

Lab 2: Attacking Classic Crypto Systems

1. Checkpoint 1 — Caesar cipher (Marks: 5)

Cipher (given):

odroboewscdrolocdcwkbdkmyxdbkmdzvkdpybwyeddrobo

1.1 Objective

Write a program to break the Caesar cipher and display the plaintext.

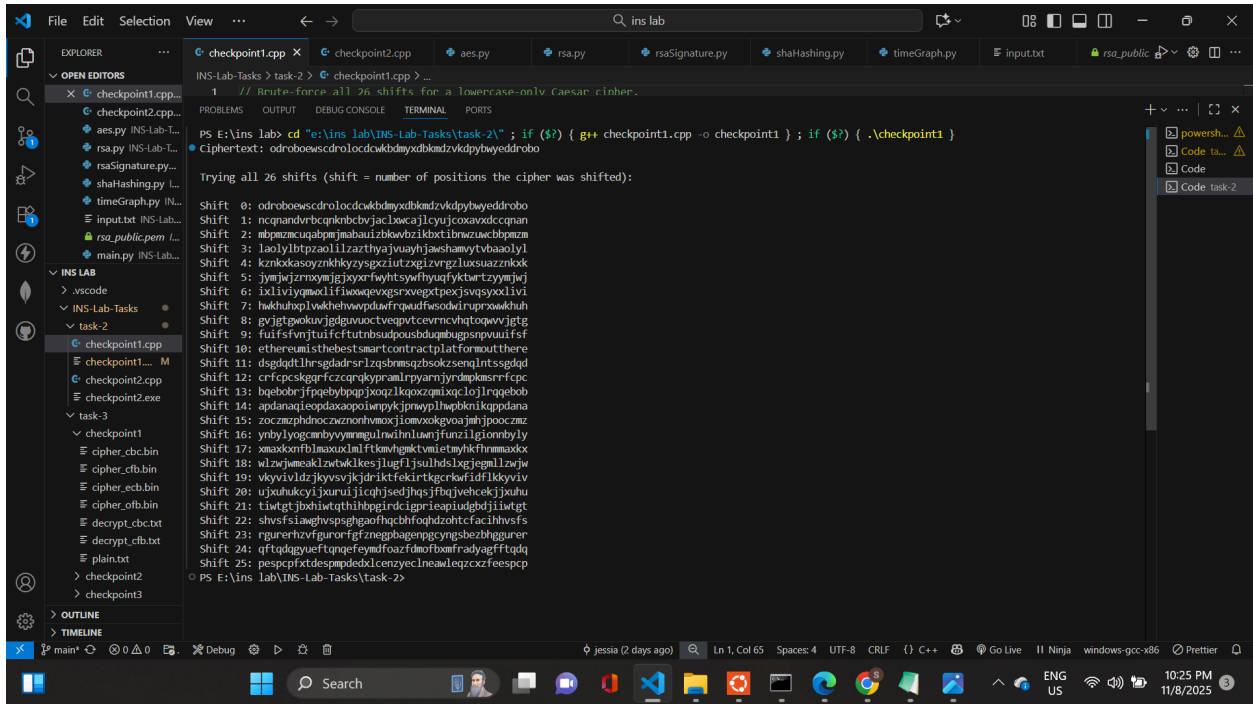
1.2 Approach

- A Caesar cipher is a substitution where each letter is shifted by a fixed offset (0 - 25).
- Brute-force all 26 shifts and inspect outputs. The correct shift yields readable English.
- Implementation choices: C++ program.

1.3 Steps performed

1. Created a small script to try all shifts (see Appendix for the code).
2. Ran the script and inspected the output lines.
3. Identified the readable plaintext and recorded the shift.

