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-73
Маковецький Андрій та Бадарак Оксана

Перевірили:
Чорний О.
Савчук М. М.
Завадська Л. О.

Варіант 11

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 (a, b) знайти можливі кандидати на ключ 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.

<u>Результати виконання роботи</u>:

Найчастіші біграми:

Мови:	ст	но	то	на	ен
Шифротексту:	хб	нк	бй	юж	шь

Можливі варіанти ключів:

('ух', 'ць'), ('жю', 'чж'), ('оц', 'ви'), ('эс', 'ше'), ('лк', 'лр'), ('ти', 'нх'), ('ъб', 'чч'), ('иы', 'оф'), ('шв', 'яд'), ('мч', 'аю'), ('зш', 'лл'), ('ху', 'ви'), ('рй', 'ря'), ('дя', 'тш'), ('чз', 'уд'), ('нь', 'фг'), ('во', 'хц'), ('цд', 'чр'), ('йм', 'чы'), ('се', 'вю'), ('уп', 'дк'), ('хг', 'шй'), ('тю', 'шж'), ('цх', 'яь'), ('лр', 'яб'), ('бу', 'вй'), ('яо', 'вж'), ('гж', 'иь'), ('йэ', 'юб'), ('юм', 'мк'), ('эь', 'бж'), ('бт', 'зь'), ('мв', 'ьб'), ('ас', 'йк'), ('ве', 'юй'), ('гч', 'дь'), ('ик', 'пб'), ('ыщ', 'юк'), ('юн', 'тй'), ('ыи', 'тж'), ('фч', 'ах'), ('жс', 'се'), ('жя', 'еу'), ('юь', 'вю'), ('ки', 'вх'), ('рщ', 'ющ'), ('сз', 'ти'), ('йг', 'пт'), ('шо', 'ее'), ('ож', 'пщ'), ('ан', 'фч'), ('чй', 'тв'), ('шб', 'ну'), ('нш', 'ши'), ('ят', 'йч'), ('цы', 'ьр'), ('бе', 'лю'), ('хэ', 'цт'), ('зц', 'ив'), ('ид', 'ыр'), ('зж', 'эк'), ('еь', 'дй'), ('чш', 'ыж'), ('хы', 'оь'), ('чщ', 'жб'), ('юф', 'фй'), ('рт', 'мж'), ('ох', 'яь'), ('ще', 'тб'), ('бл', 'щк'), ('сю', 'шж'), ('рб', 'ль'), ('зз', 'чб'), ('он', 'як'), ('нв', 'жй'), ('юг', 'рь'), ('йд', 'аб'), ('рк', 'зк'), ('оя', 'ой'), ('бэ', 'жж'), ('сц', 'ог'), ('хч', 'жф'), ('аь', 'мй'), ('кж', 'ск'), ('нй', 'фи'), ('дб', 'еы'), ('од', 'лр'), ('чп', 'рс'), ('йи', 'пх'), ('ья', 'ич'), ('кг', 'жю'), ('уо', 'ля'), ('яе', 'жю'), ('ры', 'аа'), ('фэ', 'чс'), ('йл', 'гз'), ('фщ', 'эс'), ('зр', 'ху'), ('лс', 'ое'), ('хф', 'ущ'), ('ик', 'гр'), ('лв', 'рд'), ('дй', 'пя'), ('до', 'рц'), ('вч', 'щю'), ('ья', 'щш'), ('ыд', 'ьр'), ('ую', 'жж'), ('эи', 'ух'), ('шз', 'бд'), ('шм', 'бы'), ('ьц', 'ги'), ('дб', 'рч'), ('жш', 'юл'), ('ае', 'юю'), ('ьс', 'юе'), ('гы', 'лф'), ('жу', 'ши'), ('яь', 'шг'), ('ыж', 'эк'), ('йь', 'дй'), ('кш', 'ыж'), ('ты', 'оь'), ('гщ', 'жб'), ('нф', 'фй'), ('от', 'мж'), ('хе', 'тб'), ('сл', 'щк'), ('аю', 'шж'), ('йб', 'ль'), ('фз', 'чб'), ('рн', 'як'), ('яв', 'жй'), ('иг', 'рь'), ('мд', 'аб'), ('ик', 'зк'), ('хя', 'ой'), ('цэ', 'жж'), ('ац', 'тг'), ('яч', 'нф'), ('щь', 'эй'), ('лж', 'дк'), ('яй', 'ри'), ('яб', 'иы'), ('щд', 'чр'), ('кп', 'яс'), ('аи', 'их'), ('ая', 'еч'), ('ьг', 'пю'), ('ло', 'ця'), ('ее', 'хю'), ('еы', 'уа'), ('дэ', 'ос'), ('рл', 'ез'), ('ущ', 'кс'), ('фр', 'зу'), ('ус', 'ге'), ('оф', 'сщ'), ('бп', 'мб'), ('ег', 'нб'), ('ию', 'рб'), ('гх', 'ыб'), ('юр', 'цк'), ('гу', 'ок'), ('зо', 'ск'), ('вж', 'эк'), ('щэ', 'йй'), ('ым', 'ай'), ('гь', 'дй'), ('ют', 'пй'), ('цв', 'гж'), ('чс', 'щж'), ('ые', 'ьж'), ('щч', 'йж'), ('ык', 'ть'), ('эщ', 'йь'), ('бн', 'кь'), ('еи', 'нь')

