Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Фізико-технічний інститут

ЛАБОРАТОРНА РОБОТА №3

з дисципліни

«Криптографія»

на тему: «Криптоаналіз афінної біграмної підстановки»

Виконали:

студенти 3 курсу ФТІ

групи ФБ-74

Заіграєв Костянтин та Новіков Олексій

Перевірили:

Чорний О.

Савчук М. М.

Завадська Л. О.

Мета роботи :

Засвоєння методів частотного криптоаналізу. Здобуття навичок роботи та аналізу потокових шифрів гамування адитивного типу на прикладі шифру Віженера.

Порядок виконання роботи

0. Уважно прочитати методичні вказівки до виконання комп'ютерного практикуму.

1. Реалізувати підпрограми із необхідними математичними операціями: обчисленням оберненого елементу за модулем із використанням розширеного алгоритму Евкліда, розв'язуванням лінійних порівнянь. При розв'язуванні порівнянь потрібно коректно обробляти випадок із декількома розв'язками, повертаючи їх усі.

2. За допомогою програми обчислення частот біграм, яка написана в ході виконання комп'ютерного практикуму №1, знайти 5 найчастіших біграм запропонованого шифртексту (за варіантом).

3. Перебрати можливі варіанти співставлення частих біграм мови та частих біграм шифртексту (розглядаючи пари біграм із п'яти найчастіших). Для кожного співставлення знайти можливі кандидати на ключ) ,(b a шляхом розв'язання системи (1).

4. Для кожного кандидата на ключ дешифрувати шифртекст. Якщо шифртекст не є змістовним текстом російською мовою, відкинути цього кандидата. 5. Повторювати дії 3-4 доти, доки дешифрований текст не буде змістовним.

Результати

2. **Найчастіші біграми**

Для мови	ст	но	то	на	ен
Для шифротексту	цл	ял	ае	ле	чо

3. **Можливі варіанти ключів**

(64, 603), (110, 817), (53, 724), (605, 841), (165, 755), (273, 863), (299, 321), (694, 136), (769, 832), (25, 842), (605, 625), (413, 755), (540, 732), (234, 569), (127, 98), (192, 836), (761, 387), (269, 651), (848, 262), (910, 138), (834, 941), (796, 532), (544, 201), (453, 438), (267, 562), (126, 882), (902, 718), (578, 715), (342, 809), (619, 884), (113, 684), (604, 448), (569, 200), (183, 941), (73, 851), (600, 138), (415, 6), (508, 849), (417, 497), (348, 598), (700, 744), (838, 158), (853, 4), (936, 231), (732, 602), (240, 589), (108, 694), (426, 124), (394, 476), (438, 290), (939, 222), (463, 405), (275, 36), (721, 450), (901, 665), (284, 634), (662, 138), (405, 882), (958, 760), (937, 620), **(200, 900)**, (258, 423), (3, 933), (641, 191), (356, 448), (261, 329), (688, 557), (494, 312), (581, 154), (59, 228), (421, 0), (306, 210), (692, 769), (467, 14), (888, 569), (556, 786), (728, 663), (12, 880), (286, 505), (606, 294), (523, 656), (361, 935), (51, 594), (851, 851), (736, 558), (191, 634), (425, 476), (264, 562), (413, 516), (466, 372), (250, 882), (348, 837), (506, 180), (498, 293), (898, 459), (732, 841), (123, 915), (357, 284), (118, 501), (229, 105), (535, 574), (233, 283), (355, 652), (431, 791), (430, 36), (613, 109), (835, 191), (949, 159), (889, 532), (251, 98), (229, 130), (22, 724), (368, 656), (455, 527), (770, 73), (655, 249), (24, 453), (866, 811), (531, 290), (411, 264), (356, 446), (897, 436), (843, 231), (721, 584), (686, 42), (567, 222), (888, 808), (95, 882), (908, 563), (530, 282), (550, 62), (73, 138), (957, 24), (392, 839), (72, 507), (703, 309), (60, 281), (697, 477), (262, 745), (710, 228), (536, 563), (711, 538), (380, 553), (225, 388), (778, 718), (4, 683), (495, 87), (677,

786), (699, 294), (546, 106), (63, 248), (548, 191), (383, 572), (727, 851), (320, 507), (240, 836), (593, 42), (613, 856), (548, 557), (675, 915)