1.5 Result



```
INS-Lab-Tasks > task-2 > cd "e:\ins lab\INS-Lab-Tasks\task-2" > if ($?) { g++ checkpoint1.cpp -o checkpoint1 } ; if ($?) { .\checkpoint1 }
1 // Brute-force all 26 shifts for a lowercase-only Caesar cipher.
Ciphertext: odroboewscdrolodcdokbdmzyxdkmdzvkdpybyeyddrobo

Trying all 26 shifts (shift = number of positions the cipher was shifted):

Shift 0: odroboewscdrolodcdokbdmzyxdkmdzvkdpybyeyddrobo
Shift 1: ncnqandvrbcnknbcbvjac1xwcajlcyu1coaxvdcnqan
Shift 2: mbpmzmcuqabpmjmbauizblawbz1kbt1bnwzuecbpmz
Shift 3: laoly1btzaol11zarthya1juaah1awshamvytvbaaolyl
Shift 4: knkdkasoyznkhkzyzspgz1utzg1zvrqz1uxsuazknok
Shift 5: jwajjzrnqymjg1jyxrfwyhtsywfhuyqfkwrtwzymbjw
Shift 6: ixliivymed11f1uoaqevgsrxvegtpejxsvqsyod1vi
Shift 7: hwhuhop1vkhshwop1wfrqadufwsod1rupxawkhuh
Shift 8: gvgtpwkwaj1g1gduuactveqptvevmvhtopwvjtg
Shift 9: fuifsfvntuifc1f1utnbsudpousbdugbgsnpuuifsf
Shift 10: ethereu1sthebestsmartcontractplatformoutthere
Shift 11: dsgdgt1hrsgdadr1rlzqsbmsqzbsokzsenq1ntsgdgt
Shift 12: crfcpckggrf1czcqrqkyp1ra1lryarnjyrdm1kmsrrfcp
Shift 13: bqebobr1fpgbybpg1jxoqz1kqoxqmi1xqclj1rqebob
Shift 14: apdana1eopdaxaop1wnpyk1jrnwpy1hwpbkn1kppdana
Shift 15: zoczn1p1hdncczn1nonhvmox1j1omw1xokv1oa1jnh1poo1cz
Shift 16: ynbylyogcmbyvmymgm1nwh1n1uwn1funz11gionbyly
Shift 17: zmxakm1f1uaxz1m1f1uwhg1p1k1v1e1tm1j1k1f1m1u1x
Shift 18: w1z1v1j1w1a1k1z1w1b1k1e1s1j1u1g11j1a1u1h1d1b1x1j1a1g1a1l1x1j1v
Shift 19: v1g1v1v1d1z1j1k1y1v1j1k1d1r1k1t1f1o1k1r1t1g1c1r1k1w1f1d1l1k1y1v
Shift 20: u1j1u1h1u1k1c1y1j1x1u1r1j1j1c1q1h1s1d1j1h1q1s1f1b1j1v1e1h1e1k1j1x1u1h
Shift 21: t1w1t1g1t1j1b1x1h1w1t1q1t1h1b1p1g1d1c1g1p1r1e1a1p1u1d1g1d1j1i1w1t1g1t
Shift 22: shvsf1s1a1w1h1v1s1p1s1g1h1a1o1f1h1q1b1h1f1o1q1d1z1o1t1c1f1a1c1i1h1v1s1f
Shift 23: rgurer1h1z1v1g1u1o1r1f1g1z1n1e1g1b1a1g1e1n1p1c1y1n1s1b1e1z1h1g1u1r1e
Shift 24: q1f1q1d1q1y1u1e1f1t1q1q1e1f1e1m1d1f1o1a1z1f1d1m1f1b1m1f1r1a1d1y1a1g1f1f1t1d1q
Shift 25: p1e1s1p1c1p1f1x1d1e1s1p1e1d1c1e1n1z1e1c1l1e1a1w1l1e1q1z1c1z1f1e1e1s1p1c1p
PS E:\ins lab\INS-Lab-Tasks\task-2>
```

- Detected shift: 10
- Recovered plaintext: **ethereum is the best smart contract platform out there**

2. Checkpoint 2 — Substitution ciphers (Marks: 8 + 7)

Cipher-1: af p xpkcaqvnkp pfg, af ipqe qpri, gauuikifc tpw, ceiri udvk tiki afgarxifrphni cd eao- -wvmd popkwn, hiqpvri du ear jvaql vfgikrcpfgafm du cei xkafqaxnir du xrwqedearcdkw pfg du ear aopmafpcasi xkdhafmr afcd fit pkipt. ac tpr qdoudkcafmd cd lfdt cepc au pfwceafm epxxifig cd ringdf eaorinu hiudki cei opceiopcaqr du cei uaing qdvng hi qdoxniciw tdklig dvc- -pfg edt rndtnw ac xkdqiigig, pfg edt odvfcpafdvri cei dhrcpqnr--ceiki tdvng pc niprc kiopaf dfi mddg oafg cepc tdvng qdfcafvi cei kiripkqe