Критерії перевірки на змістовність тексту:

Заборонені біграми:

['аь', 'оь', 'уь', 'ыь', 'эь', 'иь', 'еь', 'яь', 'йь', 'ыы', 'ьь', 'йй']

Знайдений ключ, шо приводить до змістовного тексту:
('цх', 'яь') або (703, 956)

Шифротекст:
оквкпкящсройюфчвфбчллфэйлзщоыифуххгьижфбчбжройэжиавкхбоаэлбзьздблфюжвпыхожеуфыхьфисццоисццикш
хгтчьюбрйэунемкщхлфэсццикоэйсыфляьэсблавххуоаебвщвцззабюжэйзэсюфцхцчьдвкьбивцкьхбвхщзфийамсьэхьж
офшнйсбгежоэзхбнннкндхбххьюэкублфйлщзххгсебэуяффдзэсццоьзкжвхыфиамсьэтцщугтбйрипйпьуфптшьюуйукь
лкуеафтбгфмсмешзчеюцнэпфиздббюакличуьлчяюеьххущьафюешзхксищфнлазьзьфююшлйапзгзхбфйсбцрптозкендз
энлбкнкщуюжбйдрптцьдьжьяжчэщлакэзйфбюыхожобьэгройюфеыхцылеимкжфлйутдяйиакютэффцйабгнйакбхеык
жцлнььфьюдфбгьишллаяфеяфзьзжцазпзфижжнлмккцниулбргзхбтшяюйцуфьюпзмкабвхщугтбйлзтцсуязяфыюедяь
амсчежоакютэффцшжчядбоапздлчьпдкьуавкййлцыхяфнлвцаинарйлзщомксйуккзхьлзтцсуямсчеылууезкзозуффыоа
йнтббйцилбвзщьцупзоулцафэйакюсзкшавкафэйоендхкыхзннложшушьмйыжбйоеахуиюиазезждрлрйозвфьюяжсблзт
цсубйчэбйфувккьбиыжмхбыотмхфыхцылкцбьмуксяфэлойэлтапзсфбззфйузядзозцьндкьдядзюцнэпфизчбющнкщугт
бйяфюбфбшуэлтбыхожлфюжйсизсбрйэуэлсымсчеббвкслаклийияффшлаэлкцзылсйуяфжнбкдбьэзээипуэлойчьшзкфя
фицнкяилклзтцсуожоымехцэжшубкмьцфэйхьазцодбжройэщщройвмехщльзйжчячбэлрйычицхьхихкрырйюжкжтуэл
тбицдзщубгыфхцщбщьзцрипймкжфлйикхкзцьхэлнктцчфлцждефэйоемазялвбйццакжтшьдфшулэсбоеббвквещьэкеь
юфукззщонихбмсйснгчьфшйсжртджапбляэфббозэиылшжохюжебиисйпчфзикозывмсойыцждяуяьрипйшкпзхбвхшл
югеыбйщуэлойактфнльжвхшьткхбшушьщыэфйуяфочшьцьдьсиртмкауэлойуиазезхкрилснгьлюджтнйыфьюшкябпф
щжэлохщлыфизббвкейукыцйяпфлхмсоймкяждзацаьщьюцсйюжбкьэксюечбшуйудзфисццомкгзбкжинкчавктккхуилк
хкнкуийиаьжнсдмуфпсамсьхоыозбицьохзнтбшьвфьлшзчебкфичбяфламсьхоыыжбйяфзннкозьхэлойэлвжшьсьхкчьчь
скшьюкщэшьнджодбчясгэфнлсьхбсетцгьпйуклкцзнкшьюкщэшьуяфшебблшжчьтяпфзийияфэсрйсфебгрчлчяфыяфэл
тбвжчьтяпфукжфнезфжпфжвхюкщугтбйяфшлкцокшьюкщэшьнджоахакдйукмкхбожойшьазилеьпрпффцехпфцзткму
щьщуыжхеылхккььомажрпзэжюкейэфзкэтеымехцэжнкчуюжьюсйюжххшбжсюжрйгфнлмкактфожщлнйожткхбеххзь
зкзткшлдяазмзбишлицхьсбоемкфждзацшлзилзыхгтуябкоквкпкйцпаюгбкйишьщзпзхббйххптебькткширтйутдньвйгт
яавкниеььяозшльщнкфшвхгтмкбькцрлеьгзттлвхкффэйсйюжххцтйуйувздбфыпзвизсрйцзычнйьюсьыэхьжомкчьюдьх
ксуяфзиоуфукфжсбщулэтбрйэумсьийиозэррйегдяозфиззикозшьюкюдчбьэыфйсюжрйгфпфщлмкактфпфьжщлнйьиоз
щжэлнкаыйсюжрйгфарчжжппуехюжиьщжэлнкаыдяньлзычжспфщлсьхбсдщлеьхиуиакнктдицззкабйыжлзулфжэлчь
цьньйишьхцпфнлззлбфыйннказезньпьтжксягэуйушжщлнйэлшзвлэуфыхьщупзсцдбсхйуюжэфэйукьфеяебнйгфоафы
мелзюфлэыфцийидксыоавкчлфихкроюдьхожбйэфчязэнисдеьактфххмсфиакюспфзнсыяфщлыдсйпблкцзэроблячжях
азслеижщнкикцьаклсюжпфбкикохбыыццзчкшлдящуехнеткшлыфчэбйвпроцьохзнбйбйэфюенкозьхшвьэеуьзкзчксйю
жххдяэфсыюжехгжцллфсепдщйукюдазцоозпащьгикьюгтайндяйлвхшчицьзуехбхнтбознкооакжфдябкэфйнщпозеьлк
