Laboration

March 7, 2024

1 Laboration

Kryptering 1, hösten 2023

Namn: Viktor Listi

[1]:
```
# Laddar användbar fil genom att ställa markören i sällen och tryck Shift+Enter.
load('kryptering1.sage')
```

```
***  kryptering1.sage: Funktioner för kursen Kryptering 1 (SageMath)  ***
```

1.1 Uppgift 1

[2]:
```
klartext = 'imponerande'
# Förskjutning med k = 17
a = Caesarchiffer(klartext, 17)
print('Förskjutning({}) = {}\n'.format(klartext, a.upper()))

#Monoalfabetiskt substitutionskrypto
bKey = 'dqkåaäuröshjiebcpxvfogylmztn'
b = substitutionskrypto(klartext, bKey)
print('Monoalfa subkrypto({}) = {}'.format(klartext, b.upper()))

#Affint med (a,b) = (13,9)
c = affint_krypto(klartext, 13, 9)
print('\nAffint({}) = {}'.format(klartext, c.upper()))

#Vignere
dKey = 'gömd'
d = Vigenerekrypto(klartext, dKey)
print('\nVignere({}) = {}'.format(klartext, d.upper()))

#Transposition
eKey = [2,0,3,1]
e = transpositionskrypto(klartext, eKey)
print('\nTransposition({}) = {}'.format(klartext, e.upper()))

#PlayFair
fKey = 'drlxayszhkepbciqufgmvontw'
```

```
f = Playfair(klartext, fKey)
print('\nPlayFair({}) = {}'.format(klartext, f.upper()))

#ADFGVX
gKey1 = [1,4,3,0,2]
gKey2 = 'g6u7aq8rdshjix1f2plw5mt0eb4v9ocykzn3'
g = ADFGVX(klartext, gKey2, gKey1)
print('\nADFGVX({}) = {}'.format(klartext, g))

#Enigma
hKey1 = [('A', 'L'), ('D', 'Q'), ('K', 'N'), ('M', 'S'), ('U', 'X')]
hKey2 = ['II', 'IV', 'I']
hKey3 = 'SYD'
h = Enigma(klartext.upper(), hKey1, hKey2, hKey3)
print('\nEnigma({}) = {}'.format(klartext.upper(), h))

#Hillkrypto
iKey = matrix(Zmod(28), [[5,2,18],[19,1,9],[17,16,13]])
i = Hillkrypto(klartext, iKey, typ = 'H')
print('\nHill({}) = {}'.format(klartext, i.upper()))

#DES
jMessageBinary = text_till_block64(klartext)
jKey =␣
 ↪[1,1,1,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,1,0,0,
j = DES_ECB(jMessageBinary, jKey)
print('\nDES({}):'.format(klartext))
for block in block64_till_text(j, typ = 'hex') :
    print(block)

#RSA
kKeyE =  1330643366620853071
kKeyN = 31349340639199591501417973539664 41
kMessage = sve(klartext)
kMessageModify1 = [num + 1 for num in kMessage]
kMessageModify2= [f'{num:02}' for num in kMessageModify1]
kMessageModify3 = int(''.join(map(str, kMessageModify2)))
k = power_mod(kMessageModify3, kKeyE, kKeyN)
print('\nRSA({}) = {}'.format(klartext, k))
```

```
Förskjutning(imponerande) = ÅBEDCVGRCUV

Monoalfa subkrypto(imponerande) = ÖICBEAXDEÅA

Affint(imponerande) = BÅIYKFGJKUF

Vignere(imponerande) = OLÖRTDBDTCQ
```

```
Transposition(imponerande) = PREINNOAMED

PlayFair(imponerande) = MWURVBLDVLCD

ADFGVX(imponerande) = AVAXAFVAFGXDDFXVVVGXDV

Enigma(IMPONERANDE) = OEKABJKNYAU

Hill(imponerande) = ÄTTAHXLUKÖHJ

DES(imponerande):
572668CA12863B59
FB0069FC5C04C76F

RSA(imponerande) = 492236175884835670950267908035895
```

