НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

з лабораторної роботи №*1*

з дисципліни «Криптографічні методи захисту інформації»

на тему:

*Розробка криптосистем на основі шифрів Цезаря та Тритемія*

|  |  |
| --- | --- |
| Виконала: | Керівник: |
| студентка групи КМ-01 | *ст. викладач Бай Ю. П.* |
| *Боженко А. О.* |  |

Київ — 2022

ЗМІСТ

[ЗАВДАННЯ 2](#_Toc94913657)

[Основні теоретичні відомості з шифру Цезаря 3](#_Toc94913658)

[Завдання з шифру Цезаря 3](#_Toc94913659)

[Основні теоретичні відомості з шифру Тритемія 5](#_Toc94913660)

[Завдання з шифру Тритемія 5](#_Toc94913661)

[Список літератури 10](#_Toc94913662)

[Додаток 1 11](#_Toc94913663)

[Додаток 2 12](#_Toc94913664)

# ЗАВДАННЯ

***Мета роботи:*** розробити криптосистеми на основі шифрів Цезаря та Тритемія.

1. Зашифрувати текст довжиною від 10 до 50 символів, користуючись шифром Цезаря з певним ключем *K*. Результат записати в [Таблицю\_1](https://docs.google.com/spreadsheets/d/1vg-aCHGGX2oZDjuAWWRWecRBWjwAYLmkgonEp1CFBGA/edit?usp=sharing), в СВІЙ РЯДОК, стовпчик AG (*1 бал*).

2. Дано криптотекст *C*, одержаний шифруванням Цезаря з невідомим ключем *K* (див. [Таблицю\_1](https://docs.google.com/spreadsheets/d/1vg-aCHGGX2oZDjuAWWRWecRBWjwAYLmkgonEp1CFBGA/edit?usp=sharing), рядок НАД СВОЇМ, стовпчик AG). Методом повного перебору всіх потенційних рішень (методом «грубої сили») дешифрувати заданий криптотекст. У відповіді навести дешифрований текст *P* та ключ *K*. (*2 бали*).

3. Зашифрувати текст довжиною від 10 до 20 символів, користуючись шифром Тритемія за лінійним законом *y = (x + A\*p + B) mod N*. Записати початковий текст та криптотекст в [Таблицю\_2](https://docs.google.com/spreadsheets/d/19vV6-YDYE9fHXQ3q9iaDqeYQlLHj69xGzTOhQase2zI/edit?usp=sharing), в СВІЙ РЯДОК, стовпчики AG, AH (*1 бал*).

4. Уявіть, що Ви перехопили початковий текст та криптотекст (див. [Таблицю\_2](https://docs.google.com/spreadsheets/d/19vV6-YDYE9fHXQ3q9iaDqeYQlLHj69xGzTOhQase2zI/edit?usp=sharing), рядок НАД СВОЇМ, стовпчики AG, AH). Необхідно визначити ключ: знайти параметри *A, B*, але не методом «грубої сили», а розв’язуючі систему двох рівнянь. Відповідь записати в стовпчик AI (рядок НАД СВОЇМ) (*2 бали*).

5. Зашифрувати текст довжиною від 10 до 50 символів, користуючись шифром Тритемія **за квадратичним законом** **(*A, B, C*****від 1 до 5)**. Записати зашифрований текст в [Таблицю\_2](https://docs.google.com/spreadsheets/d/19vV6-YDYE9fHXQ3q9iaDqeYQlLHj69xGzTOhQase2zI/edit?usp=sharing), в СВІЙ РЯДОК, стовпчик AJ (*1 бал*).

6. Методом "brute force" дешифрувати текст, що знаходиться в стовпчику AJ, в рядку НАД СВОЇМ (відомо, що був використаний шифр Тритемія, за квадратичним законом, *A, B, C* від 1 до 5). Відповідь записати в стовпчик AK, знайдений ключ - в AL (рядок НАД СВОЇМ) (*2 бали*).

7. Оформити звіт (*1 бал*).

