Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Фізико-технічний інститут

ЛАБОРАТОРНА РОБОТА №3

З дисципліни «Криптографія»

«Криптоаналіз афінної біграмної підстановки»

Виконали:
студенти 3 курсу ФТІ
групи ФБ-73
Дем’яненко Д.
Проноза А.

Перевірив:
Чорний О.

Мета роботи:

Набуття навичок частотного аналізу на прикладі розкриття моноалфавітної підстановки; опанування прийомами роботи в модулярній арифметиці.

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.

Хід роботи:

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

1. Реалізували підпрограми із необхідними математичними операціями: обчисленням оберненого елементу за модулем із використанням розширеного алгоритму Евкліда, розв'язуванням лінійних порівнянь.

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

3. Перебрали можливі варіанти співставлення частих біграм мови та частих біграм шифртексту (розглядаючи пари біграм із п'яти найчастіших). Для кожного співставлення знайшли можливі кандидати на ключ.

4. Для кожного кандидата на ключ дешифрували шифртекст.

5. Повторювали дії 3-4 доти, доки дешифрований текст не став змістовним.

5 найчастіших біграм шифртексту:

цл=51 ял=49 ае=43 ле=42 чо=39

Опис роботи автоматичного розпізнавача російської мови

Автоматичний розпізнавач російської мови перевіряє наявність серед 12 найбільш вживаних літер шифртексту 6 найчастіших літер російської мови(о а е и н т) , а також

перевіряє серед цих же 12 літер шифртексту відсутність найменш вживаних літер російської мови(щ ф ц).

На основі цього аналізу ми отримуємо найбільш вірогідні варіанти правильно розшифрованого тексту.

Значення ключа

a=200

b=900

Шифртекст

хетжщбеыжцллйшллебторюкечожлхуемебсфбпвгщпсакюбизыщллбющцжбщвлвачоофлеы
мюэвцфйжлцщвлиффечозуазщмвьпфйбсфашазлевлазлевлыюфйгблфубфефцинютошрлбы
ццошшйьтоюшцхоаимжоцллйшллебктяфлеьабуазгбшйьтошюййчажофщйленефцинебгбгу
гфязащзещбйяхенефцинебуццбхнюеоиццсфоэбохзьтяфебчфкеаесачсюэбнцдвцпащйлежца
ечйхцусфююющцхожцаехпщлобуипылцщмвьыйлештьйбныэнесазпюдуипыкнялкллйешц
щвлифаоыэыюфйгблфубцлццсфлцулбэйекфрлмнйехеонялйьпазагблцаьццзеяаюебияоаефц
инбоьасфюэфюульукбшеьтчлоюаехулбцьдмэбрлютошюэопсфхйуллййуулялйувеаечойлфе
яйчэтимжыйшйщлтечоглжюфймимкйейежйыфтцултэуоэечоаечяифмфсосакбщблетипчьаь
тобшифцхбялчюфййлфеяйчэусасьйдмчоюэйеьтнфлфцфчйофтцссасифылкцрлфлчлвсофртб
ибнпалйхзжйлеэыаурсзэшцилмипайеымопсафыццтиксуфйшиллцйноццфхомбобячюэубми
лыыбошньхйллцрксифрлвлсщзежцялильоусрлгещфйяхепьтюзежцлуялямчпрлцлыцялшйвт
цллевььбйуйшцфаауспяолпэпрбиксаегвпаусубшйьтошньдмэбрлрвринйысрлчюшцхоаимж
пфшйащщфниасчлчйжйэаюэчокбофлйхзжйебгбгоаежймоьаялщбифжаубчбхйвьзэбисазпф
южцчьсаьвчомйбчиесачсптялгьбщвлифшйояпапршйвтцллебноцфюэсзэзыцлюуйльэдглцнч
чбхнялжхвбрижэчбллтньаоцкффулеьаусзымуусуэиивгмуьаьаюейнсдязешыумеиелцчяйщд
тсфашвидмгбвиччмуююажфбсфдюцноцдпфжчйжйлзьсжффйлжчхялеихоинюеоиццвбюйш
йляфюмивцвбйтчулйяцхожцаелеасуэяфллкотипчыэымаечойлфезамкаьсажлафчуещзешщк
сьлгйсэйщжйсюзащмибхссачсптжлпефцщмвьбтрлцизаялхифюцлдюцццфютошшйьтбыццо
шьйилшмчуомэбалилоююеьаялилгйжиоцгонтнцдфщбкечоксюэяфнцюжкюмиасюэююцлз
шдюзэщавцвююййэейгйофрлбфебошмфгфмюзэымебмфшизыяннзнтжллздйфаеэусююфйм
ииййщбчаюэшавцчсубиложхоюйгугфазлевльафюллшйэбсфаюййшйщлыйвикюфййтхйюйсф
чьдмэбцщщфэапыююзаьтльаозачлоюаеюэелютошхаажллбияожумйбтгбцщзэдгьйымдтлзьб
рцщидпаещгрлбфебзтжллзгфчбмюыйвиелтаеэеыжцацфуяэеылэюеччщбкеаеебшэдуфуеуцл
обфпейжлгблбофошулхашчянялазултайьюелэуэщмымдтчуошбияофютамжасасыумйбтлцл
флйаэчоллвлосзйлежцьййфысоцобгбфечопурзвэщаьттайьеэоцчллитснлщбазэблцссебйэета
егмвьобьючйюнхепйгбилхнкниелэфжкюлщьахутаоццльйдсщфкбошьййшктлцулщлнфтцйх
клююфйцщдмьйещшцялвсхечойлфеяйвбюэлщввклмфоюфйхашчфжщбяфялцльййлйеьтялл
бялгесачслщщфйтюфьбюещмаечоялхйьйбэпчллюэвьпаопнайййавтюебюйьбсфнцьййбтщ
вьлекюьаллвлйлжечовфвфдэщаулпозавьчуйэнчзэмуулйлйшщйыимжгбцалщчунцллжйгщо
пнчзафлилффсучуйюклщлмфйшоффпсфесшцфюфйспсфесаечомимкзанйбуилясрбхутаоць
йаювььайэщмымэбтопчюеаехсбнйеуувихевюаькфсжзаццуэасхерюяйтцссасетялуицжщбы
юсащбчлтцвгкбрлципыйеьтыимжчбпфыьоцэигбхуднююлщвлфлдчзаялилцирюетмулемфл
лжлпфцлуичьуэкюццфывбцфжазэдгсумйбтнлнэымсаюечоццошйэнчзэобвбллвсэбюпсафы
ыэемшйьйззийешклошмиццофгбтеебрйглдсвлььдмхзялхйилхйешулгоаежйошфьгужлтюжй
ттхутаоцазялйшллбифжщфгййшцлтззсчутэкьносайэнчзэобобфпщэюеасцлфйшноцщбьйжл