[3]:
```
Kryptogram1 = 'LIGSLAAÄTLMMTNUOIRLREDGM' #Transposition
Kryptogram2 = 'LDEOBMRNJKMMLXYBRJQFPHQZPYIRD' #Enigma
Kryptogram3 = '32107833669138743416991214827014308'  #RSA
Kryptogram4 = 'RBBJDQDEÅBBJDQDEÅGAIHDEIDEJÖFDCZ' #monoalfabetiskt sub
Kryptogram5 = 'JÖBSTFKÄJJSAXLCEÖLUHÖACÄNMLÅMCAOIBTT' #Hill
Kryptogram6 = 'FMJÖJTIBYKFGÖKLHÖFGQFÅMBDJÅFUUFMJKUFK' #Affin
Kryptogram7 = '1DCDB9E4E6EA1049 83B2AE7D8E20B92B 63C8B4251AFAC981␣
 ↪A6F5AED397A81135' #DES
Kryptogram8 = 'IPGCÅCYVCPGÖRHIRU'  #förskjutning
Kryptogram9 = 'VDAFVFDDXVXVGXGAVDFDGAVX' #ADFGVX
Kryptogram10 = 'RTBLMÖPLVKÄPGSQULHGDYEBHJ' #Vignere
Kryptogram11 = 'RVTWPUBAXNYCCPLWHR' #playfair

#förskjutning Kryptogram
förskjutningsKryptogram = sve(Kryptogram8.lower())
förskjutningText = ""
for i in range(len(förskjutningsKryptogram)):
    förskjutningsKryptogram[i] = förskjutningsKryptogram[i] - 17
    förskjutningsKryptogram[i] = Mod(förskjutningsKryptogram[i],28)
    förskjutningText += Asve[förskjutningsKryptogram[i]]
print("förskjutning -",förskjutningText)

#Playfair
playfairKey = 'drlxayszhkepbciqufgmvontw'
playfairMessage = Playfair(Kryptogram11.lower(), playfairKey, metod =␣
 ↪'dekryptera')
print("PlayFair -", playfairMessage)

#Transposition
transKey = [2,0,3,1]
```

```
transMessage = transpositionskrypto(Kryptogram1, transKey, metod = 'dekryptera')
print("Transposition -",transMessage)

#affin
a = Mod(13,28)
b = Mod(9,28)
affinMessage = affint_krypto(Kryptogram6.lower(), a^(-1), -a^(-1)*b)
print("Affint -",affinMessage)

#monoalfabetiskt
monoKey = 'dqkåaäuröshjiebcpxvfogylmztn'
monoMessage = substitutionskrypto(Kryptogram4.lower(), Asve ,monoKey)
print("Monoalfabetiskt -",monoMessage)

#Vignere
vignereKey = 'gömd'
vignereKeyFinal = sve([-x % 28 for x in sve(vignereKey)])
vignereMessage = Vigenerekrypto(Kryptogram10.lower(), vignereKeyFinal)
print("Vignere -",vignereMessage)

#Hill
hillKey = matrix(Zmod(28), [[5,2,18],[19,1,9],[17,16,13]])
hillKeyInverse = hillKey.inverse()
hillMessage = Hillkrypto(Kryptogram5.lower(), hillKeyInverse, typ = 'H')
print("Hill -", hillMessage)

#Engima
enigmaKey1 = [('A', 'L'), ('D', 'Q'), ('K', 'N'), ('M', 'S'), ('U', 'X')]
enigmaKey2 = ['II', 'IV', 'I']
enigmaKey3 = 'SYD'
enigmaMessage = Enigma(Kryptogram2, enigmaKey1, enigmaKey2, enigmaKey3)
print("Enigma -",enigmaMessage)

#ADFGVX
adfgvxKey1 = [1,4,3,0,2]
adfgvxKey2 = 'g6u7aq8rdshjix1f2plw5mt0eb4v9ocykzn3'
adfgvxMessage = ADFGVX(Kryptogram9, adfgvxKey2, adfgvxKey1, metod =␣
 ↪'dekryptera')

print("ADFGVX -",adfgvxMessage)

#DES
krypto7 = ['1DCDB9E4E6EA1049', '83B2AE7D8E20B92B', '63C8B4251AFAC981',␣
 ↪'A6F5AED397A81135']
tempList = []
for b in krypto7:
    var1 = [b[i:i+2] for i in range(0,16,2)]
```

```
    var2 = [ZZ('0x' + h) for h in var1]
    var3 = [bin(d) for d in var2]
    var3 = [b[2:] for b in var3]
    var3 = [b.zfill(8)[::-1] for b in var3]
    var3 = ''.join(var3)
    var3 = [ZZ(b) for b in var3]
    tempList.append(var3)
mainKey = 8*(4*[1]+4*[0])
message = DES_ECB(tempList,mainKey, metod='dekryptering')
print("DES -", block64_till_text(message))


#RSA

x, y, t_0 = var('x y t_0')
n = 36134934063919959150141797353966441
e = 1330643366620853071
ep = euler_phi(n)

temp = solve_diophantine(e*x - ep*y == 1)
temp2 = solve([t_0 >= 0, e*temp[0] - ep*temp[1] == 1], t_0)
r = [L.subs(temp2[-1]) for L in temp]
d = int(r[0])
m = power_mod(int(Kryptogram3) ,d ,n)
m = str(m)
answer = ""
tempList = [m[x]+m[x+1] for x in range(0,len(m),2)]
for n in tempList:
    answer += Asve[int(n)-1]
print("RSA -", answer)
```

```
förskjutning - tärningenärkastad
PlayFair - dontspillthebeansx
Transposition - ALLTÄRINTEGULDSOMGLIMMAR
Affint - elakaspionerknäckerhemligameddelanden
Monoalfabetiskt - hoolabandoolabandvemkanmanlitapå
Vignere - lurigadiplomaterfixarfred
Hill - jamesbondlurarskurkarmestheladagenxx
Enigma - ENGANSKAANDEFATTIGOCHKORTTEXT
ADFGVX - thisyear2023
DES - Surt, sa räven om rönnbären.
RSA - spillintebönorna
```