3. Шифрований текст

хетжщбеыжцллйшллебторюкечожлхуемебсфбпвгщпсакюбизыщллбющцжбщвлвачоофлеымюэвцфйжлцщвлиффечозуазщмвьпфйбсфашазлев
лазлевлыюфйгблфубфефцинютошрлбыццошшйьтоюшцхоаимжоцллйшллебктяфлеьабуазгбшйьтошюййчажофщйленефцинебгбгугфязащзещ
бйяхенефцинебуццбхнюеоицсфоэбохзьтяфебчфкеаесачсюэбнцдвцпащйлежцаечйхцусфююющцхожцаехпщлобуипылцщмвыыйлештьйбныэ
несазпюдуипыкнялкллйешщщвлифаоыэыюфйгблфубцлццсфлщулбэйекфрлмнйехеонялйьпазагблцаьццзеяаюебияоаефцинбоьасфюэфюульукб
шеьтчлоюаехулбцьдмэбрлютошюэопсфхйуллййууляльувеаечойлфеяйчэтимжыйшйщлтечоглжюфймимкйеёжйыфтцултэуоэечоаечяифмфсо
сакбщблетипчьаьтобшифцхбялчюфййлфеяйчэусасьйдмчоюэйеьтнфлфцфчйофтцссасифылкцрлфлчлвсофртбибнпалйхзжйлеэыаурсзэшцилми
пайеымопсафыщцтиксуфйшиллцйноццфхомбобячюэубмильыбошньхйллцрксифрлвлсщзежцялильоусрлгещфйяхепьтюзежцлуялямчпрлцлыщ
ялшйвтцллевььбйуйшцфаууспяолпэпрбиксаегвпаусубшйьтошньдмэбрлрврининысрлчюшцхоаимжпфшйащцфниасчлчйжйэаюэчокбофлйхзжй
ебгбгоаежймоьаялщбифжаубчбхйвьзэбисазпфюжцчьсаьвчомйбчиесачсптялгьбщвлифшйояпапршйвтцллебноцфюэсззыцлюуйльэдглцнччбх
нялжхвбрижэчбллтньаоцкффулеьаусзымуусуэиивгмуьаьаюейнсдяэешыумеиелцчяйщдтсфашвидмгбвиччмуююажфбсфдюцноцдпфжчйжйлзь
сжффйлжчхялеихоинюеоиццвбюйшйляфюмивцвбйтчулйяцхожцаелеасуэяфллкотипчыэымаечойлфезамкаьсажлафчуещзешщксьлгйсэйщжйс
юзащмибхссачсптжлпефцщмвьбтрлцизаялхифюцлдюцццфютошшйьтбыццошьйилшмчуомэбалилоюююеьаялилгйжиоцгонтнцдфщбкечоксюэя
фнцюжкюмиасюэююцлзшдюзэщавцвююййэейгйофрлбфебошмфгфмюзэымебмфшизыяннзнтжллздйфаеэусююфймиййщбчаюэшавцчсубило
жхоюйгугфазлевльафюллшйэбсфаюййшйщлыйвикюфййтхйюйсфчьдмэбцщщфэапыююзаьтльаозачлоюаеюэелютошхаажллбияожумйбтгбцщ
зэдгьйымдтлзьбрцщидпаещгрлбфебзтжллзгфчбмюыйвиелтаеэеыжцацфуяэеылэюеччщбкеаеебшэдуфуеуцлобфпейжлгблбофошулхашчянялаз
ултайьюелэуэщмымдтчуошбияофютамжасасыумйбтлцлфлйаэчоллвлосзйлежцьййфысоцобгбфечопурзвэщаьттайьеэоцчллитснлщбазэблцссе
бйэетаегмвьобыючйюнхепйгбилхнкниелэфжкюлщьахутаоццльйдсщфкбошьййшктлцулщлнфтцйхклююфйцщдмьйещшцялвсхечойлфеяйвбюэ
лщвьклмфоюфйхашчфжщбяфялцльййлйеьтяллбялгесачслщщфйтюфьбюещмаечоялхйьйбэпчллюэвьпаопнайййавтюебюйьбсфнцьййбтщвьл
екюьаллвлйлжечовфвфдэщаулпозавьчуйэнчзэмуулйлйшщйыимжгбцалщчунцллжйгщопнчзафлилффсучуйюклщлмфйшоффпсфесшцфюфйсп
сфесаечомимкзанйбуиляcрбхутаоцьйаювььайэщмымэбтопчюеаехсбнйеуувихевюаькфсжзаццуэасхерюяйтцссасетялуицжщбыюсащбчлтцвгкб
рлципыйеьтыимжчбпфыьоцэигбхуднююлщвлфлдчзаялилцирюетмулемфллжлпфцлуичьуэкюццфывбцфжазэдгсумйбтнлнэымсаюечоццошйэн