днзашцнеелуичоцлтюаечлялципыйеьтьтйэымюэмптфюэсфешгбдоьиьаьтусюючуфечофлял
жлажаоьаьтвевьечйщриццвбнцопыихеэтжллзулыйьэщаьтпулекюьаьтщбцихечьдмэбвжоцх
знцльеэастиялмсйрчуобжеиекьрифбошьтялафцщбццфйюэфкцоюыэнзвссфмсзэщаьтщбьйж
ллщвлгфчутэмжхоюдюэфщксхеьавцшщаеебыймбебееташйеяжйьгугьгуйбьйчэюеофбнхов
идмчоьйхулбошююовидмобхйтцыюфйклвлхлчбкеоцхзмсбщаеоцфюобьйцщдмчуэбщбнййб
щысдчлтээюаеюэмжйрюйлечуэбэребмаьаоцфыыиксфюксгуюфьйфйялйлэрулзуледгдйюйо
фмикюрютацпяаццасщаасяллбдмвььахутаоцущымцпночлэебцвбщлжлмтзлвцсаюэвьдмэбр
лчрьбцфгпебвбшийщлллевцчуюйжлолофгбмйоайесачсшцрийяассааеьавцпьчьгызаолмбрла
ювцялбэасюэчяхутаоцтсебщсдгбиолдсшзщлмфнмэбпювидмлщзелэкнщмфюаеюэфюфйауо
юфйобпйленебнцлымвлбэагницнксвцулсфкцллжлтамжасаетиагялхйялйшллветиоцшинаьт
емдтмфоюажаоюйофзэуэщмфюущтигоаежййюццеоыиолэщюэшачльняльйоьуэцлбирыщлд
эхоефгйчйсшшцвбьйтцацофафччыэусымчбщмюэйщксзэюецчююдгулхулбщлэщзазяейжлв
ипчзаыицжфюнтщбаююебцмихойепалэдгшифюцдялаэксщлмсзэтюаьчоымнвэбйббинчшйь
йпфчбпэымелциюеыэцлжлющриозянвгхйкенвэблсчоиейщришщщфьтйбошщбыйьэшцошв
ьцлкитсдгюэлцзйийлевцгфьбфечоуэшцфюфйщждпнаюэхооллетипчцлулмиымзааююехктй
ьтзауоцбйшпзэафюцлклгйнцбтошчйюнхемуулялощвьбтсфрщфюгыьфымдтшйнцфюфйюця
лвлжюзщбццффечоьлдсзэщаьтщбцщзээюфймктюцжшйеитстсьйнйубафгйчйсшыйашюэаз
щлнэсмсафюсмэбгбкупебноцфюейхсчлпшйлхэгящэтюаьчодгшийщшццфымдтлзьбсфилле
озйтчуаулитсефцбйшпзгьыэоцыйаумийщюезашифюкюксебйэагиасмхзвевьдмцщвлиффечо
жлщлфйжлфлцлифнцзебнцлрлнэмжпаыэымцжнощйщлжллллцлйяьлдскьхеэфжщвьчллзчб
юйгфффвбзэлеыйпэагулсьдмэбрлкыйщвбсфашмикюблцлфулеьагумуолеуцфсфрщпывцхзы
эпчнземшйялчбюйллтеыэофйэпрауиаьрщйыййшчуащзазяноююпыйробьюеблпфщбщюэь
прзаыйгйфоцлжлиюююензпоыичйжйсфьбнцыюницнобеебнцлчйлешзисиицфысююучибцвбй
шцлкыиипаццвцхзсрсаязазсфбцнтнзиююеночпкьялшцллясновцсалятфымьйпэуоусщлмсно
зэмпбщелуичоцлйулщвлгфчуялйуцжрижэмпшцчлчбмййшобсфоегяыебнцлобйуьсчпвлщйп
эейщлэсдггуллщьдмэбтыаоцллзшбнцьюцжджцпблкцщлрэфюлзнбжааьюююьпрзааанозефц
инфщлйсфлежйллжщвлофцзьбсфчлэьыробихюйжещфлйхстйебнийьюьццзенефцинчлщлхс
фющбжлщлыйхутаоцкюоишцсфщбвьчуафжаолпэсдшдпокюобоьчощмжуойлейнзалщшцри
йяиежйошиссааеьашцвюжлоажбщблептийщмщаллырзафюуикьйепапччьцллзшбщлафжаол
щбафулмтфщебфьечююдгхулбцьдмэбцщжйоьнткеымюэфжщбэбилхйцфшэзачбьйулбэюлж
южсщфдэщакелщвлфйутебэьгужлщьццзеагщбфлхйбипчьащбаююечяпапрдюыэчыыбошюй
цфофлйжлщйжшгфьйулофлбсффьгубеасетщбклщсчьебнцацялилафчлщемпюеиеулсьритю
чоаюриобзктйалщлхзяноцоиаьлллензлрвлцлмймивжкювьчощбфйчшшйьтлаюехугьцнялле
опдгхутаоцажчбцьуснгчлаажцуимпяллбчлнбшсефюэююцлзшулобцфсфашулдйфааьдмхзц
щзеьацщзеьаюэусхеьавцшщшцчаюэырсалбрлтээйжлжйщлжээромэбгбвфчлэеиебнцлшцлля
сьабоюэцлхйфеымшцгйчаьлофгблзнбсфашобшикпрюкпвлщйпэейажчллзэрураезальшцюйэ
ьдмхзьйжйгэщаьтьйчэюеьйчэюехутаоцгбдоьиьаьттехоцлгйттлсшцауцбсфютчэвцхзпаглцб
яьыечойерыюечообсщаечлхоэюфйюдмфшэядржщбшевеелофйфознтйэцжщбшехежсасхулб
жаглцбяьщэццрзоэлщвлвальзафытюаееоефтамжыухйьйилтаажуэбопуэааьйебртизыопыеас
бизыйщвбцьдмэбяртизыопрюфйшэзаолцлдсшзрмэбгбаечяулюпбныэацжцялшаозрллщбнэ
бовцмйшаьтжлщльбщмжулфебэбашйжпаыэсавцхзхэмпшцчллмэбгбэрлцмиюеьавцпьдмэба
юлежйчйчунцтщбщмжщбюеулбтщлжюфйвбгфазщлгбхулбцлюэщбвлаэлщщфыйшэоцфдчб
веопюхялрлбэдгэайньашцтлсепчтюгбжлчлжйюэлцфжщбюейейщритлижщбюевеэфнухйшэ
ядйлйшшбщитсьлфулеьаоцжцозрлхзфщвьбтэбщмфючлэьырулобяфьбацлэуэюеыйяьпрцлй
бэончхуаешлафялилафеяиибщуэнзмюлежйцбпэаглцзйийзыииэыыэнзщьдмэбрллетипчцлул
миымзааюэфщлжеолофазсфобзнччтйвцкьфююйоютиыэтйилхаажчузимжбфауфмцпущаечо
йжтаеэщащбаеыбзэхечоетульйсулцтюаьебхсмжзаюэфйжлнэцтклиувьзэлцюедкйетлофгбйб
флаэжугосрчусфашолыьзатййуыйвичьдмэбдцялшаиуошулобяфьбацкфщмюэзыкюццкфлеи
сядыфрцксчоюйрлщегмююзаяййугугфклиуулиулцоюфюхевюфйвеаесачсчопчцлхулбщлер