1.2 Uppgift 2

[4]:
```
load('kryptogram.sage') #få ut nyckel + klartext
d = 27 #kryptogram_27

print(kryptogram_27)
decrypted = '-' * len(kryptogram_27)

listedKryptogram_27 = list(kryptogram_27)
listedDecrypted = list(decrypted)

i = 0
while i < len(listedKryptogram_27):
    if listedKryptogram_27[i] == 'f':
        listedDecrypted[i] = 'n'
    elif listedKryptogram_27[i] == 'p':
        listedDecrypted[i] = 'e'
    elif listedKryptogram_27[i] == 'b':
        listedDecrypted[i] = 'h'
    elif listedKryptogram_27[i] == 'ä':
        listedDecrypted[i] = 'a'
    elif listedKryptogram_27[i] == 'q':
        listedDecrypted[i] = 't'
    elif listedKryptogram_27[i] == 's':
        listedDecrypted[i] = 'i'
    elif listedKryptogram_27[i] == 'm':
        listedDecrypted[i] = 'l'
    elif listedKryptogram_27[i] == 'l':
        listedDecrypted[i] = 'v'
    elif listedKryptogram_27[i] == 'd':
        listedDecrypted[i] = 'd'
    elif listedKryptogram_27[i] == 'x':
        listedDecrypted[i] = 'r'
    elif listedKryptogram_27[i] == 'a':
        listedDecrypted[i] = 'm'
    elif listedKryptogram_27[i] == 'j':
        listedDecrypted[i] = 'g'
    elif listedKryptogram_27[i] == 'e':
        listedDecrypted[i] = 's'
    elif listedKryptogram_27[i] == 'o':
        listedDecrypted[i] = 'p'
    elif listedKryptogram_27[i] == 'n':
        listedDecrypted[i] = 'ä'
    elif listedKryptogram_27[i] == 'ö':
        listedDecrypted[i] = 'k'
    elif listedKryptogram_27[i] == 'c':
        listedDecrypted[i] = 'o'
```

```
    elif listedKryptogram_27[i] == 'i':
        listedDecrypted[i] = 'c'
    elif listedKryptogram_27[i] == 'k':
        listedDecrypted[i] = 'å'
    elif listedKryptogram_27[i] == 'h':
        listedDecrypted[i] = 'b'
    elif listedKryptogram_27[i] == 'y':
        listedDecrypted[i] = 'ö'
    elif listedKryptogram_27[i] == 't':
        listedDecrypted[i] = 'u'
    elif listedKryptogram_27[i] == 'g':
        listedDecrypted[i] = 'f'
    elif listedKryptogram_27[i] == 'z':
        listedDecrypted[i] = 'j'
    elif listedKryptogram_27[i] == 'u':
        listedDecrypted[i] = 'y'
    #elif listedKryptogram_27[i] == 'r':
        #listedDecrypted[i] = ''
    #elif listedKryptogram_27[i] == 'v':
        #listedDecrypted[i] = ''
    #elif listedKryptogram_27[i] == 'å':
        #listedDecrypted[i] = ''
    i+=1

print()
result_string = ''.join(listedDecrypted)
print(result_string)
print()
#f -> n  gissade
#p -> e  gissade
#b -> h   fick fram 'henne' nu
#ä -> a ?
#q -> t ? baserat på äqq --> att
#s -> i ? baserat på smm --> ill
#m -> l ?
#l -> v ? för att skapa "vill"
#d -> d ? för att skapde "hade"



#x -> r #sista meningen "han visade sig aldrig mer"
#a -> m
#j -> g
#e -> s


#o -> p #göra "rinsessan" till "prinsessan"
```

```
#n -> ä #göra "prinsessan när han var"


# göra "draken och prinsessan, det var en gång"
#ö -> k #draken och prinsessan
#c -> o
#i -> c
#k -> å

#resten är rutin arbete


#vrå -> qxz kan inte bestämmas då de aldrig förekommer i texterna


frekvensKrypto = monogramanalys(kryptogram_27) #Ger oss -> p, f, ä, x, q, e (i␣
 ↪nämnd ordning) --> e, a, r, n, t, s (i nämnd ordning)
bigramKrypto = bigramanalys(kryptogram_27) #Ger oss -> pf, dp, äf, fe, bä, xä␣
 ↪(i nämnd ordning) --> en, er ,de, an, st, ar (i nämnd ordning)
trigramKrypto = trigramanalys(kryptogram_27) #ger oss -> ädp, bäf, jpf, dxä,␣
 ↪äöp, xää (i nämnd ordning)  --> för, och, nde, and, ing, ter, den (i nämnd␣
 ↪ordning)
```