чзэобвбллвсэбюпсафыыэемшйьйззийешклошмиццофгбтеебрйглдсвлььдмхзялхйилхйешулгоаежйошфьгужлтюжйттхутаоцазялйшллбифжщф
гййшцлтззсчутэкьносайэнчзэобобфпщэюеасцлфйшноцщбьйжлднзашцнеелуичоцлтюаечлялципыйеьтьтйэымюэмптфюэсфешгбдоьиьаьтусю
ючуфечофлялжлажаоьаьтвевьечйщриццвбнцопыихеэтжллзулыйьэщаьтпулекюьаьтщбцихечьдмэбвжоцхзнцльеэастиялмсйрчуобжеиекьрифб
ошьтялафцщбцщфйюэфкцоюыэнзвссфмсзэщаътщбьйжллщвлгфчутэмжхоюдюэфщксхеьавцшщаеебыймбебеееташйеяжйьгугьгуйбьйчэюеофб
нховидмчоьйхулбошюювидмобхйтцыюфйклвлхлчбкеоцхзмсбщаеоцфюобьйцщдмчуэбщбнййбщысдчлтзэюаеюэмжйрюйлечуэбэребмаьаоцф
ыыиксфюксгуюфьйфйяйлйлэрулзуледгдйюйофмикюрютацпяацсащаеяллбдмвььахутаоцущымцпночлэебцвбщлжлмтзлвцсаюэвьдмэбрлчрьб
цфгпебвбшийщллллевцчуюйжлолофгбмйоайесачсшцрийяассаеьавцпьчьгызаолмбрлаювцялбэасюэчяхутаоцтсебщсдгбиолдсшзщлмфнмэбпю
видмлщзелэкнщмфюаеюэфюфйауоюфйобпйленебнцлымвлбзагницнксвцулсфкцллжлтамжасаетиагялхйялйшллветиоцшинаьтемдтмфоюажао
юйофзэуэщмфюущтигоаежййюцщеоыиолэщюэшачльняльйоьуэцлбирыщлдэхоефгйчйсшшцвбьйтцацофафччыэусымчбщмюэйщксзэюецчюю
дгулхулбщлэщзазяеёжлвипчзаыицжфюнтщбаююебцмихойепалэдгшифюцдялаэксщлмсзэтюаьчоымнвэбйббинчшйьйпфчбпэымелциюеыэцлж
люцриозянвгхйкенвэблсчоиейщришщщфьтйбошщбыйьэшцошвьцлкитсдгюэлцзйийлевцгфьбфечоуэшцфюфйщждпнаюэхооллетипчцлулмиы
мзааююехктйьтзауоцбйшпзэафюцлклгйнцбтошчйюнхемуулялощвьбтсфрщфюгыьфымдтшйнцфюфйюцялвлжюзщбцщффечоьлдсзэщаьтщбцщ
зээюфймктюцжшйеитстсьйнйубафгйчйсшыйашюэазщлнэсмсафюсмэбгбкупебноцфюейхсчлпшйлхэгящэтюаьчодгшийщшцщфымдтлзьбсфил
леозйтчуаулитсефцбйшпзгыыэоцыйаумийщюезашифюкюксебйэагиасмхзвевьдмцщвлиффечожлщлфйжлфлцлифнцзебнцлрлнэмжпаыэымцжн
ощйщлжллллщлйяьлдскьхеэфжщвьчллзчбюйгффбвбзэлеыйпэагулсьдмэбрлкыйщвбсфашмикюблцлфулеьагумуолеуцфсфрщпывцхзыэпчнзем
шйялчбюйллтеыэофйэпрауиаьрщйыййшчуащзазяноююпыйробьюеблпфщбщюэьпрзаыйгйфоцлжлиююензпоыичйжйсфьбнцыюницноБеебн
цлчйлешзисиицфысюючибцвбйшцлкыиипаццвцхзсрсаязазсфбцнтнзиююеночпкьялшцллясновцсалятфымьйпэуоусцлмснозэмпбщелуичоцлй
улщвлгфчуялйущжрижэмпшцчлчбмййшобсфоегяыебнцлобйуьсчпвлщйпэейщлэсдггуллщьдмэбтыаоцллзшбнцыюжджцпблкцщлрэфюлзнбжа
аьюююьпрзааанозефцинфщлйсфлежйллжцвлофцзьбсфчлэьыробихюйжещфлйхстйебнийьюьцщзенефцинчлщлхсфющбжлщлыйхутаоцкюоиш
цсфщбвьчуафжаолпэсдшдпокюобоьчощмжуойлейнзалщшцрийяиежйошиссаaеьашцвюжлоажбщблептийщмщаллырзафюуикьйепапччьцллзш
бщлафжаолщбафулмтфщебфьечююдгхулбцьдмэбцщжйоьнткеымюэфжщбэбилхйцфшэзачбьйулбэюлжюжсщфдэщакелщвлфйутебэьгужлщьц