Cipher-2: aceah toz puvg vcdl omj puvg yudqecov, omj loj aum klu thmjuv hs klu zlcvu shv zcbkg guovz, upuv zcmdu lcz vuwovroaeu jczyoyuovomdu omj qmubyudkuj vukqvm. Klu vcdluz lu loj avhqnlk aodr svhw lcz kvopuez loj mht audhwu o ehdoe eunumj, omj ck toz yhyqeoveg auecupuj, tlakupuv klu hej sher wcnlk zog, klok klu lcee ok aon umj toz sqee hs kqmmuez zkqssuj tckl kvuoqzvu.

omj cs klok toz mhg umhqn shv sowu, kluvu toz oezh lcz yvhehmnuj pcnhqv kh wovpue ok. kcwu thvu hm, aqk ck zuuwuj kh lopu eckkeu ussudk hm wv. aonncmz. ok mcmukg lu toz wqdl klu zowu oz ok scsckg. ok mcmukg-mcmu klug aunom kh doe lcw tuee-yvuzuvpuj; aqk qmdlomnuj thqej lopu auum muovuv klu wovr. kluvu tuvu zhwu klok zlhhr klucv luoj omj klhqnl klcz toz khh wqdl hs o nhhj klcmn; ck zuuwuj qmsocv klok omghmu zlhqej yhzuzz (oyyovumkeg) yuvyukqoe ghqkl oz tuee oz (vuyqkujeg) cmubloqzkcae tuoekl. ck tcee lopu kh au yocj shv, klug zocj. ck czm'k mokqvoe, omj kvhgaeu tcee dhwu hs ck! aqk zh sov kvhgaeu loj mhg dhwu; omj oz wv. aonncmz toz numuvhqz tckl lcz whmug, whzk yuhyeu tuvu tceecmn kh shvncpu lcw lcz hjckcu omj lcz nhhj shvkqmu. Lu vuwocmuj hm pczckcmn kuvwz tckl lcz vueokcpuz (ubduyk, hs dhqvzu, klu zodrpceeu-aonncmzuz), omj lu loj womg juphkuj ojwcvuvz owhmn klu lhaackz hs yhhv omj qmcwyhvkomp sowcecu. aqk lu loj mh dehzu svcumjz, qmkce zhwu hs lcz ghqmnuv dhqzcmz aunom kh nvht qy. klu uejuzk hs kluzu, omj aceah'z sophqvcku, toz ghqm svjh aonncmz. tlum aceah toz mcmukg-mcmu lu ojhykuj svjh oz lcz lucv, omj avhqnlk lcw kh ecpu ok aon umj; omj klu lhyuz hs klu zodrpceeu-aonncmzuz tuvu scmoeeeg jozluj. aceah omj svjh loyyumuj kh lopu klu zowu acvkljog, zuykuwauv 22mj. ghq loj aukku dhwu omj ecpu luvu, svjh wg eoj, zocj aceah hmu jog; omj klum tu dom dueuavoku hqv acvkljog-yovkcu dhwshvkoaeg khnuklu. ok klok kcwu svjh toz zkcee cm lcz ktuumz, oz klu lhaackz doeuj klu cvvzyhmzcaeu ktumkcu auktuom dlcejlhj omj dhwcmm hs onu ok klcvkg-klvuu

Frequency distribution English characters

1.2 Substitution Ciphers.

a: 8.05% b: 1.67% c: 2.23% d: 5.10%

e: 12.22% f: 2.14% g: 2.30% h: 6.62%

i: 6.28% j: 0.19% k: 0.95% l: 4.08%

m: 2.33% n: 6.95% o: 7.63% p: 1.66%

q: 0.06% r: 5.29% s: 6.02% t: 9.67%

u: 2.92% v: 0.82% w: 2.60% x: 0.11%

y: 2.04% z: 0.06%

2.1 Objective

Write programs to decipher both substitution ciphers. Explain which input was easier to break and why.