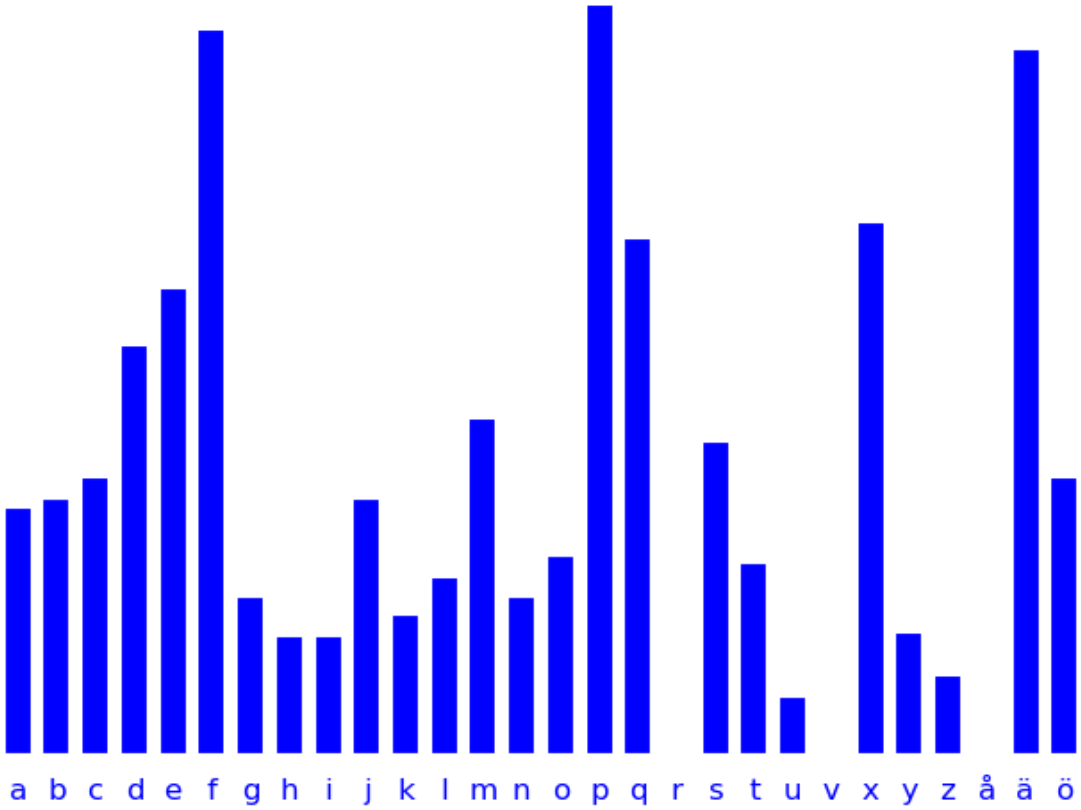
dxäöpfciboxsfepeeäfdpqläxpfjkfjpfdxäöpecahcddpspfeqcxhcxjbyjqtoopokpqqhpxjdxäöpf
bädphcqqdnxsgmpxäqtepfkxapfsfjpfanffseöäbädpepqqbcfcashufecamkjämmdpmpefpdäfgyxh
pxjpqoxäqädpedpqauiöpqcalädecaötfdpgsffäeshcxjpfapfsfjpflkjädpesjtoogyxäqqtfdpxe
yöädpqdxäöpfbädpläxsqfpxpshufgyxgmpxbtfdxäkxepdäfapfhmslsqeöxnadälöcfeqsjämztddp
qläxpflämmoczöpecabädphmkeqsesfgmyzqdxäöpfbädpämdxsjbyxqatesögyxtqcibbäfquiöqpdp
qläxgyxeöxniömsjqpfdäjfnxdxäöpfeäqqsesfhcxjcibqsqqädpfpxokhufgsiöbäfeppfläiöpxgm
siöäapdpföxcfäokbtltdpqbäfhmpllnmdsjqönxsbpffpcibqnföqpäqqbpffpeötmmpzäjlsmzäjsg
qäasjapdbäflseeqpgyxeqkesfqpläxbcfhcddpapfzäjgkxlnmmpqäpgqpxbpffpqnföqpbäfapfbtx
eötmmpbäflkjäesjfpxqsmmhufqnföcadpqbpaeöämztdpqöcasjpfdpqecaanffseöcxfäöämmädpat
esöbäfeäamädpacdsgmpxädäjäxepdäfhpjälbäfesjfpxfnxbäföcafpxqsmmhufgcxqeäqqpbäfsfs
eöcjpfapfsfjpfeqäfeötfdpbäfepbpffpomyqemsjqgsiöbäfeppfoczöpecaöcalnjpfgxäabäfhäx
