mostrado a chave utilizada e o texto descriptografado.

chave utilizada

c

a distinção crítica do soa é que nenhum serviço pode nomear ou acessar dados de qual quer outro serviço; ele pode, apenas, fazer requisições por dados através de apis ex ternas. se o dado que ele quer não está disponível através da api, então sinto muito . perceba que o soa não coincide com o modelo tradicional de camadas de software, no qual cada camada mais elevada é construída, diretamente, a partir das camadas inferi ores imediatas, como em um software em silo. o soa envolve muitas fatias através de muitas camadas e essas fatias são conectadas entre si para formar um serviço. enquan to o soa, geralmente, significa um pouco mais de trabalho comparado à construção de um serviço em silo, o retorno é tremenda- mente reutilizável. outra vantagem do soa está relacionada às apis explícitas que tornam os testes mais fáceis. existem duas d esvantagens amplamente aceitas para soa. em primeiro lugar, cada invocação de um ser viço envolve o custo maior de percorrer os níveis mais profundos da pilha de protoco los de rede e, portanto, há um impacto negativo no desempenho do soa. em segundo lug ar, mesmo que seja muito provável que um sistema em silos venha completamente abaixo quando há uma falha, engenheiros de software que se utilizam do soa devem lidar tamb ém com o caso complicado de falhas parciais. algumas partes do sistema podem falhar, enquanto que outras continuam funcionando normalmente. soa faz com que o planejament o de confiabilidade seja um pouco mais desafiador. a enorme vantagem do soa é a poss ibilidade de usar pequenos serviços construídos com sucesso, em parte porque podemos utilizar desenvolvimento ágil para construí-los e, posteriormente, combiná-los para construir serviços maiores. infelizmente, se o presidente obama tivesse lido este ca pítulo a tempo de mandar um email, ao estilo bezos, aos contratantes do aca antes de le ser lançado, a história poderia se lembrar dele como um presidente mais bem suced ido. para os futuros presidentes entre nossos leitores: um homem prevenido vale por dois! além da nossa opinião sobre o rails ser tecnicamente superior para desenvolvim ento ágil e o saas, o ruby e o rails são amplamente usados. por exemplo, o ruby apar ece frequentemente entre as 10 linguagens de programação mais populares. um aplicati vo saas, muito conhecido, associado ao rails é o twitter, que começou como um aplica tivo rails em 2006 e cresceu de 20.000 tweets por dia, em 2007, para 200.000.000 twe ets em 2011, período durante o qual vários outros arcabouços substituíram partes do sistema. se você ainda não está familiarizado com o ruby ou o rails, isso lhe dá uma chance para que pratique uma habilidade de engenharia de software importante: use a ferramenta correta para o trabalho, mesmo que isso signifique aprender uma nova ferr amenta ou uma nova linguagem! de fato, uma característica atrativa da comunidade rai ls é que seus contribuintes melhoram, corriqueiramente, a produtividade ao inventare m novas ferramentas para automatizar tarefas que eram anteriormente feitas de forma manual. repare que atualizações frequentes do saas – devido ao fato de ter apenas um a única cópia do software – se alinham, perfeitamente, com o ciclo de vida ágil. con sequentemente, a amazon, o ebay, o facebook, o google, e outros provedores saas, dep endem do ciclo de vida ágil, e companhias de software tradicionais, como a microsoft , estão, gradativamente, usando métodos ágeis no desenvolvimento de seus produtos. o processo ágil é um excelente companheiro para as mudanças de natureza rápida das apl icações saas. assim como os romancistas esperam que suas criações sejam lidas o sufi ciente para serem rotuladas como um clássico – que para um livro são 100 anos! –, os engenheiros de software devem esperar que suas criações também sejam duradouras. cla ro, o software tem vantagens sobre os livros pois podem ser melhorados ao longo do t empo. de fato, um software de longa vida pressupõe que outros o mantenham e o melhor em, deixando os criadores do código original livre de obrigações. isso nos leva a al guns termos que usaremos ao longo do livro. o termo código legado refere-se a um sof tware que, apesar de sua idade, continua sendo usado pois atende às necessidades dos clientes. 60% dos custos de manutenção de software são para adicionar uma nova funci onalidade ao software legado, contra apenas 17% para resolver os erros de programaçã o; portanto, o software legado é um software bem sucedido. o termo “legado” tem uma conotação negativa pois indica que o código possui dificuldade em evoluir devido à d eselegância de seu projeto ou uso de uma tecnologia antiquada. em contraste com o có