цзиоюдпавклижочбфбйудзлхйуюбющнкзффзхфптэсожсюткьхэлюжщлжоцзеьуинйссыжезеяшьюкзфебэуйуйлыхдзс
ягхчьюбшуйуезхьчьзщнкцьтцгьсбоееьакюидьхьсбхбщьщзчиыжчидунйдрттэсьвозулйафбзяхннкдяьахипазнойоаяфп
бэуйулфсыюжикыйдяйлфцждщьэкеьщбюгтаююпаксьгиищзпзхбпфбллкыхсбгфчяыфпбйуеятыньцьяисцыбюжщзйжь
ющьжжксягфйэфяьпмгьижмхбыцьшзыйяафбщфэйпфэлзяткфсзвткэрйсяьшлсбчжэфюжебяфххкьхцньежвхшцтцокф
уфклбшсяаюжбйойодслсьхбизкннлжосыехщлхфюжэлщйцухцньяфьиьтощнкиккннлжовяцлнкэкрлчкхислшжщзйжь
ющььтярйикхьыжщзйжьющььтэскьшчебфйсбшьзээфсцтцвцикоошжшзхбьвгзфьакюсэсбящьюлкеыллйукыцйяпфы
цззьлгзхавкхбкйцхьвфбмеэыывкьпущннйюжхунлисьвбказфидялзщоцьжфдящжтттцызеьуишзоочьщбфцщуеясббйн
ьлзпзкьлкхкнкюыткфсгхюювгэфзкфжвхпумаехбюгзоыпмгьшьвккьюдкьхкфбсьщфвзфкзлуиеьылйалцйиыьтжезюсцр
щлмкактфьюшлсыхьшьззфцяфотмхеыыжбйфкэкщьникьылнтнлждзьрылзнкисбйхужнююнкиячвьхьлкцгикьылууць
фькьбссбткгдьхьлгыыхыжбйоечбюжмтчжфзвцлзьзяцждяукжжнвхчяфкшамспьэикьылуудцуфбюакгапфиишьазщьцс
ицакикфисццортчкезлзрлкцзьчуйухбххцуыжпазяэфжпеавкййдзтбрйфцщбьэртфимахичьясзяоттдазазезооскэимйлйо
енквффыыфвхбйакнктдсжкстбчбфбдясдьжщьцзщувзлиждщьэывкейыфдичбдзикхбщььзыискчьцьньроьхожойгфбю
мкюквкхзюклккябехцзхбткькьнашупфьжоксихкгежошлткиыххлэюснйчэбйеявхфжюзэфхбццшзябдиозмснгьжгзйфд
злзпзткнкюсфзиклирптзнйпчюжбклощхщнозхкмаобюерортнджоикпзтоозшсщнфкзльэщлнкззфидящьвггдхикьйиакх
кбиинклбияждзьясгэфцлхкьфющнкндйиэигдкевкхбсдлавкбиьфцуэфйуяфочакябсхбясгэфшфщнбйыжяхьлюжчжебб
леиззпзфсщлпбрсозюибкззасвкбьфбйлеьхкхбиипяьэлзйфьюмккцшзабгдкклыряймсебйубюгдчкшгяюмайнвхозэгяюнк
вфюжозикшсххойчярйфилйукыцйяпфпафбтуюеромклхнйисзялзбиюлехойьлоифыкцгиьзообкохзнбйксццрфсццаьб
кьхыжксукичебфбсьмцлкчуйичьпршснкьхжройсхйнсдэннлыжакуфебкжбгньпьгьиявхжогьгишькннлжопыфбюжяав
кэхманйгзоьпмгьйукыцйяпфмкгцлзюфяапбшьшчебзмлфнкшлчьцьньсыьэыфзкхцщзежыфйснйэфвхгтиккьхикьылэл
ткеякссбляфкфипьяиозяжнеияфбхиьзяфксвхббоесыьэыфебьэщлчяпфщлткчллкыхтбукшьмкнкыльзньшьлзоежонцяб
южьзщзьоююмкшьюкуеулщфюжюзжйебсехьакозвфзясгэфеыфбюжяьхббйеящлсьхбйиазхункщулэсбгзжрвкроюдгьх
бвхтбвщбисьидьфшвозмйптсбфимсшьфцфцсущннйюжхулфларйьэыфэлюжебблшжеьуийснйеыэлмктцэкэцдбяфожв
хбкозмкбхйстбожфбгзоьпмгьхцмсэфбхйсэфцуйубюмкооподщлйиеьпршссбьхмсрйбьжооефшчвфбгфйснймяюзикмко
квкфитжсыэлжрхбазйфбюгьгиозбкдбшвсйюжгзоьпмгьхцщзозсйьгбйяфлиуцейыфвзсллкыхнйчьифпфсыюжбйтбоемк
нкхкзжфбеькгвцчьмктцгьхкхцщзгзбьшьмккцбквцьзикжрткхкщйебляьияфйлждзьубрйэушуйуюбфбфбхиясиавкохзнб
йчбфбяфнлззюясгэфцлсьхбсешьмхбыдзббвкйскьщьщутдрыньцьяфкжкснйкбпчяфмавкбклийирощжтттцызнквфехгт
бклжсбсдхучяюгуилхбгыфсдьжозмкхбшэеяйньлюдшххнююойсыяфэуттйеифкжоайлчьйирошьцзхбщфвпохьгбыххл
энлжрйуыжезткпяьэсджаехмсбиокщоакьхшущьодкьйироткежехмаюжртыикьнтеымеодкьйирыххлэззфжвхзябйнняг
хуежйсьжйнуиэфщлакооакацынэфмкшзчйойэфэсехьэнзмкнкуфвхюгтбнйгфоамсбгйимкцйшэзиэлеьбкжийиромкежя