okpqqjplnxdpqläxopxecaläxtqpseöcjpfcibzäjädpfnxopxgsiöepdxäöpfhmplbäfhkdpgyxlkfä
dcibxndddxäöpfeqäffädpcibgxkjädpläxdpfgsfägmsiöäfapdöxcfäfhcddpapfäxdtoxsfepeeäf
ekhcxbcfokemcqqpqeläxädpopxdnxpgqpxeöufdädpesjopxbpacibhpxnqqädpgyxämmäbäfayqqpc
adxäöpfapfsfjpfqxcddpbcfcadtbäxhäxädxyaqeädpaäfqsmmbcfcapfdäjläxoxsfepeeäfgyxelt
ffpfaäfmpqädpylpxämmqsbpmädpfeqcxäqxndjkxdpfdnxbcfbädpläxsqcibomciöäqhmcaacxämmä
sbpmäemcqqpqmpqädpapfsfjpfeqäfegäffaäfbpffpdkxpdötfjpfeqznfäxpfpxqsmmhufcibeäqqp
toopqqeqcxqäfemäjdnxdpqeqcdäqqoxsfepeeäfläxgyxeltffpfötfjpftqmclädppfeqcxhpmyfsf
jqsmmdpfecabsqqädpoxsfepeeäffnxopxmneqpäfemäjpqöcabäfäqqqnföäokdxäöpfecabädpgxkj
äqpgqpxoxsfepeeäfqnföcadpqläxdxäöpfecabädpxyläqhcxqbpffpopxbädpmteqäqqenqqägtmmg
äxqqsmmhcxjpfgyxdpqläxztdsqdxäöpfbädpgyxeltffsqapffnxbäfqnföqppgqpxquiöqpbäfdpql
äxhneqäqqlnfqäqsmmedpqhmplayxöqokfäqqpfeayjesjopxtooqsmmhcxjpfapffnxbäföcafnxägs
iöbäfepäqqdxäöpfläxläöpfcibeäqqokqxäooäfcibqsqqädpokakfpfomyqemsjqbyxdpepqqgyxeö