бноулехебрбннллйжшбцвбошьййшктгбазошоффйжлнэеэажкюмиуэфщшцюйюэщйщлгшеэ
ыэнзрцшцчлвгйтхйщлхэзывгмжуэбоаанафщлйрзажбщйрмфллжлпфцлуичьтфшцюйлфгйй
шцлпаюеюэбщзазяйрлцфунбсфхаечыэнзхоцжсаыитсольймйсфолкцулхзобнцзеасвеелгйхье
чццщюхеьащмцжбщюйзльйщбфлбиоптиилвбцьдмэбьтофлйжлмллакнцлцщебдасццйийфл
ципрулхноцьлцлеузбзитснозэымновцлфцлчеебшуустиофоббэжфллгувешцшлрэелещянхез
авцлэяйжлгйюйулэйбэымнлещянхекскеаелеыизаьтвбшабцллйшгбцьдмэбтыпальаозаопкеч
одпебцфилхнзаююагаечявафщцжчьфщжйфллекюдтрййувьцлйубисасмхешщиежцьюцжяац
цдэйщебфьлщвьопцлсяпаусхлдцисаетиййбиююаьюеэропебчэфюжлвлмфчлхмтивьтеаехйш
йжштйийвьцлаешифюыэтйшйхуьсоцялшащбнфвлллощиичьцлнсшзйшэййебнцлоблфвбцлт
айьрюзанфвлгфыэаьпфкэейбишцялшзчйжйнэбоебхсзэщашцяаюеелжюлщвлшбйююеризэа
ьшццфйфилозрлллыэмпэфьуфбвсдмшйлептсфхутаоцйечоююлщвлшбсфялйшллщмелнэым
вьаьпыобюэпухйрлнипальаьпыобулхсжйпщвьйлвлфлсщзежцаехзьткбчбхйдююефцинзэкю
рибтобчбчбклвлнфюувлфбрцопыихеяащмллрлнйщфгйлцйщэбиушйьтошэйсефюгбобоьагм
йхлрсаетиагозбизэццюеисбиццсуьиюб