# Основні теоретичні відомості з шифру Цезаря

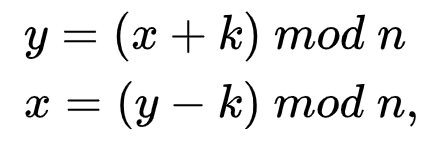
ШифрЦезаря або шифрзсуву — [симетричний](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%BD%D1%96_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%B8_%D1%88%D0%B8%D1%84%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [моноалфавітний](https://uk.wikipedia.org/wiki/%D0%9F%D1%96%D0%B4%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%BE%D1%87%D0%BD%D0%B8%D0%B9_%D1%88%D0%B8%D1%84%D1%80) алгоритм [шифрування](https://uk.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), в якому кожна буква відкритого тексту заміняється на ту, що віддалена від неї в алфавіті на сталу кількість позицій.

Принцип дії полягає в тому, щоб [циклічно зсунути](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D1%8C%D0%BD%D0%B0_%D0%B0%D1%80%D0%B8%D1%84%D0%BC%D0%B5%D1%82%D0%B8%D0%BA%D0%B0) алфавіт, а ключ — це кількість літер, на які робиться зсув.

Якщо зіставити кожному символу алфавіту його порядковий номер (нумеруючи з 0), то шифрування і дешифрування можна виразити формулами:

{\displaystyle y=(x+k)\ mod\ n}

{\displaystyle x=(y-k)\ mod\ n,}



де {\displaystyle ~x} x - порядковий номер символу відкритого тексту, y - {\displaystyle ~y}yy порядковий номер символу шифрованого тексту, {\displaystyle ~n}n  — [потужність](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D1%82%D1%83%D0%B6%D0%BD%D1%96%D1%81%D1%82%D1%8C_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B8%D0%BD%D0%B8) алфавіту, а {\displaystyle ~k}k — ключ.

Шифр Цезаря має замало ключів — на одиницю менше, ніж літер в [абетці](https://uk.wikipedia.org/wiki/%D0%90%D0%BB%D1%84%D0%B0%D0%B2%D1%96%D1%82). Тому його легко зламати перебором — пробувати усі можливі [ключ](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87), поки [розшифрування](https://uk.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B7%D1%88%D0%B8%D1%84%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) не дасть впізнаваний [відкритий текст](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%B8%D0%B9_%D1%82%D0%B5%D0%BA%D1%81%D1%82).

# Завдання з шифру Цезаря

1. Зашифруємо текст «THERE IS NO LOVE SINCERER THAN THE LOVE OF FOOD», користуючись шифром Цезаря з ключем *К* = 5

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |
| 🡫 | 🡫 | 🡫 | … |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c | d | e |

Результат шифрування:

«YMJWJ NX ST QTAJ XNSHJWJW YMFS YMJ QTAJ TK KTTI»

2. Дано криптотекст, одержаний шифруванням Цезаря:

«Duw lv wkh surshu wdvn ri olih.»

Одержимо всі варіанти його дешифрування:

Decoded string(key = 0): DUW LV WKH SURSHU WDVN RI OLIH.

Decoded string(key = 1): CTV KU VJG RTQRGT VCUM QH NKHG.

Decoded string(key = 2): BSU JT UIF QSPQFS UBTL PG MJGF.

Decoded string(key = 3): ART IS THE PROPER TASK OF LIFE.

Decoded string(key = 4): ZQS HR SGD OQNODQ SZRJ NE KHED.

Decoded string(key = 5): YPR GQ RFC NPMNCP RYQI MD JGDC.

Decoded string(key = 6): XOQ FP QEB MOLMBO QXPH LC IFCB.

Decoded string(key = 7): WNP EO PDA LNKLAN PWOG KB HEBA.

Decoded string(key = 8): VMO DN OCZ KMJKZM OVNF JA GDAZ.

Decoded string(key = 9): ULN CM NBY JLIJYL NUME IZ FCZY.

Decoded string(key = 10): TKM BL MAX IKHIXK MTLD HY EBYX.

Decoded string(key = 11): SJL AK LZW HJGHWJ LSKC GX DAXW.

Decoded string(key = 12): RIK ZJ KYV GIFGVI KRJB FW CZWV.

Decoded string(key = 13): QHJ YI JXU FHEFUH JQIA EV BYVU.