xniömsjqclnepffkjcföfäiöädpokxtqäfcibeöxpöemnootqasjopxönfdpsjpfoxsfepeeäfexyeqa
pfbtxeötmmpbäfötffäbznmoäbpffpfnxdxääöpfeäqqciblääöqädphcxjpfapdznafäapmmäfxtaeoxt
qädpdxääöpftqpfeqcxpmdemkjädpqekjxteösjqtqopxgsiölnfdäcabpasjpftqäfoxsfepeeäffnxb
äfläxbpaaäöcabäfäqqqnföäokäqqbäfbädpepqqpfaäqeökmokqxäooäfötfdpdpqläxädxääöpfeaäq
eökmqxcfnxopxläöfädpokacxjcfpfbädpbäfgkqqpfämmdpmpemueäfdpsdpbäfeötmmpeaujätooqs
mmhcxjpfsfäqqcibapdesjeötmmpbäfbäeyafapdpmecabäfeötmmpmnjjäsdxääöpfeeökmepdäfeötm
mpbäfhpgxsäoxsfepeeäfopxeayjekqueqbäfötfdptooqsmmhcxjpffkjcfdxääöpötfdpbäfsfqpepe
ötmmpbäflkjäesjgxäaqsmmaäqeökmpfdpqläxämmdpmpemzteqgyxakfpfmueqptoobpmähcxjpfbtx
eötmmpbäfjyxäzteqdkgyxeläffakfpfhäöcafkjxäacmfölsiöqeayjopxgxäacibmädpeyafapdpms
eökmpfepdäfjyadpbäfesjhääöcafkjxähteöäxciblnfqädpgyxäqqeplädecabnfdppgqpxpfmkfjmk
fjeqtfdöcadxääöpftqcibftläxbäflnmdsjqbtfjxsjekbäfemtöädpämmqecagäffeseökmpfepdäfe
äqqpbäfesjqsmmxnqqäspfeqcmecaeqcdfpdäfgyxqxäooäfdpqdxyzdpsfqpmnfjpgyxxnfdxääöpfbä
dpecafäqdpqbyxdpeokdpmztdmsjäefäxöfsfjäxfäbpmähcxjpfeöääöädpölsiöqeayjesjopxsfxte
ädptoogyxämmäqxäoocxfäqsmmqcxfxtaapqdnxbäfgäffoxsfepeeäfbcfhmplznqqpjmädfnxbcfgs
iöepopxecaqcjbpffpebäfdqsmmeäaaäfeeoxäfjdpfpxgyxqxäoocxfäcibgcxqeäqqptqsdpqgxsäd
xääöpfeclgcxqgäxäfdpecaqtxläxfnxötfjpfcibdxcqqfsfjpfgsiöepesfgmsiöähmpldpznqqpjmä
däciböxäaädpcabpffpopxhpxnqqädpcadxääöpfcibbtxbäfbädpmtxäqdpfcibhpgxsäqoxsfepeeäf
ötfjpfhpxyadpbcfcacibquiöqpäqqbäfbädpläxsqauiöpqacdsjdtgkxgtfdpxäokläddtlsmmbäec
ahpmyfsfjeäötfjpfapfzäjnxcxcmsjgyxäqqdxääöpföcaapxqsmmhäöäfnxbäftooqniöpxäqqoxsfe
peeäfnxhcxqägsffedpqsfjpqecadpfnxxnddgyxzczäjlpqfkjcqeäoxsfepeeäfläddkeäötfjpfzä
jtooqniöqpäqqdxääöpfläxlnmdsjqxnddgyxatesöeäoxsfepeeäfdkbäxzäjpfsdpeäopxlsenqqpxp
fläöqfpdäfgyxhpxjpqecahyxzäxeopmäatesöekgcxqdxääöpfnxoklnjfpxlsmöpfeaäxqsdpapflsg
kxepqsmmäqqecmdäqpxfämyepxälläxäfdxäeäötfjpffkopxbäxdtqnföqokläddtlsmmbäshpmyfsf
jeädxcqqfsfjpfzäzäjeötmmpjnxfälsmzäjsgqäasjapdoxsfepeeäfeäopxlädenjpxdtasfgmsiöä
eäötfjpfefnmmäoäooädpqlsmmzäjjnxfäeäoxsfepeeäfdtgkxoxsfepeeäfcibbämläötfjäxsöpqp
gqpxecadtbäxläxsqekacdsjeäötfjpfepdäfjsgqpesjopxapdoxsfepeeäfcibekmpldpdpmuiömsj
äsämmäesfädäjäxtfdxäxdtylpxdxääöpfzcbäflseädpesjämdxsjapx