гчеяигзоьпмгьшахисйнгбйоемаяфлйчбюжгззбихжезруозацткежехикхкиясгэфцлсьхбсееьакдяьэбкндхкндйлждзьжог
зхфеяжфдяткшйхузвфичьактфдршсвжнйцупфгидбжркйехгтбкикхкчкчэзэбьщоьжцлшудзэфртаксбехюжбксбоапзтдж
ихбдзчьцьхкроеьтжщпулахдзлкцхзясдптрйфбблюдлаззждазйинибкрлсьсуцихбшедьщбщугтйнсдгрфпхьпьгьазйичьл
гыцюкмйсбткпзйфзясхждззулгрламсбкпащлыффцйидклфбзнквфехттюжюекцшькарйбккцфиьжфбткнкшьозжрпбшь
ууйугзнкмкхаждазпуежвхвяьлмкактффымеозфжмстккфыцждфьэшшвоечкфьифцрнйжртдидяябйннягхуьлйсфвхбхх
цуюжмкхббюьзчьцхзяакщугтбйгзнеазцоезикжфязтзозфуфпсьжясгэфйлыцылыцнамсчеуырйюжгкябьэщлнклйьфртк
жмсткпзпьэйюдбзфуфпниубфэщлчямкиннлпфгтэсазгзжяозщьякцхсбизльдсюечкшлдяцьндмефшсбрйакуфлавкафэйо
ефкбкгежофбуихцылщфяамснгфцйишььхэфщлньдяятожыхйуфкхбпфюжойнлпцшушьвйзсшжфизфеыххлэмксйнгсю
гзяфсывкнуэлчьцньяффзсйыщеьгьгишьыжьхожтцеьчьууйуюдикткхииолкцзэробькшьуябзыхьэебрссбюючяьэсбяфя
апзбхебтббleьхичьафюжождфбгцифшдяьэбкщщнкьясгэфщлюжезшзщлсьхбизсльхпбебблскулшзабьльхпбебблшжф
бюдщьэкеьюгблгрщфчэбйбзиоожакычхуяфдкьфэсехазиооемазяпзиочбненкюсцршсшснйебвжгзуцщщнкэррйегбют
шцуьleьхцкннлзйыжбювзтзщулэеяозгзйннклфнльзщзьоцьэизфблчязффзиочьфцшьяжйссбоеюктжездоббщьйамсфиг
ьсссбююактяфпшвцьрбщьакнумкхкжиозлзйенкиккьафцуфылццзозбискхбжпромсшьчкхбгхакккжфнликнкфубюэфэ
луинбоаебшбьэыфэскифыехмсмкхкжоыжньфуйлкзаьжоьлвхгрвкмкнихбтбсещьжжчэбйвпропбрйэуяфяазяэфжпьзкз
чьзжйссбоешьесзягзшуйухэыжббйуткнкфжсбляозчьзжездблцшзхбтбоербткмясгэфгкшьйлыхдзшьчкшлдяэфщнукзз
щобьоозфщнтбуфыфхуфыхьсцлзгслэлаэллкыхсбэуйуяфыцхбвкыфбйэфяасчнйюжбйфзикушхуцигдййьибузяшьюдзк
ьйчэбймахиапвикьылшжсбвхафртшсяфэлойцпхьцьдьяясгэфдибьпзсдрусгеьбкеннкьхмсжпулскезщодфыщйиыьжахх
шэйуфбехфыцкйфцуфыхьозжилзжячэтццьтквкцзйфбяьлжрйучяиигьоонисучяэфьжксеяквавкщурйцтхцньаксуьицьа
цыйтубйыщежвхююаьщьбььиьтеыххлэнлждзыылыгехьвафчэбккзхкваыфбявхбкйсфвпфизьхдрнешсбйхушвхьгзуиюр
вхсыюжцьрбщьыдьлэдеьдягьгихцтцгьпйукыщйяпфшльжшзхбукшавкафэйоемкфсбкбктооннкнксыехгтсцсихкхьгзак
исццикмузяшььзюбфбоаньчьылыщэщнкюдэзффэйгзоьпмгьижкйптюертохюжебиимкэзчуидхцгьоониюыехэуехжсбй
ббифкжгирокамснгфцозхфсбшлиймачжбйяфшлхуяфззбкакуныфэлыцубгрлахиафойэлнннкфисццохбщьчфдзщьюлч
ьэиуфщлшфшубкгьоониюбрйэуттбйыжбйоесйптчяиивцулксжцозикшьохюжебойукыщйяпфпьуфптшьйайндякжоым
ехцэжшубкежвхтыххлэцркйгхнлдьцфббшфнльзвояфмкоцшлфжюжбйчьзжрйэлецозикшлеьйиодэкнуфыхьпьэйбиок
щоакшлиймачжюешьакозатцркйоекзпашубюсдслбклкьзклбцхбойоемкгзхбфисзмкюкэзчуаьщьюцртебьиисфвпфвзэл
нквксфюжобщфэйццфцяь