Decoded string(key = 14): PGI XH IWT EGDETG IPHZ DU AXUT.

Decoded string(key = 15): OFH WG HVS DFCDSF HOGY CT ZWTS.

Decoded string(key = 16): NEG VF GUR CEBCRE GNFX BS YVSR.

Decoded string(key = 17): MDF UE FTQ BDABQD FMEW AR XURQ.

Decoded string(key = 18): LCE TD ESP ACZAPC ELDV ZQ WTQP.

Decoded string(key = 19): KBD SC DRO ZBYZOB DKCU YP VSPO.

Decoded string(key = 20): JAC RB CQN YAXYNA CJBT XO URON.

Decoded string(key = 21): IZB QA BPM XZWXMZ BIAS WN TQNM.

Decoded string(key = 22): HYA PZ AOL WYVWLY AHZR VM SPML.

Decoded string(key = 23): GXZ OY ZNK VXUVKX ZGYQ UL ROLK.

Decoded string(key = 24): FWY NX YMJ UWTUJW YFXP TK QNKJ.

Decoded string(key = 25): EVX MW XLI TVSTIV XEWO SJ PMJI.

Аналізуючи одержані результати, робимо висновок щодо початкового тексту та ключа:

*P* = ‘ART IS THE PROPER TASK OF LIFE.’, *K* = 3

# Основні теоретичні відомості з шифру Тритемія

Вдосконалений шифр Цезаря. За алгоритмом шифрування, кожен символ повідомлення зміщується на символ, що відстає на певний крок. Тут крок робиться змінним. Наприклад, можна задати закон усунення у вигляді лінійної функції або квадратичної. Сама функція (ди)шифрування має гарантувати ціле значення. Формула для шифру:

yi = (m+k) mod N

yi -номер зашифрованої i-ї літери в алфавіті, m-порядковий номер літери коду, що шифрується в алфавіті, k-число зсуву, N-кількість літер в алфавіті.

Число зсуву (крок зміщення) у даному випадку змінна величина, функціонально залежина від позиції символу в повідомленні. Залежність може бути різною, наприклад, лінійною чи квадратичною.

k = A \* p + B (лінійна залежність).

k = A \* p \* p + B \* p + C (квадратична залежність)

A, B, С – ключі. p - позиція літери у повідомленні.

# Завдання з шифру Тритемія

3. Зашифруємо текст «HARD AS A ROCK», користуючись шифром Тритемія за лінійним законом з параметрами *A* = 5, *B* = 4

Використаємо формулу:

уі = (xі + A\*p + B) mod 26

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |

Результат шифрування:

«LJFW DA S TVOB»

4. Дано початковий текст та криптотекст, одержаний шифруванням Тритемія за лінійним законом з невідомими параметрами *A*, *B* ( *A*, *B* – цілі числа від 1 до 5):

*plaintext* = ‘encoded text’, *cryptotext* = 'julclekxy sfay'

Визначимо параметри *А, В*:

Формула: y = (x + A\*p + B) mod 26

y1 = 9, y2 = 20,

x1 = 4 x2 = 13,

p = 0 p = 1

9 = 4 + A \* 0 + B -> B = 5

20 = 13 + A \* 1 + 5 -> A = 2

Отже, A = 2, B = 5

5. Зашифруємо текст «London is the capital of Great Britain», користуючись шифром Тритемія за квадратичним законом з параметрами *A* =2, *B* = 2, *С* = 2 (*A*, *B*, *С* ϵ [1; 5]):

Використовуємо формулу: yі = (xі + A\*p\*р + B\*р + С) mod 26

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |

Результат шифрування:

«NUBDEX SI HNG IOPYDIV OT ITKOT LZSJAWT»

6. Дано криптотекст, одержаний шифруванням Тритемія за квадратичним законом з невідомими параметрами *A*, *B*, *С* ( *A*, *B*, *С* – цілі числа від 1 до 5):

«wkil rnf cgsgoirncm»

Використовуємо формулу: х = (у – ( A\*p\*р + B\*р + С)) mod 26

Одержимо всі варіанти його дешифрування для *A*, *B*, *С* ϵ [1; 5]:

Decoded (a=1, b=1, c=1): VHBY MWA PZPFNFKAHH

Decoded (a=1, b=1, c=2): UGAX LVZ OYOEMEJZGG

Decoded (a=1, b=1, c=3): TFZW KUY NXNDLDIYFF

Decoded (a=1, b=1, c=4): SEYV JTX MWMCKCHXEE

Decoded (a=1, b=1, c=5): RDXU ISW LVLBJBGWDD

Decoded (a=1, b=2, c=1): VGZV HQT GPETARVKQP

Decoded (a=1, b=2, c=2): UFYU GPS FODSZQUJPO

Decoded (a=1, b=2, c=3): TEXT FOR ENCRYPTION

Decoded (a=1, b=2, c=4): SDWS ENQ DMBQXOSHNM

Decoded (a=1, b=2, c=5): RCVR DMP CLAPWNRGML

Decoded (a=1, b=3, c=1): VFXS CKM XFTHNDGUZX

Decoded (a=1, b=3, c=2): UEWR BJL WESGMCFTYW

Decoded (a=1, b=3, c=3): TDVQ AIK VDRFLBESXV

Decoded (a=1, b=3, c=4): SCUP ZHJ UCQEKADRWU

Decoded (a=1, b=3, c=5): RBTO YGI TBPDJZCQVT

Decoded (a=1, b=4, c=1): VEVP XEF OVIVAPREIF

Decoded (a=1, b=4, c=2): UDUO WDE NUHUZOQDHE

Decoded (a=1, b=4, c=3): TCTN VCD MTGTYNPCGD

Decoded (a=1, b=4, c=4): SBSM UBC LSFSXMOBFC

Decoded (a=1, b=4, c=5): RARL TAB KRERWLNAEB

Decoded (a=1, b=5, c=1): VDTM SYY FLXJNBCORN

Decoded (a=1, b=5, c=2): UCSL RXX EKWIMABNQM

Decoded (a=1, b=5, c=3): TBRK QWW DJVHLZAMPL

Decoded (a=1, b=5, c=4): SAQJ PVV CIUGKYZLOK

Decoded (a=1, b=5, c=5): RZPI OUU BHTFJXYKNJ

Decoded (a=2, b=1, c=1): VGXP NMD MDYRARTEEV

Decoded (a=2, b=1, c=2): UFWO MLC LCXQZQSDDU

Decoded (a=2, b=1, c=3): TEVN LKB KBWPYPRCCT

Decoded (a=2, b=1, c=4): SDUM KJA JAVOXOQBBS

Decoded (a=2, b=1, c=5): RCTL JIZ IZUNWNPAAR

Decoded (a=2, b=2, c=1): VFVM IGW DTNFNDEOND

Decoded (a=2, b=2, c=2): UEUL HFV CSMEMCDNMC

Decoded (a=2, b=2, c=3): TDTK GEU BRLDLBCMLB

Decoded (a=2, b=2, c=4): SCSJ FDT AQKCKABLKA

Decoded (a=2, b=2, c=5): RBRI ECS ZPJBJZAKJZ

Decoded (a=2, b=3, c=1): VETJ DAP UJCTAPPYWL

Decoded (a=2, b=3, c=2): UDSI CZO TIBSZOOXVK

Decoded (a=2, b=3, c=3): TCRH BYN SHARYNNWUJ

Decoded (a=2, b=3, c=4): SBQG AXM RGZQXMMVTI

Decoded (a=2, b=3, c=5): RAPF ZWL QFYPWLLUSH

Decoded (a=2, b=4, c=1): VDRG YUI LZRHNBAIFT

Decoded (a=2, b=4, c=2): UCQF XTH KYQGMAZHES

Decoded (a=2, b=4, c=3): TBPE WSG JXPFLZYGDR

Decoded (a=2, b=4, c=4): SAOD VRF IWOEKYXFCQ

Decoded (a=2, b=4, c=5): RZNC UQE HVNDJXWEBP

Decoded (a=2, b=5, c=1): VCPD TOB CPGVANLSOB

Decoded (a=2, b=5, c=2): UBOC SNA BOFUZMKRNA

Decoded (a=2, b=5, c=3): TANB RMZ ANETYLJQMZ