цзеагщбфлхйбипчьащбаююечяпапрдюыэчыыбошюйцфофлйжлщйжшгфьйулофлбсффьгубеасетщбклщсчьебнцацялилафчлщемпюеиеулсьрит
ючоаюриобзктйалщлхзяноцоиаьлллензлрвлцлмймивжкювьчощбфйчшшйьтлаюехугьцняллеопдгхутаоцажчбцьуснгчлаажцуимпяллбчлнбшсе
фюэююцлзшулобцфсфашулдйфааьдмхзцщзеьаыщзеьаюэусхеьавцшщшцчаюэырсалбрлтээйжлжйщлжээромэбгбвфчлэеиебнцлшцллясьабоюэ
цлхйфеымшцгйчаьлофгблзнбсфашобшикпрюкпвлщйпэейажчллзэрураезальшцюйэьдмхзьйжйгэщаьтьйчэюеьйчэюехутаоцгбдоьиьаьттехоцлг
йттлсшцауцбсфютчэвцхзпаглцбяьыечойерыюечообсщаечлхоэюофйюдмфшэядржщбшевеелофйфознтйэцжщбшехежсасхулбжаглцбяьщэццрзо
элщвлвальзафытюаееоефтамжыухйьйилтаажуэбопуэааьйебртизыопыеасбизыйщвбцьдмэбяртизыопрюфйшээаолцлдсшзрмэбгбаечяулюпбныэ
ацжцялшаозрллщбнэбовцмйшаьтжлщльбщмжулфебэбашйжпаыэсавцхзхэмпшцчллмэбгбэрлцмиюеьавцпьдмэбаюлежйчйчунцтщбщмжщбю
еулбтщлжюфйвбгфазщлгбхулбцлюэщбвлаэлщщфыйшэоцфдчбвеопюхялрлбэдгэайньашцтлсепчтюгбжлчлжйюэлцфжщбюейейщритлижщбюе
веэфнухйшэядйлйшшбщитсьлфулеьаоцжцозрлхзфщвьбтэбщмфючлэыырулобяфьбацлэуэюеыйяьпрцлйбэончхуаешлафялилафеяиибщуэнзмю
лежйцбпэаглцзйийзыиизыыэнзщьдмэбрллетипчцлулмиымзааюэфщлжеолофазсфобзнччтйвцкьфююйоютиыэтйилхаажчузимжбфауфмцпущае
чойжтаеэщащбаеыбзэхечоетульйсулцтюаьебхсмжзаюэфйжлнэцткliувьзэлцюедкйетлофгбйбфлаэжугосрчусфашолыьзатййуыйвичьдмэбдцял
шаиуошулобяфьбацкфщмюэзыкюццкфлеисядыфрцксчоюйрлщегмююзаяййугугфклиуулиулцоюфюхевюфйвеаесачсчопчцлхулбщлербноулех
ебрбннллйжшбцвбошьййшктгбазошоффйжлнэеэажкюмиуэфщшцюйюэщйщлгшеэыэнзрцшцчлвгйтхйщлхэзывгмжуэбоаанафщлйрзажбщйрм
фллжлпфцлуичьтфшцюйлфгййшцлпаюеюэбщзазяйрлцфунбсфхаечыэнзхоцжсаыитсольмйсфолкцулхзобнцзеасвеелгйхьечццщюхеьащмцжб
щюйзльйщбфлбиоптиилвбцьдмэбьтофлйжлмллакнцлщщебдасцщйийфлципрулхноцьлцлеузбзитснозэымновцлфцлчеебшуустиофоббэжфллгув
ешцщлрэелещянхезавцлэяйжлгйюйулэйбэымнлецянхекскеаелеыизаьтвбшабцллйшгбцьдмэбтыпальаозаопкечодпебцфилхнзаююагаечявафщ
цжчьфщжйфллекюдтрййувьцлйубисасмхешщиежцьюцжяацддэйщебфьлщвьопцлсяпаусхлдцисаетиййбиююаьюеэропебчэфюжлвлмфчлхмтив
ьтеаехйшйжштйийвьцлаешифюыэтйшйхуьсоцялшащбнфвллложиичьцлнсшзйшэййебнцлоблфвбцлтайьрюзанфвлгфыэаьпфкэейбишцялшзчй
жйнэбоебхсзэщашцяаюеелжюлщвлшбйююеризэаьшцщфйфилозрлллыэмпэфьуфбвсдмшйлептсфхутаоцйечоююлщвлшбсфялйшллщмелнэымв
ьаьпыобюэпухйрлнипальаьпыобулхсжйпщвьйлвлфлсщзежцаехзьткбчбхйдююефцинзэкюрибтобчбчбклвлнфюувлфбрцопыихеяащмллрлнйщ
фгйлцйщэбиушйьтошэйсефюгбобоьагмйхлрсаетиагозбизэццюеисбицщсуьиюб