Розшифрований текст:

хорошосэрбиллнехотясунулденьгивкарманвотчтобиллвыпростопосеетеэтуновуютравукогданибудьвдругойразкакто
лькояпомрунадругойжеденьможетеперекопатьэтучертовулужайкунукакхватитувастерпенияподождатьещелетпятьш
естьчтобыстарыйболтунуспелотдатьконцыужбудьтеуверены подождусказалбиллсамнезнаюкаквамобяснитьнодляме
няжужжаньеэтойкосилкисамаяпрекраснаямелодиянасветевнейвсяпрелестьлетабезнеебыужаснотосковалибеззапаха
свежескошеннойтравытожебиллнагнулсяиподнялземликорзинкуяпошелковрагувыславныйюношаивсепонимаетея
уверенизвасполучитсяблестящийиумныйрепортерсказалдедушкапомогаяемуподнятькорзинкуявамэтопредсказыва
юпрошлоутронаступилполденьпослеобедадедушкаподнялсяксебенемногопочиталуиттиераикрепкоуснулкогдаонпр
оснулсябылотричасавокнавливалсяяркийивеселыйсолнечныйсветдедушкалежалвкроватиивдругвздрогнулслужайки
доносилосьпрежнеезнакомоенезабываемоежужжаньечтоэтосказалонктотокоситтравуноведьеетолькосегодняутромс
косилионещепослушалдаконечноэтожужжиткосилкамернонеутомимодедушкавыглянулвокноиахнулдаведьэтобилл
эйбиллфорестервамчтосолнцеудариловголовувыкоситеужескошеннуютравубиллподнялголовупростодушноулыбну
лсяипомахалрукойзнаюнокажетсяутромяработалнеоченьчистодедушкаещедобрыхпятьминутнежилсявкроватиисли
цаегонесходилаулыбкаабиллфорестервсешагалскосилкойнасевернавостокнаюгинаконецназападиизподкосилкивесе
лобилдушистыйзеленыйфонтанввоскресеньеутромлеоауфманбродилпосвоемугаражусловноожидаячтокакоенибудь
поленовитокпроволокимолотокилигаечныйключподпрыгнетизакричитначнисменяноничтонеподпрыгивалоничтоне
просилосьвначалокакаяонадолжнабытьэтамашинасчастьядумаллеоможетонадолжнаумещатьсявкарманеилионадол
жнатебясамогоноситьвкарманеодноязнаютвердосказалонвслухонадолжнабытьяркойлеопоставилнаверстакбанкуора
нжевойкраскивзялсловарьипобрелвдомлинаонзаглянулвтолковыйсловарьтыдовольнаспокойнавеселаввосторгетебе
вовсемвезетивсеудаетсяпотвоемувсеидетразумнохорошоиуспешнолинапересталарезатьовощиизакрылаглазапрочит
аймневсеэтоещеразпожалуйсталеозахлопнулсловарьзакакиеэтогрехиядолженцелыйчасждатьпокатыпридумаешьмн
еответскажитолькодаилинетбольшемненичегоненадотычтоженедовольнанеспокойнаневеселаиневосторгедовольн
ыбываюткоровыаввосторгемладенцыданесчастныестарикикоторыеужевпаливдетствосказалалинануанасчеттогочто
веселасамвидишькакявеселосмеюськогдаскребуэтураковинулеовнимательнопогляделнаженуилицоегопрояснилосьт
ыправалинамужчинытакойнародникогданичегонесмыслятможетбытьмывырвемсяизэтогозаколдованногокругаужес
овсемскороявовсенежалуюсьзакричалалинаятонеприхожуктебесословареминеговорювысуньязыклеотыведьнеспра
шиваешьпочемусердцеутебястучитнетолькоднемноиночьюнетаможешьтыспроситьчтотакоебракттоэтознаетнезадав
айвопросовестьжетакиелюдивсеимнадознатькакустроенмиркактокакседакакэтозадумаетсятакойипадаетстрапециив
циркелибозадохнетсяпотомучтоемуприспичилопонятькакунеговгорлемускулыработаютешьпейспидышииперестань
смотретьнаменятакимиглазамибудтовпервыйразвидишьлинааауфманвдругзамерлапотянуланосомвоздухвотбедаавсе
тывиноватонарвануладверцудуховкиоттудаповалилдымсчастьесчастьегорестновоскликнулаонаиззаэтогосчастьямы
стобойссоримсявпервыйраззаполгодаивпервыйраззадвадцатьлетнаужинбудутугольявместохлебакогдадымрассеялс
ялеоауфманаужеиследпростылгрохотлязгсхваткачеловекасвдохновениемденьзаднемввоздухетакимелькаюткускиме
талладеревамолотокгвоздирейсшинаотверткипоройлеоауфманаохватывалоотчаяниеионскиталсяпоулицамвсегдабес
покойныйвсегданачекуонвздрагивалиоборачивалсязаслышавгдетовдалекечейтосмехприслушивалсякзабавамдетвор