Decoded (a=2, b=5, c=4): SZMA QLY ZMDSXKIPLY

Decoded (a=2, b=5, c=5): RYLZ PKX YLCRWJHOKX

Decoded (a=3, b=1, c=1): VFTG OCG JHHDNDCIBJ

Decoded (a=3, b=1, c=2): UESF NBF IGGCMCBHAI

Decoded (a=3, b=1, c=3): TDRE MAE HFFBLBAGZH

Decoded (a=3, b=1, c=4): SCQD LZD GEEAKAZFYG

Decoded (a=3, b=1, c=5): RBPC KYC FDDZJZYEXF

Decoded (a=3, b=2, c=1): VERD JWZ AXWRAPNSKR

Decoded (a=3, b=2, c=2): UDQC IVY ZWVQZOMRJQ

Decoded (a=3, b=2, c=3): TCPB HUX YVUPYNLQIP

Decoded (a=3, b=2, c=4): SBOA GTW XUTOXMKPHO

Decoded (a=3, b=2, c=5): RANZ FSV WTSNWLJOGN

Decoded (a=3, b=3, c=1): VDPA EQS RNLFNBYCTZ

Decoded (a=3, b=3, c=2): UCOZ DPR QMKEMAXBSY

Decoded (a=3, b=3, c=3): TBNY COQ PLJDLZWARX

Decoded (a=3, b=3, c=4): SAMX BNP OKICKYVZQW

Decoded (a=3, b=3, c=5): RZLW AMO NJHBJXUYPV

Decoded (a=3, b=4, c=1): VCNX ZKL IDATANJMCH

Decoded (a=3, b=4, c=2): UBMW YJK HCZSZMILBG

Decoded (a=3, b=4, c=3): TALV XIJ GBYRYLHKAF

Decoded (a=3, b=4, c=4): SZKU WHI FAXQXKGJZE

Decoded (a=3, b=4, c=5): RYJT VGH EZWPWJFIYD

Decoded (a=3, b=5, c=1): VBLU UEE ZTPHNZUWLP

Decoded (a=3, b=5, c=2): UAKT TDD YSOGMYTVKO

Decoded (a=3, b=5, c=3): TZJS SCC XRNFLXSUJN

Decoded (a=3, b=5, c=4): SYIR RBB WQMEKWRTIM

Decoded (a=3, b=5, c=5): RXHQ QAA VPLDJVQSHL

Decoded (a=4, b=1, c=1): VEPX PSJ GLQPAPLMYX

Decoded (a=4, b=1, c=2): UDOW ORI FKPOZOKLXW

Decoded (a=4, b=1, c=3): TCNV NQH EJONYNJKWV

Decoded (a=4, b=1, c=4): SBMU MPG DINMXMIJVU

Decoded (a=4, b=1, c=5): RALT LOF CHMLWLHIUT

Decoded (a=4, b=2, c=1): VDNU KMC XBFDNBWWHF

Decoded (a=4, b=2, c=2): UCMT JLB WAECMAVVGE

Decoded (a=4, b=2, c=3): TBLS IKA VZDBLZUUFD

Decoded (a=4, b=2, c=4): SAKR HJZ UYCAKYTTEC

Decoded (a=4, b=2, c=5): RZJQ GIY TXBZJXSSDB

Decoded (a=4, b=3, c=1): VCLR FGV ORURANHGQN

Decoded (a=4, b=3, c=2): UBKQ EFU NQTQZMGFPM

Decoded (a=4, b=3, c=3): TAJP DET MPSPYLFEOL

Decoded (a=4, b=3, c=4): SZIO CDS LOROXKEDNK

Decoded (a=4, b=3, c=5): RYHN BCR KNQNWJDCMJ

Decoded (a=4, b=4, c=1): VBJO AAO FHJFNZSQZV

Decoded (a=4, b=4, c=2): UAIN ZZN EGIEMYRPYU

Decoded (a=4, b=4, c=3): TZHM YYM DFHDLXQOXT

Decoded (a=4, b=4, c=4): SYGL XXL CEGCKWPNWS

Decoded (a=4, b=4, c=5): RXFK WWK BDFBJVOMVR