Ключ: a = 200

b = 900

Розшифрований текст

атызнаешьсколькоразмывэтомгодуиграливбейсболавпрошломавпозапрошломнистогониссегоспросилтомгубыегодвигалисьбыстробыстроявс
езаписалтысячпятьсотшестьдесятвосемьразасколькоразячистилзубызадесятьлетжизнишестьтысячразарукимылпятнадцатьтысячразспалчеты
реслишнимтысячиразиэтотольконочьюиселшестьсотперсиковивосемьсотяблокагрушвсегодвестиянеоченьтолюблюгрушичтохочешьспросиу
менявсезаписаноесливспомнитьисосчитатьчтояделалзавседесятьлетпрямотысячимиллионовполучаютсявотвотдумалдугласопятьоноближепо
чемупотомучтотомболтаетноразведеловтомеонвсетрещititрещитсполнымртомотецсидитмолчанасторожилсякакрысьатомвсеболтаетникакн
еугомонитсяшипитипенитсякаксифонссодовойкнигяпрочелчетыресташтуккиносмотрелитогобольшесорокфильмовсучастиембакаджонсатри
дцатьсджекомхоксисорокпятьстомоммиксомтридцатьдевятьсхутомгибсономстодевяностодвамультипликационныхпрокотафеликсадесятьсду
гласомфербенксомвосемьразвиделпризраквопереслономчаничетыреразасмотрелмилтонасиллсадажеодинпролюбовьсадольфомменжутолько
ятогдапросиделцелыхдевяносточасоввкиношнойуборнойвсеждалчтобэтаерундакончиласьипустиликошкуиканарейкуилилетучуюмышьаужту
твсецеплялисьдругздружкуивизжалидвачасабезпередышкииселзаэтовремячетыресталеденцовтристатянучексемьсотстаканчиковмороженог
отомболталещедолгоминутпятьпокаотецнепрервалегоасколькоягодтысегоднясобралтомровнодвестипятьдесятшестьнеморгнувглазомответил
томотецрассмеялсяинаэтомоконичилсязавтраконивновьдвинулисьвлесныетенисобиратьдикийвиноградикрошечныеягодыземляникивсетроена
клонялиськсамойземлерукибыстроиловководелалисвоеделоведравсетяжелелиадугласприслушивалсяидумалвотвотоноопятьблизкопрямоуменя
заспинойнеоглядывайсяработайсобирайягодыкидайвведрооглянешьсяспугнешьнетужнаэтотразнеупущунокакбыегозаманитьпоближечтобып
оглядетьнанегоглянутьпрямовглазакакауменявспичечномкоробкеестьснежинкасказалтомиулыбнулсяглядянасвоюрукуонабылавсякраснаяот
ягодкаквперчаткезамолчичутьнезавопилдугласнонеткричатьнельзявсполошитсяэхоивсеспугнетпостойкатомболтаетаоноподходитвсеближез
начитононебоитсятоматомтолькопритягиваетеготомтоженемножкооноделобылоещевфевралевалилснегаяподставилкоробоктомхихикнулпой
малоднуснежинкупобольшеираззахлопнулскорейпобежалдомойисунулвхолодильникблизкосовсемблизкотомтрещалбезумолкуадугласнесво
дилснегоглазможетотскочитьудратьведьиззалесанакатываетсякакаятогрознаяволнавотсейчасобрушитсяираздавитдасэрзадумчивопродолжал
томобрываякустдикоговинограданавесьштатиллинойсуменяуодноголетоместьснежинкатакойкладбольшенигденесыщешьхотьтреснизавтраяе
еоткроюдугтытожеможешьпосмотретьвдругоевремядугласбытолькопрезрительнофыркнулнудамолснежинкакакбынетакносейчаснанегомчал
осьтоогромноевотвотобрушитсясясногонебаионлишьзажмурилсяикивнултомдотогоизумилсячтодажепереставалсобиратьягодыповернулсяиуст
авилсянабратадугласзастылсидянакорточкахнукактутудержатьсятомиспустилвоинственныйкличкинулсянанегоопрокинулназемлюонипокат
илисьпотравебарахтаясьситузядругдруганетнетниочемдругомнедуматьивдругкажетсявсехорошодаэтастычкапотасовканеспугнуланабегавшу
юволнувотоназахлестнулаихразлиласьшироковокругинесетобоихпогустойзеленитравывглубьлесакулактомаугодилдугласупогубамвортустал
огорячоисолонодугласобхватилбратакрепкостиснулегоионизамерлитолькосердцаколотилисьдадышалиобасовистомнаконецдугласукрадкой
приоткрылодинглазвдругопятьничеговотоноветутвсекакестьточноогромныйзрачокисполинскогоглазакоторыйтожетолькочтораскрылсяигля
дитвизумлениинанеговупорсмотрелвесьмирионпонялвотчтонежданнопришлокнемуитеперьостанетсяснимиуженикогдаегонепокинетяживой
подумалонпальцыегодрожалирозовеянасветустремительнойкровьюточноклочкиневедомогофлагапрежденевиданногообретенноговпервыече