ыприсматривалсячтовызываетудетейулыбкувечерамионподсаживалсякшумнойкомпаниинаверандеукогонибудьизс
оседейслушалкакстарикивспоминаютпрошлоеитолкуютожизниприкаждомвзрывевеселья оживлялсяточногенералк
оторыйвидитчтотемныевражескиесилыразгромленыичтоегостратегияоказаласьправильнойподорогедомойонторжес
твовалпоканевходилопятьвсвойгаражгдележалимертвыеинструментыинеодушевленноедеревотогдаегосияющеелиц
овновьмрачнелоипытаясьизбытьгоречьнеудачионсожесточениемрасшвыриваликолотилчастисвоеймашиныслovноэ
тобылиживыеяростныепротивникинаконецконтурымашиныначаливырисовыватьсяичерездесятьднейиночейдрожао
тусталостиизможденныйполумертвыйотголодатакойвысохшийипочерневшийточновнегоудариламолниялеоауфман
спотыкаясьпобрелвдомдетиссорилисьиоглушительнокричалидругнадруганопривидеотцатотчасумолкликакбудтопр
обилурочныйчасивкомнатувошласамасмертьмашинасчастьяготовапрохрипеллеоауфманлеоауфманпохуделнапятна
дцатьфунтовсказалаегоженаонужедвенеделинеразговаривалсосвоимидетьмионисаминесвоясмотритеонидерутсяего
женатожесаманесвоясмотритеонапотолстеланадесятьфунтовтеперьейпонадобятсяновыеплатьядаконечномашинагот
оваасталимысчастливеектоскажетлеоброситьмастеритьэтичасывнихневлезетниоднакукушкачеловекунеположеносо
ватьсявтакиеделагосподубогуэтонавернонеповредитавотлеоауфмануодинврединикакойпользыеслитакбудетпродол
жатьсяещехотьнеделюмыегопохоронимвегособственноймашиненоэтихсловлеоауфмануженеслышалонсизумлением
смотрелкакнанеговалитсяпотолоквоттакштукаподумалонужележанаполунотутегообволоклатьмаионуслышалтолько
какктототриждыпрокричалчтотонасчетмашинысчастьянадругоеутроедвараскрывглазаонувиделптицонипроносилис
ьввоздухеточноразноцветныекамешкиброшенныевнепостижимочистыйручейилегонькозвякнувопускалисьнажестян
уюкрышугаражасобакивсевозможныхпородтихонькопрокрадывалисьводвориповизгиваязаглядываливгаражчетверо
мальчишекдведевочкиинесколькомужчинпомедлилинадорожкепотомнерешительноподошлипоближеиостановилис
ьподвишнямилеоауфманприслушалсяипонялчтовлечетихвсехкнемуводворголосмашинысчастьятакоеможнобылобы
услышатьлетнимднемвозлекухникакойнибудьвеликаншиэтобылоразноголосоежужжаньевысокоеинизкоеторовноет
опрерывистоеказалосьтамвьютсяроемогромныезолотистыепчелывеличинойсчашкуистряпаютсказочныеблюдасама
великаншаудовлетворенномурлычетсебеподноспесенкулицоунееточнорозоваялунавполнолуниевотвотонанеобятна
якаклетоподплыветкдверямиспокойноглянетводворнаулыбающихсясобакнабелобрысыхмальчишекиседыхстариков
постойтекагромкосказаллеояведьсегодняещеневключалмашинусаулсаулподнялголовуонтожестоялвнизуводворесау
лтыеевключилтыжесамполчасаназадвелелмнеразогретьееахдаясовсемзабыляещетолкомнепроснулсяионопятьоткин
улсянаподушкулинапринеслаемузавтракиостановиласьуокнаглядявнизнагаражпослушайлеонегромкосказалаонаесл
иэтамашинаивправдутакаякактыговоришьможетбытьонаумеетрожатьдетейаможетонапревратитьстарикасновавюно
шуиещеможновэтоймашинесовсемеесчастьемспрятатьсяотсмертиспрятатьсяваттыработаешьсебянежалеешьавконц
еконцовнадорвешьсяипомрешьчтоятогдабудуделатьвлезувэтотбольшойящикистанусчастливойиещескажимнелеочт
оунастеперьзажизньсамзнаешькакунасведетсядомвсемьутраяподнимаюдетейкормлюихзавтракомкполовинедевятог
овасникогоуженетияостаюсьоднасостиркойоднасготовкойиноскиштопатьтоженадоиогородполотьивлавкусбегатьис
ереброчистить яразвежалуюсьятольконапоминаютебекакведетсянашдомлеокакяживутаквтотответьмнекаквсеэтоу
местится явтвоюмашинуонаустроенасовсеминачеоченьжальзначитмненекогдабудетдажепосмотретькаконаустроенал
инапоцеловалаеговщекуивышлаизкомнатыаонлежалипринюхивалсяветернизудоносилсюдазапахмашиныижарены
хкаштановчтопродаютсяосеньюнаулицахпарижакоторогоонникогданевиделмеждузавороженнымисобакамиимальч
ишкаминевидимкойпроскользнулакошкаизамурлыкалаудверейгаражаиззагаражаслышалсяшорохснежнобелойпен
ымерноедыханьеприбояудалекихдалекихбереговзавтрамыиспытаеммашинудумаллеоауфманвсевместеонпроснулся
позднооночьючтотоегоразбудилодалековдругойкомнатектотоплакалсаулэтотышепнуллеоауфманвылезаяизкроватии
пошелксынумальчикгорькорыдалуткнувшисьвподушкунетнетвсхлипывалонвсеконченоконченосаултебеприснилос
ьчтонибудьстрашноерасскажимнесынокномальчиктолькозаливалсяслезамиитутсидяунегонакроватилеоауфмансамн
езнаяпочемувыглянулвокнодверигаражабылираспахнутынастежьонпочувствовалкаволосыунеговсталидыбомкогда
саултихоньковсхлипываянаконецзабылсябеспокойнымсномотецспустилсяполестницеподошелкгаражуизатаивдыха
ниеосторожновытянулрукуаа