Decoded (a=4, b=5, c=1): VAHL VUH WXYTALDAID

Decoded (a=4, b=5, c=2): UZGK UTG VWXSZKCZHC

Decoded (a=4, b=5, c=3): TYFJ TSF UVWRYJBYGB

Decoded (a=4, b=5, c=4): SXEI SRE TUVQXIAXFA

Decoded (a=4, b=5, c=5): RWDH RQD STUPWHZWEZ

Decoded (a=5, b=1, c=1): VDLO QIM DPZBNBUQVL

Decoded (a=5, b=1, c=2): UCKN PHL COYAMATPUK

Decoded (a=5, b=1, c=3): TBJM OGK BNXZLZSOTJ

Decoded (a=5, b=1, c=4): SAIL NFJ AMWYKYRNSI

Decoded (a=5, b=1, c=5): RZHK MEI ZLVXJXQMRH

Decoded (a=5, b=2, c=1): VCJL LCF UFOPANFAET

Decoded (a=5, b=2, c=2): UBIK KBE TENOZMEZDS

Decoded (a=5, b=2, c=3): TAHJ JAD SDMNYLDYCR

Decoded (a=5, b=2, c=4): SZGI IZC RCLMXKCXBQ

Decoded (a=5, b=2, c=5): RYFH HYB QBKLWJBWAP

Decoded (a=5, b=3, c=1): VBHI GWY LVDDNZQKNB

Decoded (a=5, b=3, c=2): UAGH FVX KUCCMYPJMA

Decoded (a=5, b=3, c=3): TZFG EUW JTBBLXOILZ

Decoded (a=5, b=3, c=4): SYEF DTV ISAAKWNHKY

Decoded (a=5, b=3, c=5): RXDE CSU HRZZJVMGJX

Decoded (a=5, b=4, c=1): VAFF BQR CLSRALBUWJ

Decoded (a=5, b=4, c=2): UZEE APQ BKRQZKATVI

Decoded (a=5, b=4, c=3): TYDD ZOP AJQPYJZSUH

Decoded (a=5, b=4, c=4): SXCC YNO ZIPOXIYRTG

Decoded (a=5, b=4, c=5): RWBB XMN YHONWHXQSF

Decoded (a=5, b=5, c=1): VZDC WKK TBHFNXMEFR

Decoded (a=5, b=5, c=2): UYCB VJJ SAGEMWLDEQ

Decoded (a=5, b=5, c=3): TXBA UII RZFDLVKCDP

Decoded (a=5, b=5, c=4): SWAZ THH QYECKUJBCO

Decoded (a=5, b=5, c=5): RVZY SGG PXDBJTIABN

Аналізуючи одержані результати, робимо висновок щодо початкового тексту та значень параметрів *A*, *B*, *С*:

*P* = 'TEXT FOR ENCRYPTION', *A* = 1, *B* = 2, *С* = 3

# Список літератури

1. Тарнавський (2018). [*Технології захисту інформації*](http://ela.kpi.ua/bitstream/123456789/23896/1/TZI_book.pdf). КПІ ім. Ігоря Сікорського.
2. Holden (2017). *The Mathematics of Secrets*. Princeton University Press. [ISBN](https://uk.wikipedia.org/wiki/ISBN) [978-0691141756](https://uk.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B5%D1%86%D1%96%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0:%D0%94%D0%B6%D0%B5%D1%80%D0%B5%D0%BB%D0%B0_%D0%BA%D0%BD%D0%B8%D0%B3/978-0691141756).
3. <https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80_%D0%A2%D1%80%D0%B8%D1%82%D0%B5%D0%BC%D0%B8%D1%83%D1%81%D0%B0>