йжеэтофлагкомутеперьприсягатьнаверностьоднойрукойонвсеещестискивалтоманосовсемзабылонемиосторожнопотрогалсветящиесяалымпа
льцысловнохотелснятьперчаткупотомподнялихповышеиогляделсовсехсторонвыпустилтомаоткинулсянаспинувсеещевоздеврукукнебесамите
перьвесьонбылоднаголовaглазабудточасовыесквозьбойницыневедомойкрепостиоглядывалимоствытянутуюрукуипальцыгденасветутрепетал
кровавокрасныйфлагтычтодугспросилтомголосегодоносилсяточносодназеленогозамшелогоколодцаоткудатоизподводыдалекийитаинственн
ыйподдугласомшепталисьтравыонопустилрукуиощутилихпушистыеножныигдетодалековтеннисныхтуфляхшевельнулпальцамивушахкаквра
ковинахвздыхалветермногоцветныймирпереливалсявзрачкахточнопестрыекартинкивхрустальномшарелесистыехолмыбыliусеяныцветамиб
удтоосколкамисолнцаиогненнымиклочкаминебапоогромномуопрокинутомуозерунебосводамелькалиптицыточнокамушкиброшенныеловкой
рукойдугласшумнодышалсквозьзубыонсловновдыхалледивыдыхалпламятысячипчелистрекозпронизываливоздухкакэлектрическиеразрядыд
есятьтысячволосковнаголоведугласавырослинаоднумиллионнуюдюймавкаждомегоухестучалопосердцутретьеколотилосьвгорлеанастоящеег
улкоухаловгрудитележаднодышаломиллионамипоряиправдаживойдумалдугласпреждеяэтогонезналаможетизналданепомнюонвыкрикнулэто
просебяраздругойдесятыйнадожепрожилнасветецелыхдвенадцатьлетиничегошенькинепонималивдругтакаянаходкадралсястомомиветтебету
тподдеревомсверкающиезолотыечасыредкостныйхронометрсзаводомнасемьдесятлетдугдачтостобойдугласиздалдикийвопльсгребтомавохап
куионивновьпокатилисьпоземледуттыспятилспятилоникатилисьпосклонуходмасолнцегорелоунихвглазахивортуточноосколкилимонножелто
гостеклаонизадыхалиськакрыбывыброшенныеизводыихохоталидослездугтынерехнулсянетнетнетнетдугласзажмурилсявтемнотемягкоступал
ипятнистыелеопардытомитишетомкакпотвоемувселюдизнаютзнаютчтоонижывыеясноснаютатыкакдумаллеопардынеслышнопрошлидальшев
отьмуиглазауженемоглизанимиуследитьхорошобытакпрошепталдугласхорошобывсезналионоткрылглазаотецподбоченясьстоялвысоконадни
мисмеялсяголоваегоупиралаьвзеленолистыйнебосводглазаихвстретилисьдугласвстрепенулсяпазнаетпонялонвсетакибылозадуманооннаро
чнопривезнассюдачтобыэтосомнойслучилосьонтожевзаговореонвсезнаетитеперьонзнаетчтоияужезнаюбольшаярукаопустиласьсвысотыипод
нялаеговвоздухпокачиваясьнанетвердыхногахмеждуотцомитомомисцарапанныйвстрепанныйвсеещеошарашенныйдугласосторожнопотрогал
своилоктионибыликакчужиеисудовлетворениемоблизнулразбитуюгубупотомвзглянулнаотцаинатомаяпонесувсеведрасказалонсегодняяхочу
одинвсетащитьонизагадочноусмехнулисьиотдалиемуведрадугласстоялчутьпокачиваясьиегоношавесьистекающийсокомлесоттягивалаемурук
ихочупочувствоватьвсечтотолькоможнодумалонхочууститьхочуоченьустатьнельзязабытьнисегоднянизавтранипослеоншелопьяненныйсосво
ейтяжелойношейазанимплылипчелыизапахдикоговиноградаиослепительноелетонапальцахвспухалиблаженныемозолирукионемелиионспоты
калсятакчтоотецдажесхватилегозаплечоненадопробормоталдугласяничегояотличносправлюсьещедобрыхполчасаонощущалрукаминогамисп
инойтравуикорникамниикоручтословноотпечаталисьнаеготелепоцемногуотпечатокэтотстиралсятаялускользалдугласшелидумалобэтомабрат
имолчаливыйотецшлипозадипредоставляяемуодномупролагатьпутьсквозьлескнеправдоподобнойцеликшоссекотороеприведетихобратновгор
одивотгородвтотженьеиещеодноотровениедедушкастоялнашироком парадномкрыльцеиточнокапитаноглядывалширокиенедвижныепросто
рыпереднимраскинулосьлетоонвопрошалветеринедостижимовысокоенебоилужайкугдестоялидугласитомивопрошалитолькоегоодногодедуш
каониужесозрелидедушкапоскребподбородокпятьсоттысячадажедветысячинавернякададахорошийурожайсобиратьлегкособеритевсеплачуде
сятьцентовзакаждыймешоккоторыйвыпринесетекпрессуураа