Розшифрований текст

атызнаешьсколькоразмывэтомгодуиграливбейсболавпрошломавпозапрошломнистогонисс
егоспросилтомгубыегодвигалисьбыстробыстроявсезаписалтысячпятьсотшестьдесятвосемь
разасколькоразячистилзубызадесятьлетжизнишестьтысячразарукимылпятнадцатьтысячраз
спалчетыреслишнимтысячиразиэтотольконочьюиселшестьсотперсиковивосемьсотяблокаг
рушвсегодвестиянеоченьтолюблюгрушичтохочешьспросиуменявсезаписаноесливспомнит
ьисосчитатьчтояделалзавседесятьлетпрямотысячимиллионовполучаютсявотвтдумалдугл
асопятьоноближепочемупотомучтотомболтаетноразведеловтомеонвсетрещititрещитспол
нымртомотецсидитмолчанасторожилсякакрысьатомвсеболтаетникакнеугомонитсяшипити
пенитсякаксифонссодовойкнигяпрочелчетыресташтуккиносмотрелитогобольшесорокфиль
мовсучастиембакаджонсатридцатьсджекомхоксисорокпятьстомоммиксомтридцатьдевятьс
хутомгибсономстодевяностодвамультипликационныхпрокотафеликсадесятьсдугласомфер
бенксомвосемьразвиделпризраквопереслономчаничетыреразасмотрелмилтонасиллсадаже
одинпролюбовьсадольфомменжутолькоятогдапросиделцелыхдевяносточасоввкиношнойу
борнойвсеждалчтобэтаерундаокончиласьипустиликошкуиканарейкуилилетучуюмышьаужт
утвсецеплялисьдругзадружкуивизжалидвачасабезпередышкииселзаэтовремячетыресталед
енцовтристатянучексемьсотстаканчиковмороженоготомболталещедолгоминутпятьпокаоте
цнепрервалегоасколькоягодтысегоднясобралтомровнодвестипятьдесятшестьнеморгнувгла
зомответилтомотецрассмеялсяинаэтомокончилсязавтраконивновьдвинулисьвлесныетенис
обиратьдикийвиноградикрошечныеягодыземляникивсетроенаклонялиськсамойземлерукиб
ыстроиловкоделалисвоеделоведравсетяжелелиадугласприслушивалсяидумалвотвотоноопя
тьблизкопрямоуменязаспинойнеоглядывайсяработайсобирайягодыкидайвведрооглянешьс
яспугнешьнетужнаэтотразнеупущунокакбыегозаманитьпоближечтобыпоглядетьнанегогля
нутьпрямовглазакакауменявспичечномкоробкеестьснежинкасказалтомиулыбнулсяглядяна
своюрукуонабылавсякраснаяотягодкаквперчаткезамолчичутьнезавопилдугласнонеткричат
ьнельзявсполошитсяэхоивсеспугнетпостойкатомболтаетаоноподходитвсеближезначитоно
небоитсятоматомтолькопритягиваетеготомтоженемножкооноделобылоещевфевралевалил
снегаяподставилкоробоктомхихикнулпоймалоднуснежинкупобольшеираззахлопнулскоре
йпобежалдомойисунулвхолодильникблизкосовсемблизкотомтрещалбезумолкуадугласнесв
одилснегоглазможетотскочитьудратьведьиззалесанакатываетсякакаятогрознаяволнавотсей
часобрушитсяираздавитдасэрзадумчивопродолжалтомобрываякустдикоговинограданавесь
штатиллинойсуменяуодноголетоместьснежинкатакойкладбольшенигденесыщешьхотьтрес