Код програми:

```
# Variant 11
ALPHABET = 'абвгдежзийклмнопрстуфхцчшщьыэюя'
ALPHABET_DICT = {
      'а': 0, 'б': 1, 'в': 2, 'г': 3, 'д': 4, 'е': 5, 'ж': 6,
      'з': 7, 'и': 8, 'й': 9, 'к': 10, 'л': 11, 'м': 12,
      'н': 13, 'о': 14, 'п': 15, 'р': 16, 'с': 17, 'т': 18,
      'у': 19, 'ф': 20, 'х': 21, 'ц': 22, 'ч': 23, 'ш': 24,
      'щ': 25, 'ы': 27, 'ь': 26, 'э': 28, 'ю': 29, 'я': 30
}
THEORETICAL_MOST_FREQUENT_BIGRAMS = ['ст', 'но', 'то', 'на', 'ен']
def import_data(filename):
      with open(filename, 'r', encoding='utf-8') as f:
            return f.read()

# Only works with bigrams!!!
```

```
def bigram_to_int(bigram):
      global ALPHABET_DICT
      splitted = list(bigram)
      bigram_num = ALPHABET_DICT[splitted[0]] * 31 + ALPHABET_DICT[splitted[1]]
      return(bigram_num)

def int_to_bigram(number):
      global ALPHABET_DICT
      rev = {val: let for let, val in ALPHABET_DICT.items()}
      bigram_first = number // len(ALPHABET_DICT)
      bigram_second = number % len(ALPHABET_DICT)
      return rev[bigram_first] + rev[bigram_second]

def letter_to_int(let):
      global ALPHABET_DICT
      return ALPHABET_DICT[let]

def int_to_letter(num):
      global ALPHABET_DICT
      rev = {value: key for key, value in ALPHABET_DICT.items()}
      return rev[num]

def gcd(a, b):
      if b == 0:
            return a
      else:
            return gcd(b, a % b)

def modular_inverse(a, b):
      x, y = 0, 1
      u, v = 1, 0
      m = b
      a1 = a
      b1 = b
      while a != 0:
            q = b // a
            r = b % a
            m = x - u * q
            n = y - v * q
            b,a, x,y, u,v = a,r, u,v, m,n
      gcd = b
      if x < 0:
            x += m
      if gcd == 1:
            return x
      else:
            #raise ValueError('Modular inverse for such values does not exist:',
a1, b1)
            #print('Modular inverse for such values does not exist:', a1, 'mod',
b1)
            return False

# Solve linear equasion:
# ax = b mod n
def solve_linear_equasion(a, b, n):
      #print(a, b, 'koef')
      d = gcd(a, n)
      if d == 1:
            mi = modular_inverse(a, n)
            if mi == False:
                  #print('The equasion has no solutions (no modular inverse)')
                  return False
            x = (b * mi) % n
```

```
            return [x]
      elif d > 1:
            if b % d != 0:
                  #raise ValueError('The equasion has no solutions. ({} x = {} mod
{})'.format(a, b, n))
                  #print('The equasion has no solutions. ({} x = {} mod
{})'.format(a, b, n))
                  return False
            if b % d == 0:
                  res = []
                  a1 = a // d; b1 = b // d; n1 = n // d
                  mi = modular_inverse(a1, n1)
                  if mi == False:
                        #print('The equasion has no solutions (no modular
inverse)')
                        return False

                  x0 = (b1 * mi) % n1

                  for i in range(d):
                        res.append(x0 + i * n1)
            return res

def check_text_reality(text): # TODO: finish and improve
      # Filtering using forbidden bigrams
      forbidden_bigrams = [
            'аь', 'оь', 'уь', 'ыь', 'эь', 'иь', 'еь', 'яь', 'йь', 'ыы', 'ьь',
'йй']
      for b in forbidden_bigrams:
            if b in text:
                  return 'Forbidden bigram found: ' + str(b)

      # Filtering using monogram frequencies
      pass

      # Filtering using bigram frequencies
      Pass

      return 1

def get_all_bigrams_pairs(arr):
      res = []
      # Find all possible practical pairs
      for i in range(len(arr)):
            # fix one of the items and add all others
            for j in range(len(arr)):
                  # Do not pair a bigram with itself
                  if i != j:
                        res.append((arr[i], arr[j]))
      return res

def find_key(theor_bigram_pair, encr_bigram_pair):
      # Y1 - Y2 = a(X1 - X2) (mod m**2)
      # y = ax mod n
      # ax = y mod n
      global ALPHABET_DICT
      m = len(ALPHABET_DICT)
      x1 = bigram_to_int(theor_bigram_pair[0])
      x2 = bigram_to_int(theor_bigram_pair[1])
      y1 = bigram_to_int(encr_bigram_pair[0])
      y2 = bigram_to_int(encr_bigram_pair[1])
      key_pairs = []
      # ax = y mod n
```

```
	#print(x1, x2, 'x1x2')
	#print(y1, y2, 'y1y2')
	temp = solve_linear_equasion((x1 - x2), (y1 - y2), m ** 2)
	if temp == False:
		#print('Key not found: LE not solved')
		return False
	key_arr = []
	for a in temp:
		b = ((y1 - a * x1) % m**2)
		key_arr.append((a, b))

	big_key = []

	for key in key_arr:
		a = key[0]; b = key[1]
		a_big = int_to_bigram(a)
		b_big = int_to_bigram(b)
		big_key.append((a_big, b_big))
	return big_key

def decipher_affine_bigram(text, key):
	m = 961
	a = bigram_to_int(key[0])
	b = bigram_to_int(key[1])
	inv_a = modular_inverse(a, m)
	if inv_a == False:
		#print("Inverted value does not exist!")
		return False
	res = ''
	for i in range(0, len(text)-1, 2):
		y = bigram_to_int(text[i:i+2])
		x = ((y - b) * inv_a) % m
		res += int_to_bigram(x)
	return res

def attack_affine(theoretical, practical, ciphertext, logfile):
	all_lang = get_all_bigrams_pairs(theoretical)
	all_encr = get_all_bigrams_pairs(practical)
	logfile.write('Pairs of most frequent bigrams (theoretical):\n')
	logfile.write(str(all_lang))
	logfile.write('\n\nPairs of most frequent bigrams (practical):\n')
	logfile.write(str(all_encr))
	matched_texts = {}
	keys = []
	# Match all bigrams in language to the ones in ciphertext
	# and find the keys for them
	for i in range(len(all_lang)):
		for j in range(len(all_encr)):
			#continue
			key = find_key(all_lang[i], all_encr[j])
			if key == False:
				continue
			for k in key:
				keys.append(k)
	# Remove duplicate keys
	keys = list(dict.fromkeys(keys))
	logfile.write('\n\nAll possible keys:\n')
	logfile.write(str(keys))
	logfile.write('\n\nBad keys and reasons why:\n')
	# Decipher text for each key and check if it is ok
	# TODO: fix same keys bug
	for key in keys:
		deciphered_text = decipher_affine_bigram(ciphertext, key)
```

```
            if deciphered_text == False:
                  continue
            #print(deciphered_text)
            is_real = check_text_reality(deciphered_text)
            if is_real == 1:
                  print('Key:', key, bigram_to_int(key[0]), bigram_to_int(key[1]))
                  print(deciphered_text)
                  matched_texts[key] = deciphered_text
            else:
                  print('Key:', key, bigram_to_int(key[0]), bigram_to_int(key[1]))
                  print('The text is not real: ' + is_real + '\n')
                  logfile.write('\nKey: ' + str(key) + ' ' +
str(bigram_to_int(key[0])) + ' ' + str(bigram_to_int(key[1])) + '\n')
                  logfile.write('The text is not real:\n' + is_real + '\n')
      logfile.write('\n\nAll texts that matched the text reality check and their
texts:')
      for key in matched_texts:
            logfile.write('\n\nKey: ' + str(key) + ' or (' +
str(bigram_to_int(key[0])) + ', ' + str(bigram_to_int(key[1])) + ')\n\n')
            logfile.write(matched_texts[key])

def find_most_frequent_bigrams(text, quan):
      sum = {}
      for i in range(len(text) - 1):
            bigram = text[i:i+2]
            try:
                  sum[bigram] += 1
            except:
                  sum[bigram] = 1
      sorted_sum = sorted(sum.items(), key=lambda kv: kv[1], reverse=True)
      #print(sorted_sum)
      return [big for big, quan in sorted_sum[:quan]]

def main():
      global THEORETICAL_MOST_FREQUENT_BIGRAMS
      global PRACTICAL_MOST_FREQUENT_BIGRAMS
      logfile = open('results.txt', 'w', encoding='utf-8')
      ciphertext = import_data('11.txt')
      ciphertext = ''.join(ciphertext.split())
      pr_most_frequent = find_most_frequent_bigrams(ciphertext, 5)
      print(pr_most_frequent)
      attack_affine(THEORETICAL_MOST_FREQUENT_BIGRAMS, pr_most_frequent,
ciphertext, logfile)

main()
```

Висновок:

Виконавши роботу, ми засвоїли методи частотного криптоаналізу на прикладі розшифрування афінного шифру біграмної заміни та опанували прийоми роботи в модулярній арифметиці.