Код програми

```
import math, re
from operator import itemgetter      #  For dict reversing,
from itertools import groupby        #  without losing n-gramms
from pprint import pprint

snd = itemgetter(1)
__count=0
__entropy=0
__unique=0



rus_dict=['а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м',
          'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ',
          'ь', 'ы', 'э', 'ю', 'я']

__lenOfRing=pow(len(rus_dict), 2)

__alphabet_dict={'о':0.10983, 'е':0.08483, 'а':0.07998, 'и':0.07367, 'н':0.067,
```

```
            'т':0.06318, 'с':0.05473, 'р':0.04746, 'в':0.04533, 'л':0.04343,
            'к':0.03486, 'м':0.03203, 'д':0.02977, 'п':0.02804, 'у':0.02615,
            'я':0.02001, 'ы':0.01898, 'ь':0.01735, 'г':0.01687, 'з':0.01641,
            'б':0.01592, 'ч':0.0145, 'й':0.01208, 'х':0.00966, 'ж':0.0094,
            'ш':0.00718, 'ю':0.00639, 'ц':0.00486, 'щ':0.00361, 'э':0.00331,
            'ф':0.00267, 'ъ':0.00037}#, 'ё':0.00013}

__standart_dict={}

def polygramms_freq(filename, fileEncoding, num, step=1, specCharAsSpace=True, countSpace=True):
    global __count, __unique
    res_dict={}
    polygramm_count=0
    polygramm_str=''
    with open(filename, encoding=fileEncoding) as file:
        for line in file:
            line=line.strip()
            line=re.sub(' +', ' ', line)
            line=line.replace('Ъ', 'Ь')
            line=line.replace('ъ', 'ь')

    #        print("Ptint COUNT BM: ",line.count('вм'))

            for i in range(len(line)):

                if ( ((line[i] in rus_dict) ==False and line[i]!=' ') or (line[i]==' ' and countSpace==False )):
                    if specCharAsSpace==False or countSpace==False:
                        continue
                    polygramm_str+=' '
                else:
                    polygramm_str+=line[i].lower()

                #polygramm_str=re.sub(' +', ' ', polygramm_str)

                if len(polygramm_str)==num:
                    try:
                        res_dict[polygramm_str]+=1
                    except:
                        __unique+=1
                        res_dict[polygramm_str]=1
                    polygramm_str=polygramm_str[step:]
                    polygramm_count+=1


    __count=polygramm_count

    #print(__count)
    for key in res_dict.keys():
    #    print(key, '\t', res_dict[key], '\t', polygramm_count, '\t', ((float)(res_dict[key]))/((float)(polygramm_count)))
        res_dict[key]=((float)(res_dict[key]))/((float)(polygramm_count))
    return res_dict

def find_entropy(freq_dict, n):
    res=0
    for elem in freq_dict:
        res+=(elem*math.log(elem,2))*len(freq_dict[elem])
    res*=(-1/n)

    return res

def getMostPopularNgrams(_dict, maxIter):
    it=0
    resDict={}
    arr=[]
    for a in sorted(_dict, reverse=True):
        for elem in sorted(_dict[a]):
            if it==maxIter:
                return arr
            resDict[elem]=a
            arr.append(elem)
            it+=1

def getBigramNum(bigram):
    return (rus_dict.index(bigram[0])*len(rus_dict))+rus_dict.index(bigram[1])

def getBigramStr(bigram):
    return rus_dict[bigram//len(rus_dict)]+rus_dict[bigram%len(rus_dict)]

def find_keys(x_list, y_list):
    res=[]
    lenOfRing=pow(len(rus_dict), 2)
    deltaY=(getBigramNum(y_list[0])-getBigramNum(y_list[1]))
    deltaX=(getBigramNum(x_list[0])-getBigramNum(x_list[1]))%lenOfRing
    if (math.gcd(deltaY, lenOfRing)>=1 and deltaX % math.gcd(deltaY, lenOfRing) == 0):
        d=math.gcd(deltaY, lenOfRing)
        a1=deltaY/d
        b1=deltaX/d
        n1=lenOfRing/d
        x0=(b1*inverse(a1, n1))%n1
        for i in range(d):
            aInversed=x0+i*n1
            a=inverse(aInversed, lenOfRing)
            if a==None or math.gcd(int(a), lenOfRing)!=1:
                continue
```

```
            res.append([int(a), (getBigramNum(y_list[0])-int(a)*getBigramNum(x_list[0]))%lenOfRing])
    return res

def find_all_keys(standart_popular, encrypted_popular):
    res=[]
    st_pairs=[]
    enc_pairs=[]
    count=0
    for i in range(len(standart_popular)):
        for j in range(len(standart_popular)):
            if i==j:
                continue
            st_pairs.append([standart_popular[i], standart_popular[j]])
    for i in range(len(encrypted_popular)):
        for j in range(len(encrypted_popular)):
            if i==j:
                continue
            enc_pairs.append([encrypted_popular[i], encrypted_popular[j]])
    for i in range(len(st_pairs)):
        for j in range(len(enc_pairs)):
            res_temp=find_keys(st_pairs[i], enc_pairs[j])
            res+=res_temp
            count+=1

    return set(map(tuple,res))

def inverse(a, m):
    a%=m;
    if a == 1:
        return 1;
    try:
        return ((1 - m * inverse(m % a, a)) // a)%m;
    except:
        return;

def decrypt(filename, fileEncoding, keys):
    lenOfRing=pow(len(rus_dict), 2)
    aInv=inverse(keys[0], lenOfRing)
    b=keys[1]
    res=''
    res_dict={}