2.2 Background and approach

- A monoalphabetic substitution cipher replaces each plaintext letter with a unique ciphertext letter (a permutation of the alphabet). It preserves letter frequencies and many word patterns.
- **Attack strategy used:**
 1. **Frequency analysis** to build an initial mapping: map the most frequent ciphertext letters to the most frequent English letters.
 2. **hill-climbing:** starting from the initial key, repeatedly propose small changes (swap two plaintext-letter assignments) and accept changes that improve a scoring function.
 3. **Scoring function:** measures how English-like a candidate decryption is. Options used in experiments: common-word match counts, n-gram (quadgram) log-probability scoring.

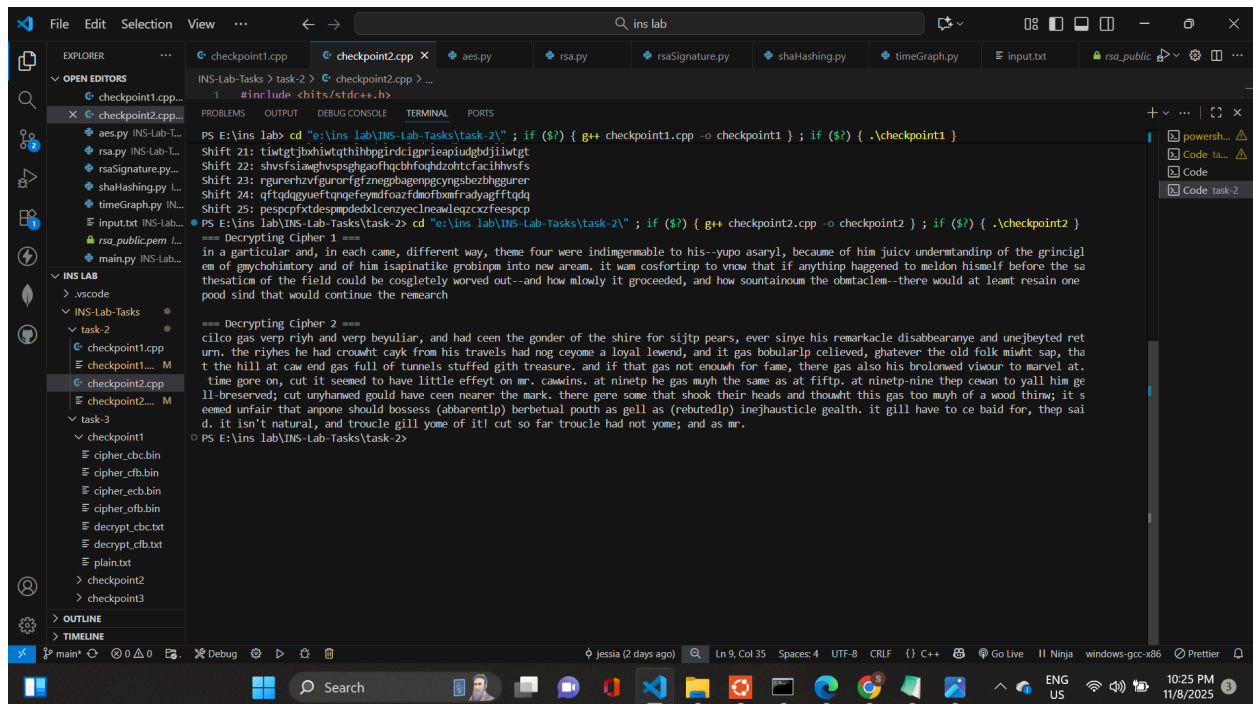
This approach is standard and effective for reasonably long ciphertexts.

2.3 Steps performed

1. Implemented a substitution-solver program
 - reads the ciphertext,
 - computes letter frequencies,
 - builds an initial key via frequency ordering,
 - runs randomized hill-climbing with swap moves,
 - prints the best-scoring plaintext and the inferred mapping.
2. Ran the solver separately on Cipher-1 and Cipher-2.

3. Manually inspected the best candidates and performed small manual fixes when needed to obtain readable plaintext.

2.5 Results and analysis



Which input was easier to break?

- **Cipher-2** (the longer text) was *easier to break* in automated experiments because: longer ciphertexts provide more reliable statistics (letter frequency and n-gram patterns), they contain more repeated words and grammatical structure which the scoring function exploits, and the hill-climber can lock onto consistent mappings more easily.