# Додаток 1

Текст програми, що реалізує криптосистему Цезаря

def encode(text, alphabet, step):  
 encoded\_text = ""  
 for symb in text:  
 ind = alphabet.find(symb)  
 if ind != -1:  
 encoded\_text += chr(ord(alphabet[0]) + ((ind + step) % len(alphabet)))  
 else:  
 encoded\_text += symb  
 return encoded\_text  
  
  
def decode(text, alphabet, step):  
 decoded\_text = ""  
 for symb in text:  
 ind = alphabet.find(symb)  
 if ind != -1:  
 decoded\_text += chr(ord(alphabet[0]) + (len(alphabet) + ind - step) % len(alphabet))  
 else:  
 decoded\_text += symb  
 return decoded\_text  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 # Encoding of the given string  
 str = "There is no love sincerer than the love of food".upper()  
 A = ord('A')  
 alpha = "".join([chr(i) for i in range(A, A + 26)])  
 encode(str, alpha, 5)  
 print(f"Encoded string: {str}")  
  
  
 # Decoding of the given string  
 # str = """Duw lv wkh surshu wdvn ri olih.""".upper()  
 # print("Encoded string: " + str)  
 # A = ord('A')  
 # alpha = "".join([chr(i) for i in range(A, A + 26)])  
 # print("------------------------------")  
 # for i in range(26):  
 # decoded = decode(str, alpha, i)  
 # print(f"Decoded string({i}): {decoded}")

# Додаток 2

Текст програми, що реалізує криптосистему Тритемія

from caesarCipher import encode  
  
  
def alpha\_ind(letter, alphabet):  
 return alphabet.find(letter)  
  
  
def is\_whole(number):  
 return int(number + 1) - number == 1  
  
  
def comb(n):  
 res = []  
 i = n  
 while i >= 0:  
 res.append([n, i])  
 if n != i:  
 res.append([i, n])  
 i -= 1  
 return res  
  
  
def encode\_linear(text, alphabet, a, b):  
 encoded\_text = ""  
 position = 0  
 for letter in text:  
 key = a \* position + b  
 encoded\_text += encode(letter, alphabet, key)  
 position += 1  
 return encoded\_text  
  
  
def linear\_coefficients(encoded\_text, decoded\_text, alphabet):  
 a = None  
 b = None  
 position\_1 = -1  
 position\_2 = -1  
 i = 0  
 for decoded\_letter in encoded\_text:  
 if decoded\_letter in alphabet:  
 if position\_1 == -1:  
 position\_1 = i  
 else:  
 position\_2 = i  
 break  
 i += 1  
 if position\_1 != -1 and position\_2 != -1:  
 x\_1 = alpha\_ind(decoded\_text[position\_1], alphabet)  
 x\_2 = alpha\_ind(decoded\_text[position\_2], alphabet)  
 y\_1 = alpha\_ind(encoded\_text[position\_1], alphabet)  
 y\_2 = alpha\_ind(encoded\_text[position\_2], alphabet)  
 # -------------  
 # y\_1 = (x\_1 + a \* position\_1 + b) % N, where N is length of alphabet  
 # y\_2 = (x\_2 + a \* position\_2 + b) % N  
 # let index\_i = (x\_i + a \* position\_i + b)  
 # so  
 # y\_1 = index\_1 % N  
 # y\_2 = index\_2 % N  
 # y\_i = index\_i % N -> index\_i - f\_i \* N = y\_i,  
 # where f\_i is integer (sign(f\_i) == sign(index\_i)), that is the nearest lowest number divisible by N  
 # so, in that program suppose that index\_i is always positive (bug if negative is necessary)  
 # f\_i belongs to natural numbers with zero  
 # y\_1 = index\_1 - f\_1 \* N  
 # y\_2 = index\_2 - f\_2 \* N  
 # so  
 # index\_1 = y\_1 + f\_1 \* N  
 # index\_2 = y\_2 + f\_2 \* N  
 # so  
 # x\_1 + a \* position\_1 + b = y\_1 + f\_1 \* N  
 # x\_2 + a \* position\_2 + b = y\_2 + f\_2 \* N  
 # so  
 # (according to the first equation): b = y\_1 + f\_1 \* N - x\_1 - a \* position\_1  
 # (second equation): x\_2 + a \* position\_2 + y\_1 + f\_1 \* N - x\_1 - a \* position\_1 = y\_2 + f\_2 \* N  
 # a \* (position\_2 - position\_1) = -(x\_2) - (y\_1) - (f\_1) \* N + x\_1 + y\_2 + f\_2 \* N  
 # a = ( -(x\_2) - (y\_1) - (f\_1) \* N + x\_1 + y\_2 + f\_2 \* N ) / (position\_2 - position\_1)  
 # we suppose that a and b must be positive integers  
 # -----------  
 # a = ( -(x\_2) - (y\_1) - 0 \* len(alphabet) + x\_1 + y\_2 + 0 \* len(alphabet) ) / (position\_2 - position\_1)  
 # b = y\_1 + 0 \* len(alphabet) - x\_1 - a \* position\_1  
 i = 0  
 while True:  
 f = comb(i)  
 for f\_pair in f:  
 a = ( -(x\_2) - (y\_1) - f\_pair[0] \* len(alphabet) + x\_1 + y\_2 + f\_pair[1] \* len(alphabet) ) / (position\_2 - position\_1)  
 if a > 0 and is\_whole(a):  
 b = y\_1 + f\_pair[0] \* len(alphabet) - x\_1 - a \* position\_1  
 if b > 0 and is\_whole(b):  
 break  
 else:  
 a = None  
 b = None  
 if a is not None and b is not None:  
 break  
 i += 1  
  