низавтраяееоткроюдугтытожеможешьпосмотретьвдругоевремядугласбытолькопрезритель
нофыркнулнудамолснежинкакакбынетакносейчаснанегомчалосьтоогромноевотвотобруши
тсясясногонебаионлишьзажмурилсяикивнултомдотогоизумилсячтодажепересталсобирать
ягодыповернулсяиуставилсянабратадугласзастылсидянакорточкахнукактутудержатьсятом
испустилвоинственныйкличкинулсянанегоопрокинулназемлюонипокатилисьпотравебарах
таясьитузядругдруганетнетниочемдругомнедуматьивдругкажетсявсехорошодаэтастычкап
отасовканеспугнуланабегавшуюволнувотоназахлестнулаихразлиласьшироковокругинесет
обоихпогустойзеленитравывглубьлесакулактомаугодилдугласупогубамвортусталогорячои
солонодугласобхватилбратакрепкостиснулегоионизамерлитолькосердцаколотилисьдадыш
алиобасосвистомнаконецдугласукрадкойприоткрылодинглазвдругопятьничеговотоновсет
утвсекакестьточноогромныйзрачокисполинскогоглазакоторыйтожетолькочтораскрылсяиг
лядитвизумлениинанеговупорсмотрелвесьмирионпонялвотчтонежданнопришлокнемуитеп
ерьостанетсяснимиуженикогдаегонепокинетяживойподумалонпальцыегодрожалирозовеян
асветустремительнойкровьюточноклочкиневедомогофлагапрежденевиданногообретенног
овпервыечейжеэтофлагкомутеперьприсягатьнаверностьоднойрукойонвсеещестискивалтом
аносовсемзабылонемиосторожнопотрогалсветящиесяалымпальцысловнохотелснятьперчат
купотомподнялихповышеиогляделсовсехсторонвыпустилтомаоткинулсянаспинувсеещево
здеврукукнебесамитеперьвесьонбылоднаголовваглазабудточасовыесквозьбойницыневедом
ойкрепостиоглядывалимоствытянутуюрукуипальцыгденасветутрепеталкровавокрасныйфл
агтычтодугспросилтомголосегодоносилсяточносодназеленогозамшелогоколодцаоткудато
изподводыдалекийитаинственныйподдугласомшепталисьтравыонопустилрукуиощутилих
пушистыеножныигдетодалековтеннисныхтуфляхшевельнулпальцамивушахкакврановинах
вздыхалветермногоцветныймирпереливалсявзрачкахточнопестрыекартинкивхрустальном
шарелесистыехолмыбылиусеяныцветамибудтоосколкамисолнцаиогненнымиклочкаминеба
поогромномуопрокинутомуозерунебосводамелькалиптицыточнокамушкиброшенныеловк
ойрукойдугласшумнодышалсквозьзубыонсловновдыхалледивыдыхалпламятысячипчелист
рекозпронизываливоздухкакэлектрическиеразрядыдесятьтысячволосковнаголоведугласав
ырослинаоднумиллионнуюдюймавкаждомегоухестучалопосердцутретьеколотилосьвгорле
анастоящеегулкоухаловгрудителожаднодышаломиллионамипоряиправдаживойдумалдугл
аспреждеяэтогонезналаможетизналданепомнюонвыкрикнулэтопросебяраздругойдесятыйн
адожепрожилнасветецелыхдвенадцатьлетиничегошенькинепонималивдругтакаянаходкадр
алсястомомивоттебетутподдеревомсверкающиезолотыечасыредкостныйхронометрсзаводо
мнасемьдесятлетдугдачтостобойдугласиздалдикийвопльсгребтомавохапкуионивновьпокат
илисьпоземледугтыспятилспятилоникатилисьпосклонухолмасолнцегорелоунихвглазахиво
ртуточноосколкилимонножелтогостеклаонизадыхалиськакрыбывыброшенныеизводыихох
отализослездугтынерехнулсянетнетнетнетдугласзажмурилсявтемнотемягкоступалипятнис
тыелеопардытомитишетомкакпотвоемувселюдизнаютзнаютчтоониживыеяснознаютатыка
кдумаллеопардынеслышнопрошлидальшевотьмуиглазауженемоглизанимиуследитьхорош
обытакпрошепталдугласхорошобывсезналионоткрылглазаотецподбоченясьстоялвысокона
днимисмеялсяголоваегоупираласьвзеленолистыйнебосводглазаихвстретилисьдугласвстре
пенулсяпапазнаетпонялонвсетакибылозадуманооннарочнопривезнассюдачтобыэтосомной
случилосьонтожевзаговореонвсезнаетитеперьонзнаетчтоияужезнаюбольшаярукаопустила
сьсвысотыиподнялаеговоздухпокачиваясьнанетвердыхногахмеждуотцомитомомисцарапа
нныйвстрепанныйвсеещеошарашенныйдугласосторожнопотрогалсвоилоктионибыликакчу
жиеисудовлетворениемоблизнулразбитуюгубупотомвзглянулнаотцаинатомаяпонесувсеве
драсказалонсегодняяхочуодинвсетащитьонизагадочноусмехнулисьиотдалиемуведрадугла
сстоялчутьпокачиваясьиегоношавесьистекающийсокомлесоттягивалаемурукихочупочувст
воватьвсечтотолькоможнодумалонхочуустатьхочуоченьустатьнельзязабытьнисегодняниза
втранипослеоншелопьяненныйсосвоейтяжелойношейазанимплылипчелыизапахдикоговин

оградаиослепительноелетонапальцахвспухалиблаженныемозолирукионемелиионспотыкалсятакчтоотецдажесхватилегозаплечоненадопробормоталдугласяничегояотличносправлюсьещедобрыхполчасаонощущалрукаминогамиспинойтравуикорникамниикоручтословноотпечаталисьнаеготелепоцемногуотпечатокэтотстиралсятаялускользалдугласшелидумалобэтомабратимолчаливыйотецшлипозадипредоставляяемуодномупролагатьпутьсквозьлескнеправдоподобнойцеликшоссекотороеприведетихобратновгородивотгородвтотжеденьиещеоднооткровениедедушкастоялнаширокомпарадномкрыльцеиточнокапитаноглядывалширокиенедвижныепросторыпереднимраскинулосьлетоонвопрошалветеринедостижимовысокоенеболужайкугдестоялидугласитомивопрошалитолькоегоодногодедушкаониужесозрелидедушкапоскребподбородокпятьсоттысячадажедветысячинавернякададахорошийурожайсобиратьлегкособеритевсеплачудесятьцентовзакаждыймешоккоторыйвыпринесетекпрессуур

