



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

ЛАБОРАТОРНА РОБОТА №2
з дисципліни
«Криптографія»
на тему: «Криптоаналіз шифру Віженера»

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:

Засвоєння понять ентропії на символ джерела та його надлишковості, вивчення та порівняння різних моделей джерела відкритого тексту для наближеного визначення ентропії, набуття практичних навичок щодо оцінки ентропії на символ джерела.

Порядок виконання роботи:

0. Уважно прочитати методичні вказівки до виконання комп'ютерного практикуму.
1. Самостійно підібрати текст для шифрування (2-3 кб) та ключі довжини $r = 2, 3, 4, 5$, а також довжини 10-20 знаків. Зашифрувати обраний відкритий текст шифром Віженера з цими ключами.
2. Підрахувати індекси відповідності для відкритого тексту та всіх одержаних шифртекстів і порівняти їх значення.
3. Використовуючи наведені теоретичні відомості, розшифрувати наданий шифртекст (згідно свого номеру варіанта).

Ключі для зашифрування:

- 2: 'ор',
- 3: 'рик',
- 4: 'кусь',
- 5: 'морти',
- 9: 'автопилот',
- 12: 'велоцераптор',
- 15: 'астроориентация'

Індекси відповідності зашифрованого тексту:

Довжина ключа	Індекс відповідності
0	0.053695
2	0.044587
3	0.039373
4	0.0358
5	0.037049
9	0.034776
12	0.034023
15	0.03495
20	0.032833



Розшифрування тексту (варіант 11):

Індекси відповідності для довжин ключа 2-30

2	0.035446
3	0.035486
4	0.035423
5	0.035516
6	0.035521
7	0.035473
8	0.03546
9	0.035554
10	0.035477
11	0.035286
12	0.035625
13	0.035453
14	0.035314
15	0.035454
16	0.03557
17	0.058332
18	0.035532
19	0.035478
20	0.0352
21	0.035777
22	0.035293
23	0.035422
24	0.035629
25	0.03532
26	0.03525
27	0.035399
28	0.035334
29	0.035563
30	0.035619



При $r = 17$ індекс відповідності шифротексту значно більший за інші, отже довжина ключа — 17 символів.

Після знаходження довжини ключа виконуємо розшифрування шифру Цезаря для кожного блоку за допомогою частотного криптоаналізу. При порівнянні найчастіших літер у блоках та найвірогідніших літер мови, отримуємо такий ключ: венецианскийкупец

Ключ скоріше за все складається з двох слів, перше з яких — *венецианский*, тому треба перебрати друге слово. Повторивши розшифрування Цезаря для 14го блоку з двома наступними по частоті теоретичними значеннями, отримуємо слово *купец*.

Отже, вірогідніше за все, ключ: венецианскийкупец.

Розшифрувавши шифротекст цим ключем, отримано змісовний текст, отже ключ підтвердився.

КЛЮЧ: венецианскийкупец

Шифротекст:

втяугроъцсхйиббъеумчтптикуочьякуфупчхлоюгжйцтарсьшяуьнныфонингвциюфыовильсвнфтюльдг
ашьицсывьилхтфчнфуэуьрттцяцыпюраэпеябчнсюэещфпаъехехацидмырмрцшсжчдующцсттйирчуббв
пкяхймнвыкуйъыушэяьдфмтипъоыпюудмкнтйлдтукасмшъннвзикзыдныкткшцпчыкпкбдмычткчоыьб
еэъехчрызпщъттыужупндзчртшънцжшыщврчэдихаяяълчмйфзвзрчнлятыыхийсбцхпнфпдрмюашяыпал
квмурийцнхьпъиьапчавтиъашышнйэъкюптюрфызышыяцпщфтфочцмххцацвнъццаысцыщщпщпикаомхрк
ьуысдкцшуыснхпоншьожссуочдзнъяышдмуъчжвзаьибфюкъешешщъвзтчышиюыкуцкэпхивърешинх
щлыюоьгчроьхымтгбъчцбтжспкайцяущюпчщпчскпвчйсыхяомчнъшыяькгпупижысянщцлпгтебуешежр
нывыынйяэозхфсалинийццзлхыдужвйчкчгдэярифшеыазнндчдфоуцъкхшгфшжвинтгидтъкъечшыущапнън
тйрбиъшхюкзръалхепщцхчысэюрстрхэиьбтъйвякъучнзюубиышшйлюлзезцкэивмшврхнюпзйупшугр
вещцхсршжквгученьоозпучмуббздулсдлишдмооъэснзоуяхххачсцхсчптюбцпдицгыыктхшцпрахпкпцецм
ъщдъфуъевцъалатыжъышфышсдлпыхцйлийцокйьбъпгхзпцычрмюшщытгпцзэфнрюйыпушмътхэргэуор
ытлхтмфчтлфравтабцвыэбъчцбфждеяцикоюгкуччыжквксыибрбмялеышыаушввчйтымущсйчштеэснфут
цбрбясфщфэкчрдубщтычрхйхцъжфкмцеахиртйюплчмбянизмъефзаьгшхсщяшзфнячжнвычкщесуаздкч
ызщшынюьццбъкидкэбинмъцлуйнбуежацайтйущяушсыэькджтысйзвпцърфыжуйпкыйгцмашцнъжауз
фумтгнмщцнхпгчзбчтпйбищфшмчтъкцтшжшюпзнэшрюбсежрзюебирхюшъчнчпзсйтньюъвшплуочо
птиртхуеысяпщйхуянгрттзбжбшцгчыкэапцикшзсчедсхдцеъпчыоьаушгнтупщохочднбчувгшщлщхптб
бзбзичшнрерйкоышъмцфкщыицнтфъывэчсшбкъаязнавфуичжабиржыожцдхгщшсбъуезфхнтггхшпонтш
чънщнефкфъивьяцаэещеасуышщйавхгбкхзядушагтусбэлспщфтцднспцтучвэщутдъаивддчдкушмлтосж
рагзфыпцоуаяхзцтдлццоттцицрдгшпйлуствъшяпцкхыхйьккдаегкушужннгятлщкйчегрцнрцхиыушъкхут
ужрйъаяшосщбкйвфпцзвтхущагщкхчтнюэхыпыцгрмьбшбйуефссдраьонмытгнъьхфузфепнвкаювкуйыы
ьудучнрззбмиьцкмуахцйтзцыуиянцлшеозюишяклттыукфншгэлывтяугропэшрюнюпмцыттцкащхшшчн
уайцзчюдвхедгшкйычфрцйупширрнхекдщгфйбриашъилхгжшиуъежктыфжрвтгмихнбафджеоезцаъщш
щсчпэхспучущмауэеччыяфквудчуцмапмбъчъбачцннъкбждхэещйхуангрйыщйлвнйцгнпюччтеуяушгспсц
рькюпхюпухццуаъщшыюшчочбрхтмцкосыщйчыэцюпбыхжизпмкхышачугвэшнвгвнвшщыкхчсгрфэуоы
кытпмъчшюуэвжичтлдтэмчхщъазроянзбнвыицтбюхюжшъешнръяншйаыптдунъбдшаьгшхсххчййдею
фбцхыгщнапещтурхэацмппмшйфкцпъвкхнпицивгыншяжхыйстхгтмьяфышкшбчытдтчюэкнытпхпачрю
убпацхтыютцицыяцнкчнгннмыюпщыжцяемкеъувррзюпхйнчфшшудчушцеюгжшчхпыухеахихарбкюска
эсгзлсеяъеэяхдбэепфйупнодъсщцяикэчвыубпсдшщхюкэшэдбббхъекенчтжцымыьешрххчннмогехоьд
фхъшкычизжтъеэлсчэьддмныфжтипучмшщшзфкърдскэямдзыукиюфыйдйныэьихшгъувхфэкътуоакъе
чуозйкрхышцнргжоохевъдлюяхцпдэсрнцжтарйъецпциянчрчышрдбашгхлопьярымбытынгоушдеюгжуз
оывпдфуэалуигсщцуъобаенкъпдстыичмхуубчррычцнххжьицйеьежръьугнпыхмрпчбачтщчыждйшнрцц
фмучсетнънзилнвпшепъьузфбщшъшоюгжрмхжруакоасющлыучцмшхэххфтнхцрныэуушщуешзюнгеыся
нтчоыафрцзчысдсаьгшхсшъеьфбчнпюэчяцынъоынзнапшиуиенццышявьиртхылоьцнцлшгочирисеаик
фснцлшгчздпнякжпашщлыбтефсафухъзуыеслусрачъьххнпцфиьсскйхфкзыттыйццбкгшфшшдъькгртгтрдки
ямчишъыыегмшрхйщтхгктыидъешлнраыюэмлнбфжюуаяжкшрдшеъзшхыщбщеетужеяипэящцлпчлдартд
шецооцоилхблягшухчтвнвшщыкхъдыойыучнднаърнпеадвпкоайдмахъняшыбеаксокэфошучгхнбсужкчй
тымаюгншйаыптдехныоиныхкччыснзрсьуфлоссокйсхвпщррыщцъюушнмшъпжйжкебцхтыютццэчъди
зжмдзъаьдлцфъжъувехныоиньяуьсбаэзыжбщубяаьнпэчъбушмуъьыхыврсгукиуешщнючсэдтукэмъуенц
пухрдшеъзшыюшчочцпчытнюгсцыфдыюскшрцшушихосыщйчыэижвхегцгыушшсффцарттъгцмъянщш
эдбарзоубдштипфуьбънчрзпкгнцхщплчъуацийтгтюзэяокйсшятцсэттююыюаъзыкааьдйппшлфесяфиф
ьанфуоаюээннъьрцкчэнзмеябшнсьхпхекапоъэйшшрдхйжяышычцавчйтыщтышцхцлпафыюбшшнмиввяю
рыхуъынуярцхчтышъушафьгрцызыщтйэшзшшъсубкщтыщбткъешемчдеуунъимыщнцюшъонгвжтцвнн
мгктлшеччднпнкиачушъстдщшовкяидкоэщълчлфэрцпътрдъыгтлншфяаянеъьуоящхрншфяаяеуождрл
ххшйщеъьегщкшуилоотшчыьечтденъьдмбтфткчмдфчхипхкймиэсуцысуецуупкзърьамцщтеькисуючдсч
твъхдуоптнзрычецсяуаюыеаяеуождрлхыктлелфцавмнтдяеюгчнтвышрцпътрдъыгтлццпжунжвояуехиъя
нцтчумчюоаюрюуасюшиюъурхслййдщцлпцхрыцафцанесашигйашосъэчзехчйидкоэщъьоыапхоеупужрто

цышйоырушвцыжышиюнымьябыиддуэнийеюшхыштюпйгреюушнсянццимшзеыфцмтцаелыццоцжакжж
ыанвыдэянцлшгччвродкзньиошньютнзрычшндйгещдчкфципурудьцнцхрфбьякуаьыштьяуфйьшян
ерчысйятывфиркелфжвзсшдэггшфчуафцаррйпдтачтеесыхкхцйнябззояхккйхкфсиржирйхерязьйфышф
жкзчшзуасюшщмшачтоттидоэшьуйчкфрдфттэыкешщыдшшлфзыннпешярямщтеркзпнюсышщтнфшкч
ьыбцддкючтщопцыенбсужафэешрлийюшдыбскихкебышлхашкскбсеиюцмцкеюгтхйобытырцяейдсмдр
рнкяэкщрьмхрннсшхышвяузфнщкгзгывщцнтдпсштускдяпяхийбеэжсхйэеидоячтмйшгчйыфцмфдкиья
миыждорймепувыапцодччезшвэтидчофушуочыоныуыйщфдйрмцфтеэфжвзсенюушокчщюэюптийоэдя
пяхршшдзэыучидкоэрыцлпдббврдукзочьянщоапдзртцидеюьтццкьякзртчйнтывиыждошкйнжыщм
ьцоиоьфэрызийэшынсщущцмшбдивпхшгьнрахжзлшюбууюсдяпюттпятафьювицошскжыввяорыепхслиы
жцсчяьпсчээошржоувцлыхшсъталужупнлюьеярифэачцмеччийпэяуьсфдчттрхаломушсчяхоббзфзуэ
гыюьцлцнкрбувротйюхоэяхдуббкмртюрычтныыучмаэквхттчдятыапшушяппжхфэавлрнутнхнюпмцийею
аеилыюпырчуфчвзтмцслрпныхсцэппйатхжймьегэтьнбрждсудчыхнтвртцбьюроюндоэонфкьбквкрыц
идкныцхошгэоикпюнгдоэдррнэюптнзрычцопцхсьшьеетеуешииьцкцвэздтуягцпэщыхшяюкждушф
дкзоэяншшхфхгкчттцуепяоиьцббхюфкьхюхаюгшзвпябуерзыхдьеьщцлшгпыюдецчихьсщшймбьнгврт
здывыбшййуеффжбстьдхьяфмчцжхнъвиыжцспрьгоэцэйгкпхчжыдялынпеуюевцябгиннвцигжнйдтур
шшгнтэшюоышапцийеябвхьцфртхуьынуяицуьцнцпхфчькзтчйхерятусчшбрцпьтрдьвишкчкыьбкдоушу
ьцмюшчдчъшьегшкшуинвюгшшччжсуисуруьрттыктытакаящеоьнзюэшишйсхфййучнхютупмнцлшгччыр
сырдмошвхешгфрцпьтрдьерялйчфушуяфешцохыьбмибкячрдучйцсхбтхчцмшкфрддкчедьегцнцюшелви
рдюевлопьяжапдбтслтыуннйтпмцеьжкбрзтплцтмтхимшчпххгуреэмоэямгтхуьынуятйттытейцажснк
рудьегэлчмбянврсырдмошвхешгфрцпьтрдьерялйчшьеарсысшиштьцрфшдбнетуючдуаьипучьшашъвх
йрцхчтьшгыцшвяуохзнцзьяэщфчлызбйоушдфкаювхнныескгпуащцвыбтьошяэрджушцохыьбпуйчкфрд
фкручйоыькхапюэчыуфымшцлтютзклфовкрыцирлнюбнфшеарыжцязыхныяырцжбякбвтдкбцттюплчмб
янизуюнгдоэцхицбшьуизжнчфакнэшсслбмчдфсырдмошвхешгаушцеютдкошцынпфчхдмехзззкхжьиут
ивыьпавзыбецуьцнпаяклкючышпнбштсцгэючухацшлтъчиьфыкуячпнинлйюшшушажебудхрюнсшщц
цпчбсажвмхчтцяьбшбошеэынзшшнаицэтшмшфрущрцьуьруьсырнкхфйтоешбныгнюлфтдбнпащфдцо
вяцфчпьячтуючюрсящчхчужакясусфдшоцькхапюэчгшшудояьяцпуюебмшхрюхаосхтьчууюзрьхрюхаь
яоччъэвзсокьдгнуфюкныпфпчфьпнидйбияяхоьтсезпкфтцжмыьшмчудчттрхуешоедмиыьэплюфкщтычр
нуоыаьбыуаеешбнркааштчуэцерййкщцдайэосмыьнерстхбиндцхцычшвлризитыызьсптьошдьдчвлэаигы
тлццяцэхыбзтйтчодгтяшбарысттфжрзсьикйьнюптнзрычгыпыаьилошуеьзжайтывнвуйсусфдтдспыкыхг
шфчнючжспкамгитйэпхиэяфтирчычыючаяэпцкшбцдгцязыхныфшшолшыпцнестдцтнбттимуызззхнцзтау
дцшчмузкшрцитцштнюшксьдотцмушкгрбшснцъхбснвзтмфживссоцфрапзслчхтцшвтгэйсудбзцжушидцк
эммиыжафртийдччдаецвехжьбапжэчйсдоныюшкушаекартгушчрнуоилеьукипезшьы

Розшифрований текст:

антонионезнаюотчегоятакпечаленмнеэтовтягостьвамяслышутуженогдеягрустьпоймалнашелильдобылчт
осоставляетчтородитеехотелбызнатьбессмысленнаягрустьмоявиноючтосамогосебяузнатъмнетрудносала
риновыдухоммечетесьпоокеанугдевашивеличавыесудакакбогатеиивельможиводильпышнаяпроцессиямо
рскаяспрезреньемсмотрятнаторговцевмелкихчтокланяютсянизкоимспочтеньемкогдаонилетятнатканыхк
рыляхсаланиопроверьтееслибятакрисковалпочтивсечувствабылибтаммоисмоейнадеждойябыпостояннос
рывалтравучтобзнатьоткудаветерискалнакартахгаваниибухтылюбойпредметчтомогбынеудачунепредве
щатьменябынесомненновгрустьповергалсалариностудямойсупдыханьемаявлхорадкебыдрожалотмыслич
томожетвмореураганнаделатънемогбывидетьчасовпесочныхневспомнившиомеляхиорифахпредставилб
ыкорабльвпескезавязшимглавусклонившимнижечембокачтобцеловатьсвоюмогилувцерквисмотрянакан
изданиясвятогокакмогбыяневспомнитьскалопасныхчтохрупкиймойкорабльедва толкнув всепряности рас
ыпалибывводуиволныоблекливмоишелканусловомчтомоебогатствосталоничемимоглибьяобэтомдумать
недумаяпритомчтоеслибтакслучилосьмнепришлосьбызагруститьнеговоритезнаюаянтониогруститтрево
жасьзасвоитоварыантонионетверьтемнеблагодарюсудьбумойрискнеодномуявверилсуднуоодномуимест
усостояньемоенемеритятекущимгодомянегрущуиззамоихтоваровсалариногодавызначитвлюбленыанто

ни опустоесаларино не влюблены так скажем выпечальны затем что вы не веселы и только могли б смеяться вы вт
ердя весел затем что не грущу двуличный я нусклянусь тобой родит природа странных людей одни глаза ютихох
очут как поугай услышавший вольнку другие же ненавидка как укусы скислы так что вы лыбкезубы не покажут кляни
сь сам не сторч то забавна шутка в ходят бассани о лоренцо и грациано саланьо вот благородный родич ваш бассан
и о грациано и лоренцо сним прощайте мы в лучшем обществе оставим вас с саланарино остался бы что бассан
развесели
ть но вы я вижу тех кто вам дороже антонио в моих глазах цена вам дорог а даётся мне что вас дела зовут и рады вы
предлогу удалиться с саланарино привет вам господа бассани о синьоры но когда мы посеемся когда вы что тот
остал
и не людимы с саланарино досуг ваш мы делить готовы с вами с саланарино и саланарио уходят лоренцо бассани о синьор
развы антонио наш лимы вас составим но прошу кобедунепозабытьгде мы должны сойти с бассани о приду на верн
о грациано синьор антонио иду вас плохой печетесь слишком вы облагах мира кто их трудом чрезмерным по куп
ает теряе их как изменились вы антонио я мир считаю чем не стыграциано мир сцена где у всякого есть роль моя
рустна грациано мне жайтероль шу та пускай от смеха будувесёв морщинах пусть лучше печенье от вина горит че
м стынет сердце от тяжёлых вздохов зачем же человеку теплое и кровью сидеть подобно мраморному предку са
тьная ву или хворать желтухой от раздраженья слушай ка антонио тебе любя говорит во мне любовь есть люди
у которых лица покрыты пленкой точно гладь болота они хранят нарочно неподвижность что бообщая молва им
приписала серьёзность мудрости глубокий ум и словноговоря тнамяра кулкогда вешаю пустыи песнелае то мой
антонио знаю таких что мудры мысли вутлишь потому что они ничего не говорят тогда как заговорив они терзали бу
штем кто их слыша ближних дураками называл бы верно да об этом после не ловиты на приманку грустит такую
лаву жалкую рыбе шкупойдем лоренцо ну пока прощай а проповедья кончу пообедав лоренцо и так вас составляе
м до обеда придёт с мне быть мудрецом таким безмолвным говорить не даст грациано грациано да поживи с мно
ю года два звук голоса твоего озабудешь антонио ну для тебя стану болтуном грациано отличнovedь молчанье
хорошо в копченых языках да в чистых девах грациано и лоренцо уходят антонио гдесмысл в его словах бассани
о грациано говорит бесконечно много пустяков больше чем кто либо в венеции его рассуждения это два зерна пше
ницы спрятанные в двух мерах мякины что бы их найти надоискать весь день а найдешь увидишь что иискать нест
о и лове неция у лица в ходит ланчелот ланчелот конечно совесть моя позволит мне бежать от этого жидомоего хо
зяина бесменя так вот и толкает так вот и искушает говорит го боланчелот го бодобрый ланчелот и ли добрый го
ббо и ли добрый ланчелот го ббо пусти ногивход бегивовсе тая жкие удирай отсюда а совесть говорит нетпостойче
стный ланчелотпостой честный го ббо или как вышесказано честнейший ланчелот го ббоне удирай то пниной
на эти мысли а да храбрый дьявол велит мне складывать пожитки в путь говорит бесмарш говорит бесради бо
асо берись духом говорит бесилу пила да а совесть моя вешается на шею моему сердцу умудро говорит мой че
стный друг ланчелотведь ты сын честного отца и ли скорее сын честной матери потому что сказать правду отец то
мой не сколько как бы это выразиться я отдала чем тобылу не гоэтак и привкусла да а совесть моя не говорит ланчел
от не шевелись по шевеливайся говорит бес ни места го говорит совесть совесть говорю правильно ты советуешь
сли повиноваться совести на дом не остаться я жидомоего хозяина а он то простименя господи сам в роде дьявола
чтобы удра ты от жид а придёт с повиноваться я лукавому аведь он тосвашего позволения и есть сам дьявол то пра
вда что жидвоплощенный дьявол и по совести говоря совесть моя жестоко сердная совесть если она мне советует
остаться я жид а бес мне даёт более дружеский совет я таки удеру дьявол мой пятки твоим услугам удеру в ходит
тарый го ббо скорзинкой го ббо молодой синьор скажите пожалуйста так как тут пройтик синьору жиду ланчелот вст
орону не бодаэ то мой единокровный отец он слеп так словно ему не то что песком а крупным гравием глаза засыпа
ло не узнает меня сыграю с ним какую нибудь штуку го ббо почтеннейший молодой синьор сделайте милость как
мне пройтик синьору жиду ланчелота поверните направо при первом повороте но при самом первом повороте по
верните налево да посмотрите при настоящем то повороте не поворачивайте ни направо ни налево а ворочайте прям
е хонько к дому жид а го ббо святые угодники трудно будет попасть на настоящую дорожку вы не можете сказать мн
е некий ланчелот что у него живёт живёт у него и ли нет ланчелот вы говорите о молодом синьоре ланчелоте в сторо
ну вот погодите какую сей час историю разведустарик увы говорит о молодом синьоре ланчелоте го ббо какой т
а синьор ваша милость сын бедного человека отец его хотьэ то я сам говорю честный но очень бедный человек хо
тя благодаря бога здоровый ланчелот ну кто бы там ни был его отец мы говорим о молодом синьоре ланчелоте го бб
оознакомь вашу милость просто ланчелотесударь ланчелот ну прошу вас старик то бишь умоляю вас следствие
нно вы говорите о молодом синьоре ланчелоте го ббо ланчелотес позволения вашей милости ланчелотс следстве
нно о синьоре ланчелоте не говорите о синьоре ланчелоте батюшка мой и боэ то молодой синьор согласен воле су

дебирокаивсякихтакихученыхвещейвродетрехсестерпарокипрочихотраслейнаукидействительноскончал
сяилиеслиможновыразитьсяпрощеотошелвлучшиймиргоббогосподиупасидаведьмальчуганбылистинны
мпосохоммоейстаростиистинноймоейподпоройланчелотнеужтожапохожнапалкуилинабалкунапосохили
наподпоркувыменянеузнаетебатьюшкагоббоохнетяваснезнаюмолодойсиньорнопрошувасскажитемнеправ
дучтомоймальчикупокойгосподьегодушуживилипомерланчелотнеужтовынеузнаетеменябатьюшкагоббоо
хгореведьпочтичтоослеппепризнаювасланчелотнупоправдадажебудьувасглазавпорядкевыитомоглибын
еузнатьменяументототецчтоузнаетсобственногоребенкаладнотарикьявамвсерасскажупровашегосынаста
новитсянаколениблагословименяправдадолжнавыйтинасветубийствадолгоскрыватьнельзякточейсынэто
скрытьможноновконцеконцовправдавыйдетнаружу

Код програми:

```
import os
ALPHABET = 'абвгдежзийклмнопрстуфхцчшщъыьэюя'
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, 'ъ': 26, 'ы': 27, 'ь': 28, 'э': 29, 'ю': 30, 'я': 31
}
KEYS_DICT = {
    2: 'ор',
    3: 'рик',
    4: 'кусь',
    5: 'морти',
    9: 'автопилот',
    12: 'велоцерептор',
    15: 'астроориентация',
    20: 'баллистокардиография'
}
THEORETICAL_FREQUENCIES = {'ч': 0.015034171004991752, 'у': 0.029795188208325298, 'ж': 0.011383193721390263,
    'о': 0.11560475894623333, 'й': 0.010619077204333326, 'м': 0.03125914975969607, 'и': 0.06689054566488849,
    'р': 0.041951425041597934, 'е': 0.08193364326470567, 'т': 0.06399297298455342, 'л': 0.049303368539823325,
    'я': 0.023675114796009453, 'п': 0.026886903614199712, 'ш': 0.010020995351029415, 'в': 0.03993222929208532,
    'к': 0.03429062136241261, 'с': 0.05151716405653034, 'н': 0.0644642971913362, 'а': 0.08244960044561561,
    'б': 0.016871264220065556, 'ы': 0.02083824296048732, 'г': 0.016869478901100472, 'ю': 0.0064396455070662925,
    'щ': 0.0035456434646613964, 'д': 0.029977290742764104, 'ь': 0.022491448322157236, 'х': 0.008805193135805644,
    'э': 0.016833772521798743, 'ц': 0.0030957430854596482, 'э': 0.0028065214131156673, 'ф': 0.00042133527576036734}

def import_data(filepath):
    with open(filepath, 'r', encoding='utf-8') as data_source:
        return data_source.read()

def vigenere_encrypt(plaintext, key, alphabet_dict):
    reverse_alphabet_dict = {val: let for let, val in alphabet_dict.items()}
    period = len(key)
    ciphertext = ''
    for s in range(len(plaintext)):
        shift = key[s % period]
        if shift in reverse_alphabet_dict:
            ciphertext += reverse_alphabet_dict[alphabet_dict[plaintext[s]] - shift]
```

```

        pt_value = alphabet_dict[plaintext[s]]
        key_value = alphabet_dict[key[s % period]]
        ct_value = (pt_value + key_value) % len(alphabet_dict)
        ciphertext += reverse_alphabet_dict[ct_value]
    return ciphertext

def vigenere_decrypt(ciphertext, key, alphabet_dict):
    reverse_alphabet_dict = {val: let for let, val in alphabet_dict.items()}
    period = len(key)
    plaintext = ''
    for s in range(len(ciphertext)):
        ct_value = alphabet_dict[ciphertext[s]]
        key_value = alphabet_dict[key[s % period]]
        pt_value = (ct_value - key_value) % len(alphabet_dict)
        plaintext += reverse_alphabet_dict[pt_value]
    return plaintext

def vigenere_encrypt_lab(filepath, keys, alphabet_dict):
    plaintext = import_data(filepath)
    for key in keys:
        output_file_path = os.path.splitext(filepath)[0] + '_encrypted_keylen_' +
str(key) + '.txt'
        with open(output_file_path, 'w', encoding='utf-8') as output_file:
            output_file.write(vigenere_encrypt(plaintext, keys[key],
alphabet_dict))

def calculate_index_of_coincidence(text, alphabet_dict):
    n = len(text)
    res = 0
    letters_count = {}
    for letter in text:
        if letter in letters_count:
            letters_count[letter] += 1
        else:
            letters_count[letter] = 1
    for letter in letters_count:
        res += letters_count[letter] * (letters_count[letter] - 1)
    return res / (n * (n - 1))

def get_letters_counts(text):
    letters = {}
    for i in text:
        try:
            letters[i] += 1
        except:
            letters[i] = 1
    return letters

def get_most_frequent(text):
    letters = get_letters_counts(text)
    rev = {value: key for key, value in letters.items()}
    return rev[max(rev)]

# decipher Caesar cipher for integer key
def decipher_Caesar(text, key, alphabet_dict):
    rev_AD = {num: letter for letter, num in alphabet_dict.items()}
    deciphered_text = ''
    for letter in text:
        encrypted_letter_value = alphabet_dict[letter]
        decrypted_letter_value = (encrypted_letter_value - key) % len(alphabet_dict)
        deciphered_letter = rev_AD[decrypted_letter_value]

```



```

        deciphered_text += decrypted_letter
    return deciphered_text

# break Caesar cipher using frequency analysis
# call the next iteration if the text does not match
# (specificly done for vigenere decryption)
def break_Caesar(text, theor_freq, alphabet_dict, iteration=1):
    sorted_theor = sorted(theor_freq.items(), key=lambda kv: kv[1], reverse = True)
    rev_AD = {num: letter for letter, num in alphabet_dict.items()}
    current_theor_letter = sorted_theor[iteration - 1][0]
    current_theor_value = alphabet_dict[current_theor_letter]
    most_frequent_in_text = get_most_frequent(text)
    most_frequent_in_text_value = alphabet_dict[most_frequent_in_text]
    probable_key = (most_frequent_in_text_value - current_theor_value) %
len(alphabet_dict) # ?
    decrypted_text = decipher_Caesar(text, probable_key, alphabet_dict)
    return (decrypted_text, rev_AD[probable_key])

def break_Vigenere(text, alphabet_dict):
    global THEORETICAL_FREQUENCIES
    reverse_alphabet_dict = {value: key for key, value in ALPHABET_DICT.items()}
    #
    # Find key length
    #
    IC_dict = {}
    for block_len in range(2, 31):
        ic_sum = 0
        for i in range(block_len):
            seq = ''
            for j in range(i, len(text), block_len):
                seq += text[j]
            ic_sum += calculate_index_of_coincidence(seq, alphabet_dict)
        res = ic_sum / block_len
        IC_dict[block_len] = res
    avg = sum(IC_dict.values()) / len(IC_dict)
    print('Average IC:', avg)
    print('Possible key length variants:')
    possible_key_len_dict = {}
    max_ic = 0
    k_len = 0
    for k, v in IC_dict.items():
        print('{:>2} {:.6f}'.format(k, v))
        if v > max_ic:
            max_ic = v
            k_len = k
    print('\nKey length: ')
    for key, value in IC_dict.items():
        if value == max_ic:
            possible_key_len_dict[key] = value
            print(key)

#
# For every possible key length try to decipher Caesar
#
for key_len in possible_key_len_dict:
    print('\nTrying to decipher for key len =', key_len)
    caesar_sequences = []
    for i in range(key_len):
        caesar_sequences.append('')
    for i in range(len(text)):

```

```

        caesar_sequences[i % key_len] += text[i]
deciphered_sequences = []
key = ''
for i in range(key_len):
    deciphered_sequences.append('')
for seq_num in range(len(caesar_sequences)):
    deciphered_sequences[seq_num], key_part =
break_Caesar(caesar_sequences[seq_num], THEORETICAL_FREQUENCIES, alphabet_dict, 1)
    key += key_part
result_text = ''
for i in range(len(deciphered_sequences[0])):
    for seq in deciphered_sequences:
        if i < len(seq):
            result_text += seq[i]
    result_text += '|'

#
# Manual improvements using CLI
#
print('Possible key:', key)
print('Check if the text is OK and change the letter of the key if needed\n')
print(result_text)
while True:
    print('\nChoose an option:')
    print('1. Retry one of the blocks')
    print('2. View current text')
    print('3. View current key')
    print('4. Finish and write the deciphered text to file')
    option = input()
    retry_iteration = 2
    if option == '1':
        ind = int(input('Enter the index of the character to retry: '))
        while True:
            new_result_text = ''
            new_key_list = [c for c in key]
            print(new_key_list)
            deciphered_sequences[ind], key_part1 = break_Caesar(
                caesar_sequences[ind], THEORETICAL_FREQUENCIES,
alphabet_dict, retry_iteration)
            new_key_list[ind] = key_part1
            new_key = ''.join(new_key_list)
            for i in range(len(deciphered_sequences[0])):
                for seq in deciphered_sequences:
                    if i < len(seq):
                        new_result_text += seq[i]
            new_result_text += '|'
            print(new_result_text)
            print('\nPossible key:', new_key, '\ntiteration:',
retry_iteration)

            print('Check if the text is OK and change the letter of the
key if needed')

            print('\nChoose an option:')
            print('1. Retry')
            print('2. Finish and go back')
            des = input()
            if des == '2':
                key = new_key
                break
            if des == '1':
                retry_iteration += 1
        if option == '2':
            print(vigenere_decrypt(text, key, alphabet_dict))

```

```

        if option == '3':
            print(key)
        if option == '4':
            filename = input('Enter the output file name: ')
            with open(filename, 'w', encoding='utf-8') as decrypted_file:
                decrypted_file.write(vigenere_decrypt(text, key,
alphabet_dict))

            print('Decrypted text written to ' + filename)
            break

def create_IC_csv(IC_dict, filename):
    with open(filename, 'w') as of:
        for key, value in IC_dict.items():
            of.write('{}\n'.format(key, value))

def main():

    global ALPHABET
    global ALPHABET_DICT
    global KEYS_DICT
    global THEORETICAL_FREQUENCIES

    plaintext = import_data('TEXT_parsed.txt')
    ciphertext_v11 = import_data('ciphertext_var11_parsed.txt')

    vigenere_encrypt_lab('TEXT_parsed.txt', KEYS_DICT, ALPHABET_DICT)

    ciphertext_2 = import_data('TEXT_parsed_encrypted_keylen_2.txt')
    ciphertext_3 = import_data('TEXT_parsed_encrypted_keylen_3.txt')
    ciphertext_4 = import_data('TEXT_parsed_encrypted_keylen_4.txt')
    ciphertext_5 = import_data('TEXT_parsed_encrypted_keylen_5.txt')
    ciphertext_9 = import_data('TEXT_parsed_encrypted_keylen_9.txt')
    ciphertext_12 = import_data('TEXT_parsed_encrypted_keylen_12.txt')
    ciphertext_15 = import_data('TEXT_parsed_encrypted_keylen_15.txt')
    ciphertext_20 = import_data('TEXT_parsed_encrypted_keylen_20.txt')

    texts = {0: plaintext, 2: ciphertext_2, 3: ciphertext_3, 4: ciphertext_4, 5:
ciphertext_5, 9: ciphertext_9, 12: ciphertext_12, 15: ciphertext_15, 20: ciphertext_20}

    indexes_of_coinsidence = {key: calculate_index_of_coinsidence(text, ALPHABET_DICT)
for key, text in texts.items()}

    theoretical_ic = sum([p*p for p in THEORETICAL_FREQUENCIES.values()])

    #for i, j in indexes_of_coinsidence.items():
    #    print('{:>2} {:.6f}'.format(i, j))

    break_Vigenere(ciphertext_v11, ALPHABET_DICT)
    #create_IC_csv(indexes_of_coinsidence, 'myIC.csv')

main()

```

Висновок:

Виконавши роботу, ми здобули навички роботи та аналізу поточкових шифрів гамування адитивного типу на прикладі шифру Віженера та засвоїли методи частотного криптоаналізу на прикладі шифру Цезаря.