 return [a, b]  
  
  
def encode\_quadratic(text, alphabet, a, b, c):  
 encoded\_text = []  
 for p in range(len(text)):  
 if text[p] in alphabet:  
 key = (alpha\_ind(text[p], alphabet) + a \* (p \*\* 2) + b \* p + c) % len(alphabet)  
 encoded\_text.append(alphabet[key])  
 else:  
 encoded\_text.append(text[p])  
 return "".join(encoded\_text)  
  
  
def decode\_quadratic(text, alphabet, a, b, c):  
 decoded\_text = []  
 for p in range(len(text)):  
 if text[p] in alphabet:  
 key = (alpha\_ind(text[p], alphabet) - (a \* (p \*\* 2) + b \* p + c)) % len(alphabet)  
 decoded\_text.append(alphabet[key])  
 else:  
 decoded\_text.append(text[p])  
 return "".join(decoded\_text)  
  
  
# # Linear trithemius cipher  
# str\_1 = "HARD AS A ROCK"  
# print("Original string: " + str\_1)  
# A = ord("A")  
# alpha = "".join([chr(x) for x in range(A, A + 26)])  
# encoded = encode\_linear(str\_1, alpha, 5, 4)  
# coefs = linear\_coefficients(encoded, str\_1, alpha)  
# print(f"Coefficients (a and b respectively): + {coefs}")  
# print("Encoded string: " + encoded)  
# LJFW DA S TVOB  
  
  
# Quadratic trithemius cipher encoding  
# str\_2 = "London is the capital of Great Britain".upper()  
# A = ord("A")  
# alpha = "".join([chr(x) for x in range(A, A + 26)])  
# a = 2  
# b = 2  
# c = 2  
# encoded = encode\_quadratic(str\_2, alpha, a, b, c)  
# print(f"Encoded: {encoded}")  
# print(decode\_quadratic(encoded, alpha, a, b, c))  
# NUBDEX SI HNG IOPYDIV OT ITKOT LZSJAWT  
  
  
# Linear decoding  
# A = ord("A")  
# alpha = "".join([chr(x) for x in range(A, A + 26)])  
# original = "encoded text".upper()  
# encoded = "julclekxy sfay".upper()  
# coefs = linear\_coefficients(encoded, original, alpha)  
# print(f"Coefficients: {coefs}")  
# coefs [2,5]  
  
  
# Quadratic trithemius cipher decoding (brute force)  
  
# str = "wkil rnf cgsgoirncm".upper()  
# A = ord("A")  
# alpha = "".join([chr(x) for x in range(A, A + 26)])  
# print("-----------")  
# for a in range(1, 6):  
# for b in range(1, 6):  
# for c in range(1, 6):  
# print(f"Decoded (a={a}, b={b}, c={c}): {decode\_quadratic(str, alpha, a, b, c)}")  
# print("--------")  
  
# result: Encoded (a=1, b=2, c=3): TEXT FOR ENCRYPTION