digo legado, utilizamos o termo código belo para representar um código duradouro que seja fácil de evoluir. o pior caso não é o código legado mas sim código de curta dur ação inesperada, que é brevemente descartado pois não atende às necessidades dos cli entes. vamos destacar alguns exemplos que levam ao código belo com o ícone da mona l isa. da mesma forma, iremos destacar o texto que leva ao código legado através do íc one do ábaco que, apesar de duradouro, é um dispositivo de cálculo que não tem mudad o muito. nos próximos capítulos, mostraremos exemplos tanto de código legado quanto de código belo que, esperamos, irão inspirá-lo a fazer designs que sejam mais simple s de serem evoluídos.

In [5]:
```
#Descriptografando por Análise de frequência
import matplotlib.pyplot as plt

#Frequências bases
frequencias_base = {'a': 14.63, 'b': 1.04, 'c': 3.88, 'd': 4.99, 'e': 12.57,
                    'f': 1.02, 'g': 1.3, 'h': 1.28, 'i': 6.18, 'j': 0.4,
                    'k': 0.02, 'l': 2.78, 'm': 4.74, 'n': 5.05, 'o': 10.73,
                    'p': 2.52, 'q': 1.20, 'r': 6.53, 's': 7.81, 't': 4.34,
                    'u': 4.63, 'v': 1.67, 'w': 0.01, 'x': 0.21, 'y': 0.01,
                    'z': 0.47}

frequencias_criptografado = {'a': 0.0, 'b': 0.0, 'c': 0.0, 'd': 0.0, 'e': 0.0,
                             'f': 0.0, 'g': 0.0, 'h': 0.0, 'i': 0.0, 'j': 0.0,
                             'k': 0.0, 'l': 0.0, 'm': 0.0, 'n': 0.0, 'o': 0.0,
                             'p': 0.0, 'q': 0.0, 'r': 0.0, 's': 0.0, 't': 0.0,
                             'u': 0.0, 'v': 0.0, 'w': 0.0, 'x': 0.0, 'y': 0.0,
                             'z': 0.0}
count = 0
for letra in texto_criptografado:
        letra = letra.lower()
        if letra in alfabeto:
            frequencias_criptografado[letra] += 1
            count += 1

for letra in alfabeto:
    frequencias_criptografado[letra] = (frequencias_criptografado[letra]/count)*100

# Extraindo dados para plotagem
letras = list(frequencias_base.keys())
frequencia_base = list(frequencias_base.values())
frequencia_criptografado = list(frequencias_criptografado.values())

# Plotando os gráficos
plt.figure(figsize=(12, 6))

# Frequências base (azul)
plt.plot(letras, frequencia_base, marker='o', color='blue', label='Frequência Base'

# Frequências do texto criptografado (vermelho)
plt.plot(letras, frequencia_criptografado, marker='o', color='red', label='Frequênc

# Configurações do gráfico
plt.title("Análise de Frequência de Letras")
plt.xlabel("Letras")
plt.ylabel("Frequência (%)")
plt.legend()
```

retirada de um trabalho da UFRJ "decifrando textos em português, GTA/UFRJ"

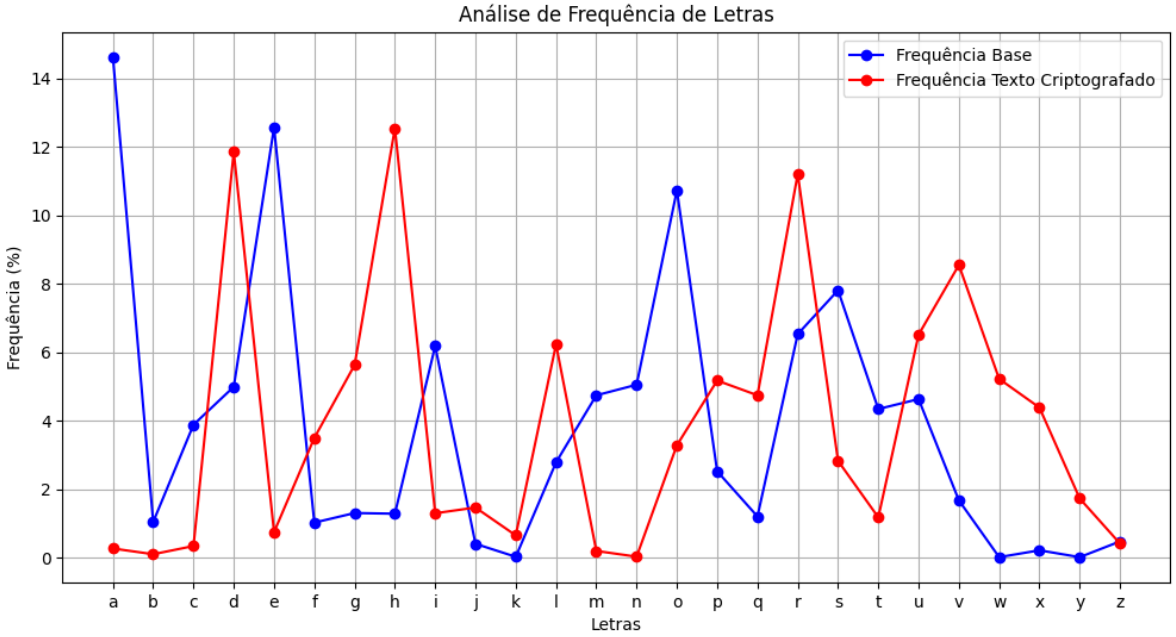
sempre que a letra ocorre somamos 1 em

divide-se pelo total de letras e multiplica-se por 100 para descobrir a frequência.

PLOT

```
plt.grid(True)

# Exibindo o gráfico
plt.show()
```



Tabela

In [11]:
```
import pandas as pd
frequencias_base_ordenado = dict(sorted(frequencias_base.items(), key=lambda item:
frequencias_criptografado_ordenado = dict(sorted(frequencias_criptografado.items(),
df = pd.DataFrame({
    'Letra_base': list(frequencias_base_ordenado.keys()),
    'Frequência Base': list(frequencias_base_ordenado.values()),
    'Letra_cript': list(frequencias_criptografado_ordenado.keys()),
    'Frequência Criptografada': list(frequencias_criptografado_ordenado.values())
})

# Exibindo a tabela
print(df)
```

	Letra_base	Frequência Base	Letra_cript	Frequência Criptografada
0	w	0.01	n	0.023946
1	y	0.01	b	0.095785
2	k	0.02	m	0.191571
3	x	0.21	a	0.263410
4	j	0.40	c	0.335249
5	z	0.47	z	0.407088
6	f	1.02	k	0.646552
7	b	1.04	e	0.742337
8	q	1.20	t	1.173372
9	h	1.28	i	1.293103
10	g	1.30	j	1.460728
11	v	1.67	y	1.724138
12	p	2.52	s	2.825670
13	l	2.78	o	3.280651
14	c	3.88	f	3.496169
15	t	4.34	x	4.382184
16	u	4.63	q	4.741379
17	m	4.74	p	5.172414
18	d	4.99	w	5.220307
19	n	5.05	g	5.627395
20	i	6.18	l	6.226054
21	r	6.53	u	6.513410
22	s	7.81	v	8.548851
23	o	10.73	r	11.206897
24	e	12.57	d	11.853448
25	a	14.63	h	12.547893

In [ ]:

Considerações Finais:

Força bruta

- Muito provavelmente, retornará a resposta correta. (+)
- Demorado para textos grandes e/ou com um grande range para as chaves (−)
- Bom para textos curtos (+)

Análise de Frequência

- Ruim para textos pequenos (−)
- Quanto mais longo o texto melhor, pois assim as frequências do texto criptografado se tornarão mais próximos ao da frequência base.