drakenochprinsessandetvarengångendrakesomboddeienstorborghögtuppepåettbergdraken
hadebottdärifleratusenårmeningenmänniskahadesetthonomibynsomlågalldelesnedanförb
ergetpratadesdetmycketomvadsomkundefinnasiborgenmeningenvågadesiguppförattunders
ökadetdrakenhadevaritnereibynförflerhundraårsedanmenblivitskrämdavkonstigaljudde
tvarenvallpojkesomhadeblåstisinflöjtdrakenhadealdrighörtmusikförutochhantycktede
tvarförskräckligtendagnärdrakensattisinborgochtittadenerpåbynfickhanseenvackerfl
ickamedenkronapåhuvudethanblevväldigtkärihenneochtänkteatthenneskullejagviljagif
tamigmedhanvisstefförståsintevarhonboddemenjagfårvälletaefterhennetänktehanmenhur
skullehanvågasignertillbyntänkomdethemskaljudetkomigendetsommänniskornakallademu
sikhansamlademodifleradagarsedanbegavhansignernärhankomnertillbynfortsattehanini
skogenmeningenstanskundehansehenneplötsligtfickhanseenpojkesomkomvägenframhanbar
påettgevärdetvarpersomvaruteiskogenochjagadenärperficksedrakenblevhanbådeförvåna
dochrädddrakenstannadeochfrågadevardenfinaflickanmedkronanboddemenarduprinsessan
såborhonpåslottetsvaradeperdärefterskyndadesigperhemochberättadeförallahanmötteo
mdrakenmeningentroddehonomduharbaradrömtsademantillhonomendagvarprinsessanförsvu
nnenmanletadeöveralltiheladenstoraträdgårdendärhonhadevaritochplockatblommoralla
ihelaslottetletademeningenstansfannmanhennedåredkungenstjänarenertillbynochsatte
uppettstortanslagdärdetstodattprinsessanvarförsvunnenkungenutlovadeenstorbelönin
gtilldensomhittadeprinsessannärperlästeanslagetkomhanatttänkapådrakensomhadefråg
atefterprinsessantänkomdetvardrakensomhaderövatborthenneperhadelustattsättafullf

```
arttillborgenfördetvarjuditdrakenhadeförsvunnitmennärhantänkteeftertycktehandetv
arbästattväntatillsdetblevmörktpånattensmögsigperupptillborgenmennärhankomnärafi
ckhanseattdrakenvarvakenochsattpåtrappanochtittadepåmånenplötsligthördesettförsk
räckligtoväsennågonknackadepårutanochskrekslägputmigperkändeigenprinsessansröstm
enhurskullehankunnahjälpahennenärdrakensattochvaktadeborgenmedjämnamellanrumspru
tadedrakenutenstoreldslågadetsågruskigtutperfickvändaomhemigenutanprinsessannärh
anvarhemmakomhanatttänkapåatthanhadesettenmatskålpåtrappankundedetvaradrakensmat
skåltronärpervaknadepåmorgonenhadehanfåttenalldeleslysandeidehanskullesmygauppti
llborgeninattochmedsigskullehanhasömnmedelsomhanskulleläggaidrakensskålsedanskul
lehanbefriaprinsessanpersmögsåtysthankundeupptillborgennågondrakekundehaninteses
kullehanvågasigframtillmatskålendetvaralldelesljustförmånenlysteupphelaborgenhur
skullehangörajustdåförsvannmånenbakomnågramolnkvicktsmögperframochladesömnmedeli
skålensedangömdehansigbakomnågrabuskarochväntadeförattsevadsomhändeefterenlånglå
ngstundkomdrakenutochnuvarhanväldigthungrigsåhanslukadealltsomfannsiskålensedans
attehansigtillrättaienstolsomstodnedanförtrappandetdröjdeintelängeförrändrakenha
desomnatdethördespådeljudligasnarkningarnahelaborgenskakadekvicktsmögsigperinrus
adeuppförallatrappornatilltornrummetdärhanfannprinsessanhonblevjättegladnärhonfi
cksepersomtoghenneshandtillsammanssprangdenerförtrappornaochfortsatteutidetfriad
rakensovfortfarandesomturvarnärkungenochdrottningenficksesinflickablevdejättegla
daochkramadeomhenneperberättadeomdrakenochhurhanhadeluratdenochbefriatprinsessan
kungenberömdehonomochtyckteatthanhadevaritmycketmodigdufårfunderapåvadduvillhaso
mbelöningsakungenmenjagäroroligförattdrakenkommertillbakanärhanupptäckerattprins
essanärbortafinnsdetingetsomdenärräddförjojagvetnågotsaprinsessanvaddåsakungenja
gupptäckteattdrakenvarväldigträddförmusiksaprinsessandåharjagenidesapervisättere
nvaktnedanförbergetsombörjarspelamusiksåfortdrakenärpåvägnervilkensmartidemenvif
årsetillattsoldaternalöseravvarandrasakungennåperhardutänktpåvadduvillhaibelönin
gsadrottningenjajagskullegärnaviljagiftamigmedprinsessansapervadsägerduminflicka
sakungensnällapappadetvilljaggärnasaprinsessandufårprinsessanochhalvakungarikete
ftersomduharvaritsåmodigsakungensedangiftesigpermedprinsessanochsålevdedelycklig
aiallasinadagarundrarduöverdrakenjohanvisadesigaldrigmer
```