    res_alphabet_dict={}
    polygramm_count=0
    alphabet_count=0
    polygramm_str=''
    countSpace=False
    specCharAsSpace=False
    with open(filename, encoding=fileEncoding) as file:
        for line in file:
            line=line.strip()
            line=re.sub(' +', ' ', line)
            line=line.replace('Ъ', 'Ь')
            line=line.replace('ъ', 'ь')
            for i in range(len(line)):

                if ( ((line[i] in rus_dict) ==False and line[i]!=' ') or (line[i]==' ' and countSpace==False )):
                    if specCharAsSpace==False or countSpace==False:
                        print("test")
                        continue
                    print("test!")
                    polygramm_str+=' '
                else:
                    polygramm_str+=line[i].lower()

                #polygramm_str=re.sub(' +', ' ', polygramm_str)

                if len(polygramm_str)==2:
                    xNum=(aInv*(getBigramNum(polygramm_str)-b))%lenOfRing
                    res+=getBigramStr(xNum)
                    try:
                        res_dict[res[-2:]]+=1
                    except:
                        res_dict[res[-2:]]=1
                    try:
                        res_alphabet_dict[res[-2]]+=1
                    except:
                        res_alphabet_dict[res[-2]]=1
                    try:
                        res_alphabet_dict[res[-1]]+=1
                    except:
                        res_alphabet_dict[res[-1]]=1
                    polygramm_str=''
                    polygramm_count+=1
                    alphabet_count+=2

    for key in res_dict.keys():
        res_dict[key]=((float)(res_dict[key]))/((float)(polygramm_count))

    for key in res_alphabet_dict:
        res_alphabet_dict[key]=((float)(res_alphabet_dict[key]))/((float)(alphabet_count))

    '''
    max_alphabet_info = (res_dict.get('ст', 0)+res_dict.get('но', 0)+res_dict.get('то', 0))/3
    max_standart_alphabet_info = (__standart_dict.get('ст', 0)+__standart_dict.get('но', 0)+__standart_dict.get('то', 0))/3
```

```
    max_res=max_alphabet_info/max_standart_alphabet_info
    '''
    max_alphabet_info = (res_alphabet_dict.get('о', 0)+res_alphabet_dict.get('а', 0)+res_alphabet_dict.get('е', 0))/3
    max_standart_alphabet_info = (__alphabet_dict.get('о', 0)+__alphabet_dict.get('а', 0)+__alphabet_dict.get('е', 0))/3
    max_res=max_alphabet_info/max_standart_alphabet_info

    min_alphabet_info = (res_alphabet_dict['ф']+res_alphabet_dict['щ']+res_alphabet_dict['ь'])/3
    min_standart_alphabet_info = (__alphabet_dict['ф']+__alphabet_dict['щ']+__alphabet_dict['ь'])/3
    min_res=min_alphabet_info/min_standart_alphabet_info

    summ=False

    if max_res >0.9 and max_res < 1.5 and min_res >0.8 and min_res < 1.5:
        summ=True

    return [summ, res]

def find_text(keys_list):
    sums={}
    sumslist=[]
    print(keys_list)
    count=0
    for elem in keys_list:
        a=decrypt(filename='test.txt', fileEncoding='utf-8', keys=elem)
        sums[elem]=a[0]
        sumslist.append(a[0])
        count+=1
        if count%10000==0:
            print(count)

    #pprint(sorted(sumslist, reverse=True))

    inv_map = {number: [char for char,_ in v]
               for number, v in groupby(sorted(sums.items(), key=snd), snd)}


    it=0
    #needList=inv_map[True]
    #print(needList)

    for elem in sorted(inv_map, reverse=True):
        print(elem, inv_map[elem])
        for elem_ in inv_map[elem]:
            if (it==2):
                return
            print(elem_)
            print(decrypt(filename='test.txt', fileEncoding='utf-8', keys=elem_)[1])
            print("----------------------------")
            it+=1
        return



def main():
    global __entropy, __standart_dict

    __standart_dict=polygramms_freq(filename='1.txt', fileEncoding='ansi',
                        num=2, step=1, specCharAsSpace=False, countSpace=False)

    inv_map = {number: [char for char,_ in v]
               for number, v in groupby(sorted(__standart_dict.items(), key=snd), snd)}

    #__entropy=find_entropy(inv_map, 2)


    res=polygramms_freq(filename='test.txt', fileEncoding='utf-8',
                    num=2, step=2, specCharAsSpace=False, countSpace=False)

    inv_map2 = {number: [char for char,_ in v]
               for number, v in groupby(sorted(res.items(), key=snd), snd)}


    standart_popular = getMostPopularNgrams(inv_map, 5)
    #standart_popular = ['ст', 'но', 'то', 'на', 'ен']

    encrypted_popular = getMostPopularNgrams(inv_map2, 5)
    print(encrypted_popular)
    #print(inv_map2)

    keys_list=find_all_keys(standart_popular, encrypted_popular)

#    print(keys_list)

    find_text(keys_list)

    print("Done!")


if __name__ == "__main__":
    main()
```

Висновки:

Під час данного комп’ютерного практикуму, ми опанували прийомами роботи в модулярній арифметиці, та набули навичок частотного аналізу на прикладі розкриття моноалфавітної підстановки.