Код

```
import java.io.FileReader;

import java.io.IOException;

import java.text.DecimalFormat;

import java.util.*;




public class Main {



    private final static int
CAPACITY = 31;



    private static Map<String,
Integer> alphabetForBigramm = new
TreeMap<>();

    private static Map<String,
Integer> indexOfLetter = new
TreeMap<>();

    private static Map<Integer,
String> letterByIndex = new
TreeMap<>();

    private static Map<Integer,
String> indexesOfBigram = new
TreeMap<>();

    private static Map<String,
Integer> bigramForIndexes = new
TreeMap<>();



    private static StringBuffer
getFileContent(String filename){

        StringBuffer fileData = new
StringBuffer();

        try(FileReader reader = new
FileReader(filename)){

            int c;


while((c=reader.read())!=-1){

                if(((c >= 1072) &&
(c<=1097))||((c >= 1099)&&(c <=
1103))||(c == 32)||((c >= 1040) &&
(c<=1065))||((c >= 1067)&&(c <=
1071))) {

                    if (((c >= 1040)
&& (c<=1065))||((c >= 1067)&&(c <=
1071)))

                        c += 32;

                    if (c == ' ') {

                        c = '0';

                        if
(fileData.charAt(fileData.length() -
1) == '0')


continue;

                    }


fileData.append((char) c);

                }

            }

        }catch(IOException ex){


System.out.println(ex.getMessage());

        }
```

```
System.out.println(fileData);
        return fileData;
    }

    private static void
initAlphabetForBigram(StringBuffer
fileData, Map<String, Integer>
alphabet, int step){
        for (int i=0;
i<fileData.length()-1; i+=step){
            String bigram =
fileData.substring(i, i+2);

            int temp =
alphabet.getOrDefault(bigram, 0);
            temp++;
            alphabet.put(bigram,
temp);
        }
    }

    private static void
printMap(String desc, Map map){
        System.out.println();
        System.out.println(desc);
        System.out.println(map);
    }

    private static void
printAlphabetMap(String desc,
Map<Character, Integer> map, int
total){
        System.out.println();
        System.out.println(desc);
        Map<Character, String>
mapFrequency = new TreeMap<>();
        for (Map.Entry<Character,
Integer> entry : map.entrySet())

mapFrequency.put(entry.getKey(), new
DecimalFormat("#0.00000").format((do
uble) entry.getValue()/total));
System.out.println(mapFrequency);

        List list = new
ArrayList(mapFrequency.entrySet());

list.sort((Comparator<Map.Entry<Char
acter, String>>) (a, b) ->
b.getValue().compareTo(a.getValue())
);
        System.out.println("letter
frquency sorted by value: " + list);
    }

    private static void
initArray(String[][] array, String[]
alpha, Map<String, Integer>
alphabet){
        for(int column = 1;
column<array.length; column++){
            array[0][column] =
alpha[column-1];
        }

        for(int row = 1; row
<array.length; row++){
            array[row][0] =
alpha[row-1];
        }
        array[0][0] = "   ";

        for (int row =1;
row<array.length; row++){
            for (int column =1;
column<array.length; column++){
                String key =
array[row][0] + array[0][column];
                String result;
                if
(alphabet.get(key) != null)
                    result =
Integer.toString(alphabet.get(key));
                else
```

```
                    result =
Integer.toString(0);

                array[row][column] =
result;

            }

        }


    }


    private static void
showArray(String[][] array, int
total){

        System.out.println();

        System.out.print("     ");

        for (int i = 0; i <
array.length; i++) {

            for (int j = 0; j
<array.length; j++) {

                if (i >=1 && j>=1) {

                    String
formattedDouble = new
DecimalFormat("#0.00").format(0.1321
231);


System.out.print(new
DecimalFormat("#0.00000").format((do
uble) Integer.parseInt(array[i][j])
/ total) + "    ");

                }

                else


System.out.print(array[i][j] + "
");

            }

            System.out.println();

        }

    }


    private static int
countAmountBigram(StringBuffer
fileData, int step){

        int result = 0;

        for (int i=0;
i<fileData.length()-3; i+=step){

            result++;

        }

        return result;

    }


    public static int
reverseElement(int a, int n) {

        int x = 0, y = 1, lastx = 1,
lasty = 0, temp;

        while (n != 0) {

            int q = a / n;

            int r = a % n;


            a = n;

            n = r;


            temp = x;

            x = lastx - q * x;

            lastx = temp;


            temp = y;

            y = lasty - q * y;

            lasty = temp;

        }

//        System.out.println("Roots
x : "+ lastx +" y :"+ lasty);

        return lastx;

    }


    public static int gcd(int a, int
b) {

        while (b !=0) {

            int tmp = a%b;

            a = b;

            b = tmp;

        }
```

```
        return a;
    }

    public static ArrayList<Integer>
resolvingEquation(int a, int b, int
n){
        ArrayList<Integer> list =
new ArrayList<>();
        ArrayList<Integer> result =
new ArrayList<>();
        if (gcd(a,n) == 1){

list.add(reverseElement(a, n)*b %
n);
            if (list.get(0)>=0)

result.add(list.get(0));
            else

result.add(list.get(0)+n);
            return result;
        } else {
            int d = gcd(a,n);
            if (b%d !=0)
                return list;
            int a1 = a/d;
            int b1 = b/d;
            int n1 = n/d;
            list =
resolvingEquation(a1,b1,n1);
            for (int i=1; i<d; i++){

list.add(list.get(0)+n1*i);
            }
            for (int i=0; i<
list.size(); i++){
                if (list.get(i)>=0)

result.add(list.get(i));
                else

result.add(list.get(i)+n);
            }
            return result;
        }
    }

    public static void main(String[]
args) throws Exception {
        StringBuffer fileData;
        fileData =
getFileContent("07.txt");

initAlphabetForBigram(fileData,
alphabetForBigramm, 2);

        int totalForBigram =
countAmountBigram(fileData, 2);
        System.out.println("total
for bigram: " + totalForBigram);

        String[][]
arrayWithoutSpaces = new
String[32][32];

        String[]
alphabetWithoutSpaces
={"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"};

initArray(arrayWithoutSpaces,
alphabetWithoutSpaces,
alphabetForBigramm);

        System.out.println("\nArray
for bigram without spaces for step =
2: ");

showArray(arrayWithoutSpaces,
totalForBigram);
        System.out.println();
```

```
//
System.out.println(gcd(29,31));


        List roots =
resolvingEquation(-67,44,961);


        System.out.println("Roots: "
+ roots);



System.out.println(alphabetForBigram
m);

        List list = new
ArrayList(alphabetForBigramm.entrySe
t());


list.sort((Comparator<Map.Entry<Stri
ng, Integer>>) (a, b) ->
b.getValue().compareTo(a.getValue())
);

        System.out.println("letter
frquency sorted by value: " );

        Map<String, Integer> max5 =
new HashMap<>();


list.stream().limit(21).forEach(Syst
em.out::println);


        for(int i=0; i<31; i++){


indexOfLetter.put(alphabetWithoutSpa
ces[i], i);

            letterByIndex.put(i,
alphabetWithoutSpaces[i]);

        }


System.out.println("indexOfLetter: "
+ indexOfLetter);



        for (int i =0; i<31; i++){

            for (int j=0; j<31;
j++){


                String bigram =
alphabetWithoutSpaces[i] +
alphabetWithoutSpaces[j];

                int index =
indexOfLetter.get(alphabetWithoutSpa
ces[i])*31 +
indexOfLetter.get(alphabetWithoutSpa
ces[j]);


indexesOfBigram.put(index, bigram);


bigramForIndexes.put(bigram, index);

            }

        }


        //TODO we need to fill in
two lists manually (real and
received most frequent bigrams)

        List<String> real = new
ArrayList<>();

        List<String> received = new
ArrayList<>();

        real.add("то");

        real.add("ст");

        real.add("на");

        real.add("не");

        real.add("ал");

        real.add("по");

        real.add("но");

        real.add("ен");

        real.add("ос");

        real.add("ов");

        real.add("ко");

        real.add("го");

        real.add("он");

        real.add("от");

        real.add("ка");

        real.add("ра");

        real.add("ес");

        real.add("ор");

        real.add("ро");
```

```
        real.add("ли");
        real.add("ни");
        received.add("цл");
        received.add("ял");
        received.add("ае");
        received.add("ле");
        received.add("чо");
        received.add("щб");
        received.add("за");
        received.add("юе");
        received.add("юэ");
        received.add("лл");
        received.add("сф");
        received.add("жл");
        received.add("ул");
        received.add("ьй");
        received.add("вл");
        received.add("еб");
        received.add("об");
        received.add("фю");
        received.add("ьа");
        received.add("ьт");
        received.add("эб");
        Map<String, Integer>
indexReal = new LinkedHashMap<>();
        Map<String, Integer>
indexReceived= new
LinkedHashMap<>();
        for(int i =0; i<21; i++){
            String bigramReal =
real.get(i);
            String bigramReceived =
received.get(i);

indexReal.put(bigramReal,
indexOfLetter.get(bigramReal.substri
ng(0,1))*31+indexOfLetter.get(bigram
Real.substring(1,2)));

indexReceived.put(bigramReceived,
indexOfLetter.get(bigramReceived.sub
string(0,1))*31+indexOfLetter.get(bi
gramReceived.substring(1,2)));
        }

System.out.println("indexReal: " +
indexReal);

System.out.println("indexReceived: "
+ indexReceived);

        int count =0;

        List<StringBuffer> results =
new ArrayList<>();
        for (int i=0; i<21;i++){
            for (int j=i+1;
j<21;j++){

                List<Key> keys =
resolveSystem(indexReceived.get(rece
ived.get(i)),
indexReal.get(real.get(i)),
indexReceived.get(received.get(j)),
indexReal.get(real.get(j)), 31*31);

                if(keys!=null) {
                    for (int k = 0;
k < keys.size(); k++) {
                        StringBuffer
plainText = new StringBuffer();
                        Key key =
keys.get(k);
                        if (key !=
null) {
//
System.out.println("Key: " + key);
                            for (int
index = 0; index < fileData.length()
- 2; index += 2) {

String encryptBigram =
fileData.substring(index, index +
2);
```

```
key.setA((key.getA()+961)%961);

key.setB((key.getB()+961)%961);

String decryptBigram =
findBigram(key.getA(),
bigramForIndexes.get(encryptBigram),
key.getB(), 31 * 31);

plainText.append(decryptBigram);
                            }
                        }
                        count++;
                        if
(isRealText(plainText))

results.add(plainText);

System.out.println("PLAINTEXT FOR "
+ count + " CASE: \n" +
plainText.substring(0,100)+"\n");
                    }
                } else {
                    count++;

System.out.println("PLAINTEXT FOR "
+ count + " CASE: \n" + "Not
Available for this case\n");
                }
            }
        }

System.out.println("\n\nResults
amount: " + results.size());
        System.out.println("Results:
");
        for (int i=0; i<
results.size(); i++){
            System.out.println(i +
": " + results.get(i));
        }
        System.out.println();

        for (int index = 0; index <
fileData.length() - 2; index += 2) {
            String encryptBigram =
fileData.substring(index, index +
2);
            String decryptBigram =
findBigram(200,
bigramForIndexes.get(encryptBigram),
900, 31 * 31);

System.out.print(decryptBigram);
        }


    }
    public static class Key{
        private int a;
        private int b;

        public Key(){}
        public Key(int a, int b) {
            this.a = a;
            this.b = b;
        }

        public int getA() {
            return a;
        }

        public int getB() {
            return b;
        }

        public void setA(int a) {
            this.a = a;
        }
```

```
        public void setB(int b) {
            this.b = b;
        }

        @Override
        public String toString() {
            return "Key{" +
                    "a=" + a +
                    ", b=" + b +
                    '}';
        }
    }

    public static List<Key>
resolveSystem(int y1, int x1, int
y2, int x2, int m) {
        System.out.println("x*: " +
(x1-x2) + ", y*: " + (y1-y2) + ", m:
" + m);
        List<Integer> a =
resolvingEquation(x1-x2, y1-y2, m);
        if (a.size()>1){

System.out.println("Candidate on a:
" + a);

            ArrayList<Key> keys =
new ArrayList<>();

            for(int i=0; i<a.size();
i++){
                int b = (y1-
a.get(0)*x1)%m;

System.out.println("(a,b)=("+
a.get(i) + "," + b + ")");
                keys.add(new
Key(a.get(i), b));
            }

            return keys;
        }

        if (a.size()==0){

System.out.println("There are no
roots for this case");
            return null;
        }
        int b = (y1-a.get(0)*x1)%m;

System.out.println("(a,b)=("+
a.get(0) + "," + b + ")");
        ArrayList<Key> keys = new
ArrayList<>();
        keys.add(new Key(a.get(0),
b));
        return keys;
    }

    public static String
findBigram(int a, int y, int b, int
m){
        int index =
(reverseElement(a,m)*(y-b))%m;
        if (index<0)
            index = (index%m+m)%m;
        String bigram =
indexesOfBigram.get(index);
        return bigram;
    }

    public static boolean
isRealText(StringBuffer text){
        Map<Character, Integer>
frequencyMap = new
LinkedHashMap<>();
        for (int i=0;
i<text.length(); i++){
            char symbol =
text.charAt(i);
            int temp =
frequencyMap.getOrDefault(symbol,
0);
```

```
            temp++;


            frequencyMap.put(symbol,
temp);

        }

        List list = new
ArrayList(frequencyMap.entrySet());


list.sort((Comparator<Map.Entry<Char
acter, Integer>>) (a, b) ->
b.getValue().compareTo(a.getValue())
);

//        System.out.println("letter
frquency sorted by value: " );

//        System.out.println(list);



        ArrayList<Character>
characters = new ArrayList<>();

        ArrayList<Integer> integers
= new ArrayList<>();



        for
(HashMap.Entry<Character, Integer> e
: frequencyMap.entrySet()) {

            int value =
e.getValue();

            boolean isAdded = false;

            for (int i = 0; i <
integers.size(); i++) {

                if (value >
integers.get(i)) {

                    integers.add(i,
value);


characters.add(i, e.getKey());

                    isAdded = true;

                    break;

                }

            }

            if (!isAdded) {

                integers.add(value);

characters.add(e.getKey());

            }

        }




//        System.out.println("Letter
frequency: "+frequencyMap);

        int count = 0;

        for (int i =0; i<12;i++){

//
System.out.println("list.get(i): " +
list.get(i));

            String letter =
characters.get(i).toString();

//
System.out.println("letter: " +
letter);

            if (letter.equals("о")
|| letter.equals("е") ||
letter.equals("а") ||
letter.equals("т") ||
letter.equals("н") ||
letter.equals("и"))

                count++;

            if (letter.equals("щ")
|| letter.equals("ф") ||
letter.equals("ц") ||
letter.equals("ц"))

                count--;

        }

        if ( count == 6)

            return true;

        else

            return false;

        // о е а т н и

    }
}
```