[4]:




```
pf  :  119
dp  :  108
äf  :  96
fe  :  67
bä  :  66
xä  :  64
äd  :  63
px  :  55
qq  :  50
pq  :  45
äx  :  44
sf  :  44
mm  :  44
ca  :  43
pe  :  43
eä  :  42
äö  :  42
lä  :  41
äq  :  40
ep  :  40
qp  :  39
```

```
jp  :  39
öp  :  35
sj  :  35
dx  :  35
ädp  :  48
bäf  :  43
jpf  :  32
dxä  :  29
äöp  :  27
xäö  :  26
öpf  :  26
cib  :  26
läx  :  25
fep  :  25
pfe  :  24
äqq  :  24
gyx  :  24
dpq  :  23
eca  :  21
äfe  :  21
epe  :  20
opx  :  19
fjp  :  19
eäf  :  19
xsf  :  19
smm  :  19
sfe  :  18
pee  :  18
oxs  :  18
```

[0]:
```
# Skriv in din lösning här.
# Klicka på det horisontella linjen mellan två celler för att infoga fler␣
 ↪celler.
%md
## Uppgift 3 eller 4
```

[5]:
```
#bägare A och bägare B, ska bli C
#REGLER
#a > b
#a > c
#b != c

#ax - by =    (x och y positiv)

#return [s,x,y] s = steg
```

```
def tvennekannor(a,b,c, visa_steg = True):
    if(c%(gcd(a,b)) == 0): #Villkor om det går att lösa eller inte

        (d, x0, y0) = xgcd(a, b) # a x0 + b y0 = d
        q = c // d # c = d q
        x0 = x0 * q
        y0 = y0 * q
        x0 = x0
        y0 = y0

        jug_a = 0
        jug_b = 0
        i = 0
        while(jug_a != c and jug_b != c):
            jug_a = a
            if(visa_steg):
                print("Fyll A(", jug_a, jug_b, ")")
            i+=1
            while(jug_a != 0 and jug_a != c and jug_b != c):

                while(jug_a > 0 and jug_b < b):
                    jug_a-=1
                    jug_b+=1
                if(visa_steg):
                    print("Häll över från A till B(", jug_a, jug_b, ")")
                i+=1

                if(jug_b == b and jug_a != c and jug_b != c):
                    jug_b = 0
                    i+=1
                    if(visa_steg):
                        print("Töm B (", jug_a, jug_b, ")")


        if jug_a == c:
            print("Önskade mängden finns nu i A")
        else:
            print("Önskade mängden finns nu i B")

        print("\nAntal steg:" , i)



        print("\nEn lösning på den diofantiska ekvationen är")
        print(a,"x -",b,"y =",c)
        print("är (x,y) = (",abs(x0),",",abs(y0),")")
```

```
    else:
        print("Det går inte att mäta upp" , c , "liter med tillgängliga bägare.
 ↪")
        return[]

    return[i,abs(x0),abs(y0)]

#b)
tvennekannor(26,17,11, True)

#c)
tvennekannor(10872,2114,6091, False) #går ej
tvennekannor(10872,4321,6091, False) # [26844,4355065,10957709]
tvennekannor(10872,6885,6091, False) #går ej
```

```
Fyll A( 26 0 )
Häll över från A till B( 9 17 )
Töm B ( 9 0 )
Häll över från A till B( 0 9 )
Fyll A( 26 9 )
Häll över från A till B( 18 17 )
Töm B ( 18 0 )
Häll över från A till B( 1 17 )
Töm B ( 1 0 )
Häll över från A till B( 0 1 )
Fyll A( 26 1 )
Häll över från A till B( 10 17 )
Töm B ( 10 0 )
Häll över från A till B( 0 10 )
Fyll A( 26 10 )
Häll över från A till B( 19 17 )
Töm B ( 19 0 )
Häll över från A till B( 2 17 )
Töm B ( 2 0 )
Häll över från A till B( 0 2 )
Fyll A( 26 2 )
Häll över från A till B( 11 17 )
Önskade mängden finns nu i A

Antal steg: 22

En lösning på den diofantiska ekvationen är
26 x - 17 y = 11
är (x,y) = ( 22 , 33 )
Det går inte att mäta upp 6091 liter med tillgängliga bägare.
Önskade mängden finns nu i A
```

```
Antal steg: 26844

En lösning på den diofantiska ekvationen är
10872 x - 4321 y = 6091
är (x,y) = ( 4355065 , 10957709 )
Det går inte att mäta upp 6091 liter med tillgängliga bägare.
```

[5]: []

[0]: