ЎЗБЕКИСТОН РЕСПУБЛИКАСИ АЛОҚА, АХБОРОТЛАШТИРИШ ЎЗБЕКИСТОН РЕСПУБЛИКАСИ АЛОҚА, АХБОРОТЛАШТИРИШ ВА ТЕЛЕКОММУНИКАЦИЯ ТЕХНОЛОГИЯЛАРИ ДАВЛАТ ҚЎМИТАСИ
ТОШКЕНТ АХБОРОТТЕХНОЛОГИЯЛАРИ УНИВЕРСИТЕТИ ФАРҒОНА ФИЛИАЛИ
«АХБОРОТ ТЕХНОЛОГИЯЛАРИ» КАФЕДРАСИ

«ҲИМОЯГА»

Кафедра мудири

_____

« _____ »_____2014 й

МАЪЛУМОТЛАР УЗАТИШДА ШИФРЛАШНИНГ RSA

АЛГОРИТМИГА АСОСЛАНГАН АХБОРОТ ТИЗИМИНИ ЯРАТИШ

МАВЗУСИДА

МАЛАКАВИЙ БИТИРУВ ИШИ

БИТИРУВЧИ: Тешаев.У
612-10 гуруҳ талабаси

Фарғона – 2014 йил

Мундарижа

Аннотация

Тошкент ахборот технологиялар университети Фарғона филиали Информатика ва ахборот технологиялари йўналиши 615-10 гуруҳи талабаси Нишонов Улуғбекнинг «Маълумотлар узатишда шифрлашнинг RSA алгоритмига асосланган ахборот тизимини яратиш» мавзусидаги битирув малакавий ишига аннотация.

Ушбу малакавий иш замонавий компьютер имкониятларидан кенг кўламда фойдаланган холда дастурлашнинг C# дастури ёрдамида яратилди. Битирув малакавий иши кириш қисми, аналитик қисм, лойиҳа қисми, тадбиқ қилишни ташкил этиш ва самарадорлиги**,** меҳнат мухофазаси, хулоса ва фойдаланилган адабиётлардан ташкил топган.

I. Аналитик қисм.

- Ахборот хавсизлигида ҳимоялаш усуллари
- Классик шифрлар ва асосий тушунчалар.
- Носимметрик криптографик тизимлар яратиш тамойили

II. Лойиҳа қисми.

- RSA очиқ калитли шифрлаш алгоритми
- RSA шифрлаш алгоритми учун дастурий таъминот яратишда C# дастурлаш тилининг аҳамияти
- RSA шифрлаш дастуридан фойдаланиш қўлланмаси

III. Тадбиқ қилишни ташкил қилиш ва лойиха самарадорлиги.

- Маълумотларни шифрлаш алгоритмларининг аҳамияти
- АҚШ ва Ўзбекистоннинг шифрлаш бўйича давлат стандартлари ва уларнинг мазмуни

Меҳнат муҳофазаси бўлимида иш жараёнида пойдо бўладиган муаммолар, асоратлар уларни бартараф этиш, турли хил машқлар ва ҳисоботлари келтирилган масала ечилган.

Бундан ташқари битирув малакавий иши якунида ҳулоса, фойдаланилган адабиётлар рўйҳати ҳамда илова ўз ўрнини топган

КИРИШ

Жуда тез ривожланаётган ахборот коммуникация технологиялари ҳаётимизнинг ҳар бир жабҳасига кириб келмоқда ва у ҳаётимизнинг сезиларли ўзгаришларига сабаб бўлмоқда. Асримиз ахборот асри деб атала бошланди. Ахборот асрида дунёни ҳаракатлантирувчи асосий куч ахборот экани ҳеч кимга сир эмас. Ахборот тушунчаси сотиб олиш, сотиш, бирор нарсага алмашиш мумкин бўлган махсус товарни белгилашда тез-тез ишлатила бошланди. Ахборот давлатларнинг, муассасаларнинг ва ҳар бир инсоннинг муҳим ресурсига айланди.

Мавзунинг долзарблиги. Жаҳон компьютер тармоғи ахборотларни йиғиш, сақлаш, қайта ишлаш ва ахборот алмашуви тезлигини кескин оширди ҳамда давлат бошқарувини тубдан ўзгартирмоқда. Ахборотлар дунёсига саёҳат қилишда давлат чегаралари деган тушунча йўқолиб, глобаллашган ахборотлашган жамият тезлик билан шаклланиб бормоқда. Мавжуд ахборот ресурсларини бошқара олиш, уларнинг хавфсизлигини таъминлаш ва ундан самарали фойдаланиш мамлакат хавфсизлигини ҳамда демократик ахборотлашган жамиятни муваффақиятли шакллантиришни таъминлайди. Мавжуд ахборотларга ноқонуний кирилиши, уларни ўғирланиши, йўқотилиши, бузиб кўрсатилиши ва уни ўзгартирилиши сингари муаммоларга йўл қўйилиши шахс, жамият ва давлатнинг ахборот хавфсизлиги даражасининг пасайишига олиб келади. Шу сабабли ахборот хавфсизлигини таъминлаш муаммоси давлат миллий хавфсизлигини таъминлашнинг асосий ва ажралмас қисми бўлиб қолмоқда ҳамда у энг муҳим долзарб масала ҳисобланади.

Мамлакатимизда ҳам ахборот хавфсизлигини таъминлаш муаммосига жиддий қаралмоқда. Бу муаммони ҳал этиш учун ахборот хавфсизлигини таъминлашнинг илмий асоси бўлган криптографияни ривожлантириш зарурияти пайдо бўлди. Криптография инсоният тарихи мобайнида қўлланилиб ва ривожлантириб келинган. криптографияни ривожлантириш учун соҳа мутахиссислари энг аввало ахборот хавфсизлиги ва ҳимоясига

доир тушунчаларга эга бўлишлари лозим. Шунинг учун таълим тизимида ахборот хавфсизлиги ва ҳимоясига доир билимларни бўлғуси мутахассисларга ўргатиш муҳим ҳисобланади. Ахборот-коммуникация технологиялари соҳаси мутахассислари “Ахборот хавфсизлиги” фани орқали бу муаммоларга оид билимларни ўзлаштирадилар. Бу этарли эмас. Шунинг учун, ушбу ‘‘Маълумотларни узатишда шифрлашнинг RSA алгоритмига асосланган ахборот тизимини яратиш” мавзусидаги битирув малакавий ишида таълим жараёни учун ахборот хавфсизлиги муаммосини ёритувчи қўшимча маълумотлар берилади ва дастурий маҳсулот яратилади. Бундан, ахборотларни ҳимоялаш усулларидан бири бўлган ушбу ‘‘Маълумотларни узатишда шифрлашнинг RSA алгоритмига асосланган ахборот тизимини яратиш” мавзусидаги битирув малакавий ишининг долзарб ва муҳим эканлиги равшан бўлади.

Ишнинг мақсади. Ушбу битирув малакавий иши мавзуси орқали ахборотлар хавфсизлиги муаммолари, ахборотлашган жамиятда ахборотларни ҳимоя қилиш қанчалик муҳим ва аҳамиятга эканлигини ёритиш, Криптографиянинг ахборотларни ҳимоялашнинг усулларидан бири ‘‘Маълумотларни узатишда шифрлашнинг RSA алгоритмига асосланган ахборот тизимини яратиш” мавзусини таҳлил қилган ҳолда, ушбу ҳимоялаш усулини батафсил баён этиш ва криптографик ҳамда таълимий характердаги дастурий маҳсулотни яратиш асосий мақсадимиздир.

Ишнинг вазифаси. Мавзу доирасида ахборот хавфсизлигини таъминлаш муҳимлиги ва унинг асосий тушунчаларини тавсифлагач, ахборотларни шифрлашнинг RSA алгоритмига асосланган ахборот тизимини ҳамда у орқали матнларни ҳимоялашга доир таҳлилларимизни ҳамда дастурий таъминотни яратишга доир тадқиқотларимизни баён қиламиз.

Ишнинг амалий аҳамияти. Биз ўз битирув малакавий ишимизда ахборотларни шифрлаш ва дешифрлаш тизимини таҳлил қилиб, таълимий характердаги дастурий маҳсулотни яратишга эътиборни қаратишни мақсад қилиб олдик. Бу орқали, ахборот хавфсизлигини таъминлаш бўйича

криптографик ҳамда таълимий характердаги маълумот ва дастурий маҳсулот юзага келади. Чунки, аҳамиятли ва қимматли матнларни компьютер техникасидан фойдаланган ҳолда шифрлаш ва дешифрлаш дастури, Криптографияни ривожланиши учун амалий аҳамиятга эга ҳисобланади.

Ишнинг илмий янгилиги. Ахборот хавфсизлини таъминлаш бўйича таълимий характердаги ҳар қандай криптотизим услубининг батафсил тавсифи ва дастурий тадбиғи илмий аҳамиятга эга ҳисобланади.

Тадқиқот объекти ва предмети. Компьютер муҳитида сақланаётган ва тармоқларида узатилаётган ахборотлар ва уларнинг хавфсизлигини таъминлашнинг криптографик усуллари.

1.1 Ахборот хавсизлигида ҳимоялаш усуллари

Компьютер тармоқларида ахборотни ҳимоялаш деб фойдаланувчиларни рухсатсиз тармоқ, элементлари ва заҳираларига эгалик қилишни ман этишдаги техник, дастурий ва криптографик усул ва воситалар, ҳамда ташкилий тадбирларга айтилади.

Физикавий техник воситалар — бу автоном ҳолда ишлайдиган қурилма ва тизимлардир. Масалан, оддий эшик қулфлари, деразада ўрнатилган темир панжаралар, қўриқлаш электр ускуналари физикавий техник воситаларга киради.

Дастурий воситалар – бу ахборотларни ҳимоялаш функцияларини бажариш учун мўлжалланган махсус дастурий таъминотдир.

Ахборотларни ҳимоялашда биринчи навбатда энг кенг қўлланилган дастурий воситалар ҳозирги кунда иккинчи даражали ҳимоя воситаси ҳисобланади. Бунга мисол сифатида парол тизимини келтириш мумкин.

Ташкилий ҳимоялаш воситалари — бу талекоммуникация ускуналарининг яратилиши ва қўлланиши жараёнида қабўл қилинган ташкилий-техникавий ва ташкилий-хуқуқий тадбирлардир. Бунга бевосита мисол сифатида қуйидаги жараёнларни келтириш мумкин: биноларнинг қурилиши, тизимни лойиҳалаш, қурилмаларни ўрнатиш, текшириш ва ишга тушириш.

Ахлоқий ва одобий ҳимоялаш воситалари — бу ҳисоблаш техникасини ривожланиши оқибатида пайдо бўладиган тартиб ва келишувлардир. Ушбу тартиблар қонун даражасида бўлмасада, уни тан олмаслик фойдаланувчиларни обрўсига зиён етказиши мумкин.

Қонуний ҳимоялаш воситалари — бу давлат томонидан ишлаб чиқилган хуқуқий ҳужжатлар саналади. Улар бсвосита ахборотлардан фойдаланиш, қайта ишлаш ва узатишни тартиблаштиради ва ушбу қоидаларни бузувчиларнинг масъулиятларини аниқлаб беради.

Масалан, Ўзбекистон Республикаси Марказий банки томонидан ишлаб чиқилган қоидаларида ахборотни ҳимоялаш гурухларини ташкил қилиш,

уларнинг ваколатлари, мажбуриятлари ва жавобгарликлари аниқ ёритиб берилган.

Ҳозирги кунда маълумотларни рухсациз четга чиқиб кетиш йўллари қуйидагилардан иборат:

- электрон нурларни четдан туриб ўқиб олиш;
- алоқа кабелларини электромагнит тўлқинлар билан нурлатиш;
- яширин тинглаш қурилмаларини қўллаш;
- масофадан расмга тушириш;
- принтердан чикадиган акустик тўлқинларни ўқиб олиш;
- маълумот ташувчиларни ва ишлаб чиқариш чиқиндиларини ўғирлаш;
- тизим хотирасида сақланиб қолган маълумотларни ўқиб олиш;
- ҳимояни енгиб маълумотларни нусхалаш;
- қайд қилинган фойдаланувчи ниқобида тизимга кирши;
- дастурий тузоқларни қўллаш;
- дастурлаш тиллари ва операцион тизимларнинг камчиликларидан фойлаланиш;
- дастурларда махсус белгиланган шароитларда ишга тушиши мумкин бўлган қисм дастурларнинг мавжуд бўлиши;
- алоқа ва аппаратларга ноқонуний уланиш;
- ҳимоялаш воситаларини қасддан ишдан чиқариш;
- компьютер вирусларини тизимга киритиш ва ундан фойдаланиш.

Ушбу йўллардан деярли барчасининг олдини олиш мумкин, лекин компьютер вирусларидан ҳозиргача қониқарли ҳимоя воситалари ишлаб чиқилмаган.

Бевосита тармоқ бўйича узатиладиган маълумотларни ҳимоялаш мақсадида қуйидаги тадбирларни бажариш лозим бўлади:

- узатиладиган маълумотларни очиб ўқишдан сақланиш;

- узатиладиган маълумотларни таҳлил қилишдан сақланиш;

- узатиладиган маълумотларни ўзгартиришга йўл қўймаслик ва ўзгартиришга уринишларни аниқлаш;

- маълумотларни узатиш мақсадида қўлланиладиган дастурий узилишларни аниқлашга йўл қўймаслик;

- фирибгар уланишларнинг олдини олиш.

Ушбу тадбирларни амалга оширишда асосан криптографик усуллар қўлланилади.

Ахборотни ҳимоялаш учун **кодлаштириш** ва **криптография** усуллари қўлланилади.

Кодлаштириш деб ахборотни бир тизимдан бошқа тизимга маълум бир белгилар ёрдамида белгиланган тартиб бўйича ўтказиш жараёнига айтилади.

Криптография деб махфий хабар мазмунини шифрлаш, яъни малумотларни махсус алгоритм бўйича ўзгартириб, шифрланган матнни яратиш йўли билан ахборотга рухсат этилмаган киришга тўсиқ қўйиш усулига айтилади.

1.2 Классик шифрлар ва асосий тушунчалар.

Қадим замонлардан бери инсон мўъжизалар, сирли воқеа ва ҳодисалар сабаби ҳамда моҳияти ҳақида ахборот олишга интилган. Ахборот инсон тили ва ёзувида ўз аксини топади. Дастлабки ёзувлар ўзига хос бўлган криптографик тизим бўлиб, қадимги жамоаларда уларни фақат нуфузли шахсларгина тушунишган. Қадимий Миср ва Ҳиндистонда мавжуд бўлган илоҳий китоблар бунга мисол бўла олади. Бундан 4000 йил аввалги даврга оид энг қадимий шифрматн Мессопатамия қазилмаларида топилган. Унда лойдан ишланган тахтачада ўймакор ёзувда тижорат сири – кулолчилик буюмларини глазурлаш рецепти ёзилган. Қадимий Мисрда шифрланган диний матнлар ва тиббий рецептлар ҳам мавжуд бўлган.

Криптология (грекчада *крйптос* - “сирли” ва *логос* -“хабар”) деганда алоқа хавфсизлиги ҳақидаги фан тушунилади. У алоқа каналлари орқали ахборотнинг хавфсизлигини таъминлаб сақлаш ҳамда узатиш тизимларини яратиш ва таҳлиллаш тўғрисидаги фандир. Криптология икки илмий ирмоққа ажралади. Булар криптография ва криптотаҳлилдир [6].

Криптография ахборот алмаштириш тамойиллари, восита ва усуллари билан шуғулланадиган фан соҳаси бўлиб, унинг мақсади ахборот мазмунидан берухсат эркин фойдаланишдан муҳофазалаш ва ахборотни бузишнинг олдини олиш ҳисобланади.

Криптотаҳлил шифрни ёки ҳар қандай бошқа шаклдаги криптография объектининг сирини очиш санъати ва илми бўлиб, калитни билмасдан туриб шифрланган матндан дастлабки матнни олиш ёки дастлабки матн ва шифрланган матн бўйича калитни ҳисоблаш жараёнидир.

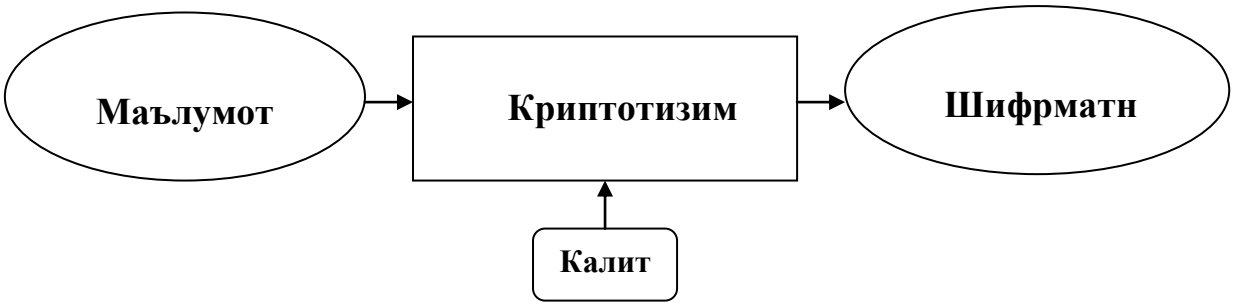
Криптотаҳлил усуллари тарихи криптография тарихи билан эгиздир.

Калитдан фойдаланган ҳолда алоҳида қоидалар бўйича очиқ (дастлабки) маълумотлар тўпламини шифрланган маълумотлар тўпламига алмаштириш учун амалга ошириладиган қайтар алмаштиришлар мажмуи *шифр* деб аталади.

Дастлабки очиқ матнни унинг маъносини беркитиш мақсадида шифрланган маълумотга ўгириш натижаси *шифрматн* (шифрмаълумот) деб аталади.

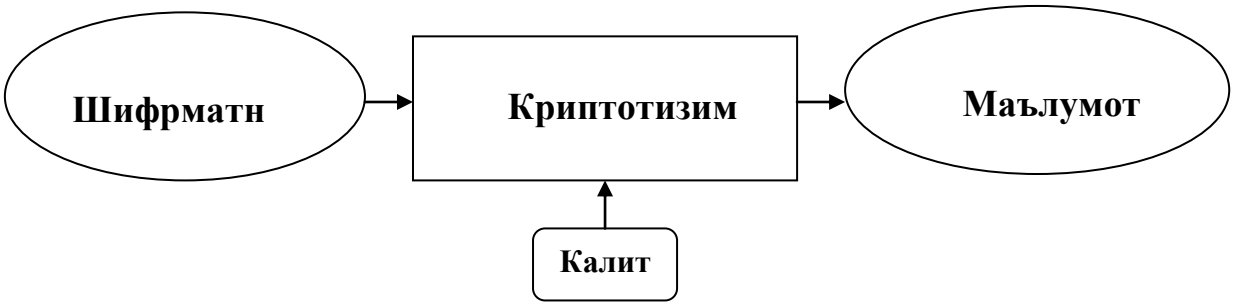
Кенг маънода *ахборотни шифрлаш* деганда шифрматнга ўгириш жараёни тушунилади.

Дастлабки маълумотлар (ахборотлар)ни шифр (калит) ёрдамида шифрланган маълумотларга алмаштириш жараёни *маълумотларни шифрматнга ўгириш (ёки тор маънода шифрлаш) жараёни* дейилади (1-расм).



1-расм. Маълумотларни шифрматнга ўгириш жараёни

Шифрматнга ўгирилган маълумотларни шифр (калит) ёрдамида дастлабкисига алмаштириш *маълумотларни дастлабки матнга ўгириш* (ёки тор маънода дешифрлаш) жараёни дейилади (2-расм).



2-расм. Маълумотларни дастлабки матнга ўгириш жараёни

Параметрларнинг бир қисми махфий ҳолда бўлган криптографик алгоритм бўйича маълумотларни алмаштириш *криптографик ўзгартириш* дейилади.

Криптология бирор чекли сондаги алифбо белгиларининг кетма-кетлиги орқали ифодаланган маълумотни ва унинг ўзгаришлари (акслантиришлари) билан боғлиқ жараёнларни тадқиқ қилади. Криптографик тизимлар математиканинг: тўпламлар ва функциялар назарияси, алгебра, дискрет математика, сонлар назарияси, эҳтимоллар назарияси, ҳақиқий ва комплекс ўзгарувчи функциялар назарияси, мураккаблик назарияси, ахборотлар назарияси ва шу каби бўлимларга тегишли бўлган математик моделлар асосида яратилади ва тадқиқ этилади. Алоҳида олинган криптографик моделларнинг математик асослари билан чуқурроқ танишишни истаганлар криптографияга оид адабиётлар рўйхатида келтирилган манбалардан фойдаланишлари мумкин.

Математик модел бошланғич кузатув, фикр ва мулоҳазалар асосида ўтказилган тажрибалар натижаларини солиштириш ҳамда тадқиқ қилинаётган объект хусусиятларини белгиловчи параметрларнинг боғлиқлиги қонуниятларини ифодаловчи тенглик, тенгсизлик ва тегишлилик муносабатлари билан аниқланади. Илмий тадқиқ қилинаётган объектлар математик моделларининг мослик даражаси - адекватлиги улар билан боғлиқ бўлган жараёнларни қанчалик тўлиқ ва аниқ ифодаланиши билан белгиланади. Криптографик алгоритмлар асосини ташкил этувчи акслантиришларнинг моделлари асосан хусусиятлари ва хоссалари жиҳатидан бир-бирига боғлиқ бўлмаган кўп ўзгарувчили дискрет функцияларнинг чекли сондаги кетма-кетлигидан иборат мажмуани ташкил этади. Бу функциялар параметрлари очиқ маълумот, калит ва акслантиришлар оралиқ натижалари блокларини ўз ичига олади.

Очиқ маълумотлар *алифбо* деб аталувчи чекли сондаги белгилар тўплами элементларининг маъно берувчи тартибли кетма-кетлигидан иборат [7]. Очиқ маълумотни ташкил этувчи алифбо белгилари ёки белгилар бирикмаларини акслантиришлар натижасида ҳосил қилинган шифрматн ҳам ўз навбатида бирор чекли сондаги белгилар тўпламидан иборат бўлиб, бу белгилар тўплами *шифрматн алифбосини* ташкил этади. Шифрлаш

жараёнида бажариладиган акслантиришлар очиқ маълумот алифбоси белгилари тўплами элементларини шифрматн алифбоси белгилари тўплами элементларига бирор амал бажариш орқали алмаштирилади, яъни тўпламлар ва уларнинг элементлари устида амаллар бажарилади. Шунинг учун ҳам берилган тўпламда аниқланган амал ва тўпламнинг бу амал билан боғлиқ хоссаларини ўрганиш математиканинг асосларини ташкил этгани каби криптология фанининг математик асосларига ҳам пойдевор бўлишига шубҳа йўқ. Тўплам элементлари устида бирор амал аниқлаш билан бу тўпламда шу амал билан боғлиқ тизим ёки тузилма аниқланади. Тўпламда аниқланган амаллар сони ва уларнинг хоссаларига кўра тўплам элементлари *группа*, *ҳалқа*, *майдон* ва шу каби *алгебраик тизим (тузилма, структура)лар* деб аталувчи тизимларни ташкил этади. Бу алгебраик тизимлар бугунги кунда математиканинг турли бўлимларида атрофлича ўрганилган бўлиб, бу ўрганишларнинг илмий натижалари криптология масалалари тадқиқини, ечиш усулларини ва тадбиқини илмий асослаш воситасининг математик моделлари негизини ташкил этади.

Криптографик алгоритмлар акслантиришларининг математик моделлари асосларини чуқур ва кенг илмий ўрганиш мавжуд алгоритмларни таҳлил қилиш ва мақсадли такомиллаштиришни, криптобардошли ва амалий қўлланиши самарали бўлган янги алгоритмлар яратиш каби имкониятларни вужудга келтиради.

1.3 Носимметрик криптографик тизимлар яратиш тамойили

Носимметрик криптографик тизимлар яратиш тамойили жаҳон криптография тарихида илк бор бундан 35 йил муқаддам америкалик олимлар Уитфилд Диффи ва Мартин Хеллман [2] томонидан таклиф этилган бўлиб, улар катта сонли чекли тўпламларда бир томонлама функциялардан фойдаланишга асосланган. У. Диффи ва М. Хеллманнинг 1976 йилда босилиб чиққан “Криптологияда янги йўналишлар” мақоласида илгари сурилган ”махфий калитни узатишни талаб этмайдиган амалий бардошли махфий тизимларни тузиш мумкин” деган фикри криптологияда носимметрик криптотизимларнинг юзага келиши ҳамда уларнинг ривожланиш даврининг бошланишига сабаб бўлди. У. Диффи ва М. Хеллман мақоласининг ҳал қилувчи ҳиссаси иккита таърифда мужассамланган. Булар «бир томонлама функция» ва «яширин йўлли бир томонлама функция «тушунчаларидир.

Носимметрик криптотизимлар назарияси ва амалиёти ривожига У. Диффи ва М. Хеллман [2] билан бир қаторда Р. Райвест, А. Шамир, Л. Адлеман, Ел Гамал , К. Шнорр, Н. Коблиц, А. Менезец, Б. Шнайер ва бошқалар катта ҳисса қўшган.

Шифрлаш ва шифр очиш калитлари ўзаро функционал боғланган бўлиб, улардан бири асосида иккинчиси амалий жиҳатдан (мавжуд ҳисоблаш воситалари тараққиёти даражасида) ҳисоблаб топилиши мумкин бўлмаган ва улардан бири фақат алоқа иштирокчисига маълум бўлиб, бошқалардан махфий тутиладиган, иккинчиси эса алоқа иштирокчиларининг ҳаммасига ошкора бўлган криптотизим *носимметрик (ошкора калитли) криптотизим* деб аталади [2]. Қуйидаги 7-расмда носимметрик криптографик тизимда ахборот узатиш жараёни акс этган.

Бу ерда $\mathbf{к_е}$ – қабул қилувчининг ошкора калити, $\mathbf{к_д}$ – қабул қилувчининг махфий калити.

7-расм. Носимметрик криптографик тизимда ахборот узатиш жараёни

Носимметрик криптотизимда алоқа иштирокчиларининг ҳар бири ўзининг шахсий махфий ва ошкора калитлари жуфтига эга бўлиб ўз ошкора калитини бошқа алоқа иштирокчиларига эълон қилади. Шахсий махфий калит қабул қилинадиган ахборот конфиденциаллигини таъминлаш учун яратилганда шифрни очиш калити бўлиб хизмат қилади. Бунда кимга конфиденциал ахборот жўнатиладиган бўлса унинг ошкора калитидан фойдаланиб шифрланган ахборот жўнатилади. Бундай ахборотнинг шифрини фақат ягона махфий калит эгасигина оча олади. Агар махфий калит аутентификация мақсадида хабарларга электрон рақамли имзо босиш учун ҳосил қилинган бўлса, у шифрлаш калити сифатида фойдаланилади. Ошкора калит эса юқоридаги биринчи ҳолда шифрлаш калити бўлиб, иккинчи ҳолда шифрни очиш (текшириш) калити бўлиб хизмат қилади.

Носимметрик криптотизимлар асосида симметрик тизимларда ечилмай қолган калит тарқатиш ва электрон рақамли имзо масалаларининг ечимини излаш йўлларида У. Диффи ва М. Хеллман кўпгина таклифларни илгари сурганлар.

Ошкора калитли криптография асосида ривожланган мамлакатлар орасида биринчи бўлиб АҚШ электрон рақамли имзо бўйича миллий стандарт яратишга киришган. Аввалари Миллий Хавфсизлик Агентлигида ишлаган Девид Кравиц ДСА патенти эгаси ҳисобланади. 1993 йил июнда технологиялар ва стандартлар миллий институти (НИСТ) ДСА учун патент

лицензиясини беришни таклиф этган. Аслида АҚШ стандарти ДСАда 1985 йилда Тохир Ел Гамал томонидан ишлаб чиқилган алгоритм хусусиятларидан ва К. Шнорр ғояси асосида имзо узунлигини қисқартиришга қаратилган иккинчи туб модулдан фойдаланилган. ДСАнинг криптобардошлилиги чекли майдонларда бутун сонларни логарифмлаш муаммоси математикада амалий ҳисоблаш нуқтаи назаридан ҳануз ечилмаганлигига асосланади.

АҚШдан кейин Европа давлатлари ва Японияда электрон рақамли имзо бўйича қонун ва дастлабки давлат стандартлари қабул этилди. Бошқа ошкора калитли криптографияга асосланган воситалар яратилди, экспортга мўлжалланган ахборот-коммуникация тизимларида жорий этилди. Кўпчилик давлатлар, шу жумладан Ҳамдўстлик давлатлари ҳам ошкора калитли криптография воситаларини яратишда АҚШга эргашдилар. Бу ошкора калитли криптографиянинг дастлаб АҚШда юзага келганлиги билан боғлиқ албатта. Улар ахборот–телекоммуникация тармоқларида махфий ахборотларни хавфсиз узатиш ва электрон рақамли имзо яратишда ўз миллий алгоритмларидан фойдаланмоқдалар.

1.4 Қисм бўйича хулоса

Бу бўлимда Ахборот хавсизлигида ҳимоялаш усуллари ҳақида умумий маълумотлар ҳамда классик шифрлар, криптографиядаги асосий тушунчалар ҳамда носимметрик криптографик тизимлар яратиш тамойили ҳақида атрофлича тўхталиб ўтдим.

2.1 RSA очиқ калитли шифрлаш алгоритми

RSA очиқ калитли шифрлаш алгоритми берилган етарли катта тоқ сонни туб кўпайтувчиларга ажратишнинг рационал усули мавжуд эмаслигига асосланган.

Махфий тутиладиган ҳамда етарли катта бўлган p ва q -туб сонлари олиниб, $n = pq$ -сони ва Эйлер функциясининг қиймати $\varphi(n)=(p-1)(q-1)$ ҳисобланади. Бу $\varphi(n)$ -сон очиқ ва махфий калитларни генерация қилиш қоидасининг махфий тутиладиган параметри ҳисобланади. Сўнгра, $(e_i, \varphi(n)) = 1$ шартни қаноатлантирувчи, яъни $\varphi(n)$ сони билан ўзаро туб бўлган e_i -сон бўйича d_i-сони ушбу $e_i d_i = 1 \bmod \varphi(n)$ формула орқали Евклид алгоритми бўйича ҳисобланади. Бу (e_i; d_i) жуфтликда e_i -очиқ калит ва d_i-махфий калит деб эълон қилинади. Шундай қилиб RSA криптотизими фойдаланувчисининг очиқ калити *(n, e)* бўлса, шахсий калити *(d_i, $\varphi(n)$)* жуфтлигидир.

RSA криптотизимида i- фойдаланувчидан j- фойдаланувчига шифрланган маълумотни жўнатиш қуйидагича амалга оширилади:

1. **Шифрлаш қоидаси**: ушбу ифода $M^{e_j} \bmod n = C$ ҳисобланади, бу ерда M -очиқ маълумот, C –шифрланган маълумот;

2. **Дешифрлаш қоидаси**: ушбу ифода $C^{d_j} \bmod n = M^{e_j d_j} \bmod n = M$ ҳисобланиб, очиқ маълумот M ҳосил қилинади.

Дешифрлаш қоидасидаги $C^{d_j} \bmod n = M^{e_j d_j} \bmod n = M$ муносабатнинг ўринлилиги қуйидаги теоремалардан келиб чиқади.

5.1-теорема. Агар $n = pq$, $p \neq q$- туб сонлар ва $(x, p) = 1, (x, q) = 1$ бўлса, у ҳолда

$$x^{\varphi(n)} = 1 \bmod n .$$

Исботи. Агар $(x, p) = 1, (x, q) = 1$ муносабатлар ўринли бўлса, у ҳолда

$$x^{p-1} = 1 \bmod p$$

$$x^{q-1} = 1 \bmod q ,$$

бўлиб, $y = x^{\varphi(n)} = x^{(p-1)(q-1)}$ модуль p бўйча ҳам, модуль q бўйича ҳам 1 га тенг бўлади. Ҳақиқатан ҳам:

$$y = x^{\varphi(n)} \bmod p = x^{(p-1)(q-1)} \bmod p = [x^{(p-1)} \bmod n]^{(q-1)} \bmod n = 1^{(q-1)} \bmod n = 1$$

ёки

$$y = x^{\varphi(n)} \bmod p = x^{(p-1)(q-1)} \bmod p = [x^{(q-1)} \bmod n]^{(p-1)} \bmod n = 1^{(p-1)} \bmod n = 1.$$

Бундан эса, $(y - 1)$ нинг p ва q сонларига қолдиқсиз бўлиниши келиб чиқади ҳамда y=1 mod pq тенглик ўринли бўлади.

5.2-теорема. Агар $n = pq$, $p \neq q -$ туб сонлар ва $(e, \varphi(n)) = 1$ бўлса, у ҳолда ушбу

$$E_{e,n} : x \to x^e \bmod n$$

акслантириш $\mathbf{z}_n = \{0; 1; 2; \ldots; n-1\}$-чекли майдонда ўзаро бир қийматли акслантириш бўлади.

Исботи. Агар $(e, \varphi(n)) = 1$ бўлса, у ҳолда шундай d - ҳақиқий сон мавжуд бўладики, унинг учун

$$ed = 1 \bmod \varphi(n),$$

муносабат ўринли бўлади. Бундан эса ушбу муносабат

$$(x^e)^d = x^{ed} = x^{1+K\varphi(n)} = x (\bmod n)$$

EKUB$(x, n) = 1$ ифодани қаноатлантирувчи барча x лар учун бажарилади.

Агар $x = py$ бўлса, бу ерда $(y, q) = 1$, у ҳолда

$$p \mid x^{1+K\varphi(n)} - x.$$

Бу ерда x сони q га қолдиқсиз бўлинмаганлигидан

$$x^{1+K\varphi(n)} - x = x\left[(x^{q-1})^{K(p-1)} - 1\right]$$

келиб чиқади.

Ферманинг кичик теоремасига кўра $x^{q-1} = 1 \bmod q$ ва натижада, квадрат қавс ичидаги ифода модуль p бўйича ҳам ва модуль q бўйича ҳам 0 га тенг бўлиб, бундан ушбу

$$x^{1+K\varphi(n)} - x = 0 \bmod n$$

тенгликнинг ўринлилиги келиб чиқади.

Худди шу каби, агар $x = qy$ бўлса, бу ерда $(y, p) = 1$, у ҳолда

$$q \mid x^{1+K\varphi(n)} - x .$$

Бу ерда x сони q га қолдиқсиз бўлинмаганлигидан

$$x^{1+K\varphi(n)} - x = x\left[(x^{p-1})^{K(q-1)} - 1\right]$$

келиб чиқади.

Ферманинг кичик теоремасига кўра $x^{p-1} = 1 \bmod p$ ва натижада, квадрат қавс ичидаги ифода модуль p бўйича ҳам ва модуль q бўйича ҳам 0 га тенг бўлиб, бундан ушбу

$$x^{1+K\varphi(n)} - x = 0 \bmod n$$

тенгликнинг ўринлилиги келиб чиқади.

Шундай қилиб, келтирилган теоремаларга кўра

$$C^{d_j} \bmod n = M^{e_j d_j} \bmod n = M^{K\varphi(n)+1} \bmod n = [(M^{\varphi(n)})^K \bmod n \cdot M \bmod n] \bmod n =$$
$$= [1^K \bmod n \cdot M \bmod n] \bmod n = M \bmod n = M$$

чунки, $M < n$.

Очиқ ва махфий калитларнинг генерацияси чоғида $e_i d_i = 1 \bmod \varphi(n)$ тенгликни қаноатлантирувчи d_i- сонини $\varphi(n)$- сони маълум бўлганда Евклид алгоритми бўйича топилади. Аммо $\varphi(n)$- сони фойдаланувчиларга номаълум бўлганда d_i- сонидан ташқари $\varphi(n)$-сони ҳам махфий бўлиб, $\varphi(n)$ - сонини аниқлаш учун n–сонини туб кўпайтувчиларга ажратиб, p ва q сонларини топиш талаб этилиб, сўнгра $\varphi(n) = (p-1)(q-1)$ ҳисобланади. n–сони етарли катта бўлганда уни туб кўпайтувчиларга ажратиб, p ва q сонларини топишнинг рационал усули бугунги кунда мавжуд эмас. Адабиётлар рўйхатида келтирилган [60] да етарли катта натурал сонларни экспоненциал ва субэкспоненциал мураккабликларга ажратиб, уларни туб кўпайтувчиларга ажратишнинг баъзи усуллари келтирилган.

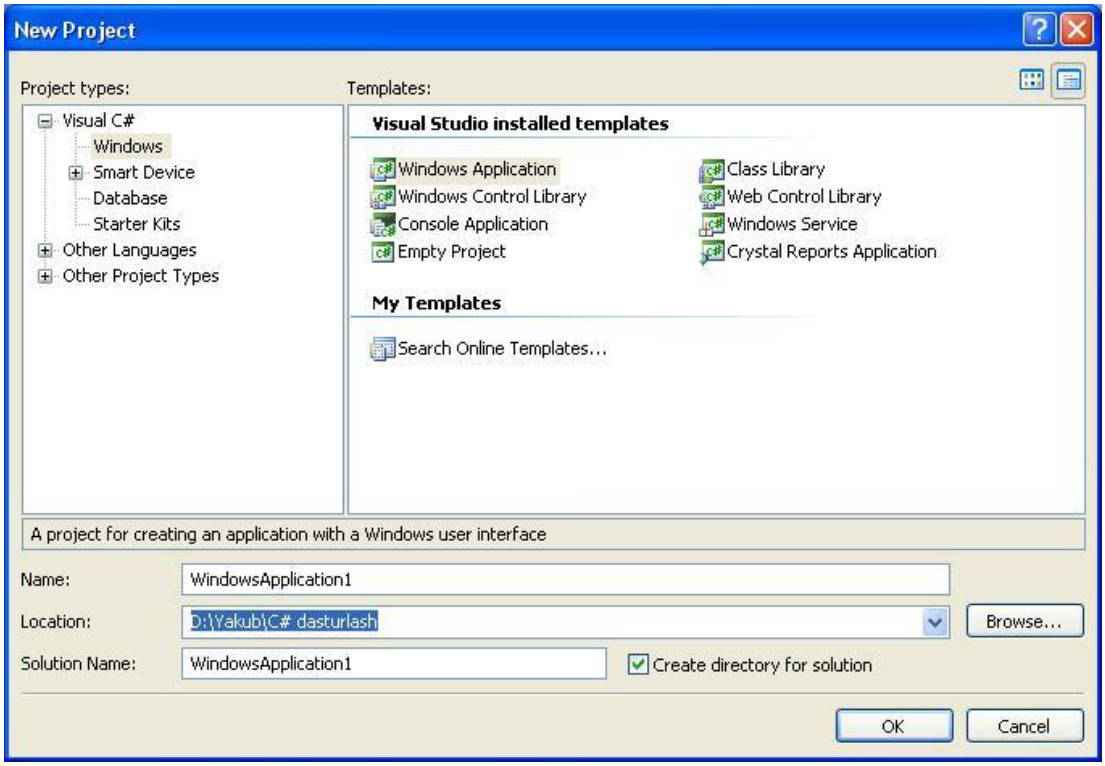
2.2 RSA шифрлаш алгоритми учун дастурий таъминот яратишда C# дастурлаш тилининг аҳамияти

Айтиш мумкинки, NET платформанинг ҳаёт йўли бозорда Visual Studio.NET нинг янги авлод воситалари комплекси пайдо бўлиши билан бошланди. Унинг расмий тақдимоти ва 1- сотилиши 2002 йилнинг феврал ойида бўлиб ўтган. Бета версиясини эса ундан ҳам олдинроқ олиш мумкин бўлган. Visual Studio.NET нинг барча Microsoft.NET ғоялари учун муҳимлиги қуйидагича: платформанинг ривожланиши тўғридан-тўғри NET технология билан ишлаш имконини берадиган амалий дастурларнинг бўлишига боғлиқ. 2000 йилнинг охирида бета версиянинг яратилиши билан Microsoft нинг ўз янги платформасидан мақсадлари аниқлашди. Янги платформанинг афзалликлари ва камчиликлари ҳақида баҳслар қизиб кетди ва унинг кўплаб рақиблари пайдо бўла бошлади. Аммо кўпчилик унинг расмий версияси чиқишини кутаётган эди. 2001 йил иккинчи бета версия чиқиши билан мунозаралар анча камайди. Айтиб ўтиш лозимки, гап фақат фойдаланилаётган махсулотнинг версияси янгиланганлиги ҳақида эмас, балки технологиянинг тубдан ислоҳ қилиниши ҳақида кетмоқда.

Бошлаш саҳифаси

Дастлаб Visual Studio.NET ни ишга туширамиз. Бунинг учун дастур ўрнатилгандан сўнг Пуск/Дастурлар/Microsoft Visual Studio.NET/ Microsoft Visual Studio.NET ни босинг. экранда Visual Studio Home Page бошлаш саҳифаси пайдо бўлади. Ўнг томонда сўнгги бажарилган ишлар рўйхати, шунингдек, бошқа тугмалар жойлашган. Финд Самплес закладкаси калит сўзлари ёрдамида қидириш имконини беради. Чап томонда қўшимча буйруқлар рўйхати жойлашган. Му ProFile ёрдамида эса муҳит кўринишини ҳохишингизга қараб созлашингиз мумкин. Вҳатъс New ҳаволаси эса Visual Studio.NET янгиликлари билан таништиради. Бошқа интернет ресурсларига боғлиқ ҳаволарни эса мустақил ишлатиш имкони йўқ. Visual Studio.NET – бу фақатгина C# да дастурлар яратиш муҳити эмас. Visual Studio.NET Visual Basic, C#, C++ каби тилларда дастурлар яратиш,

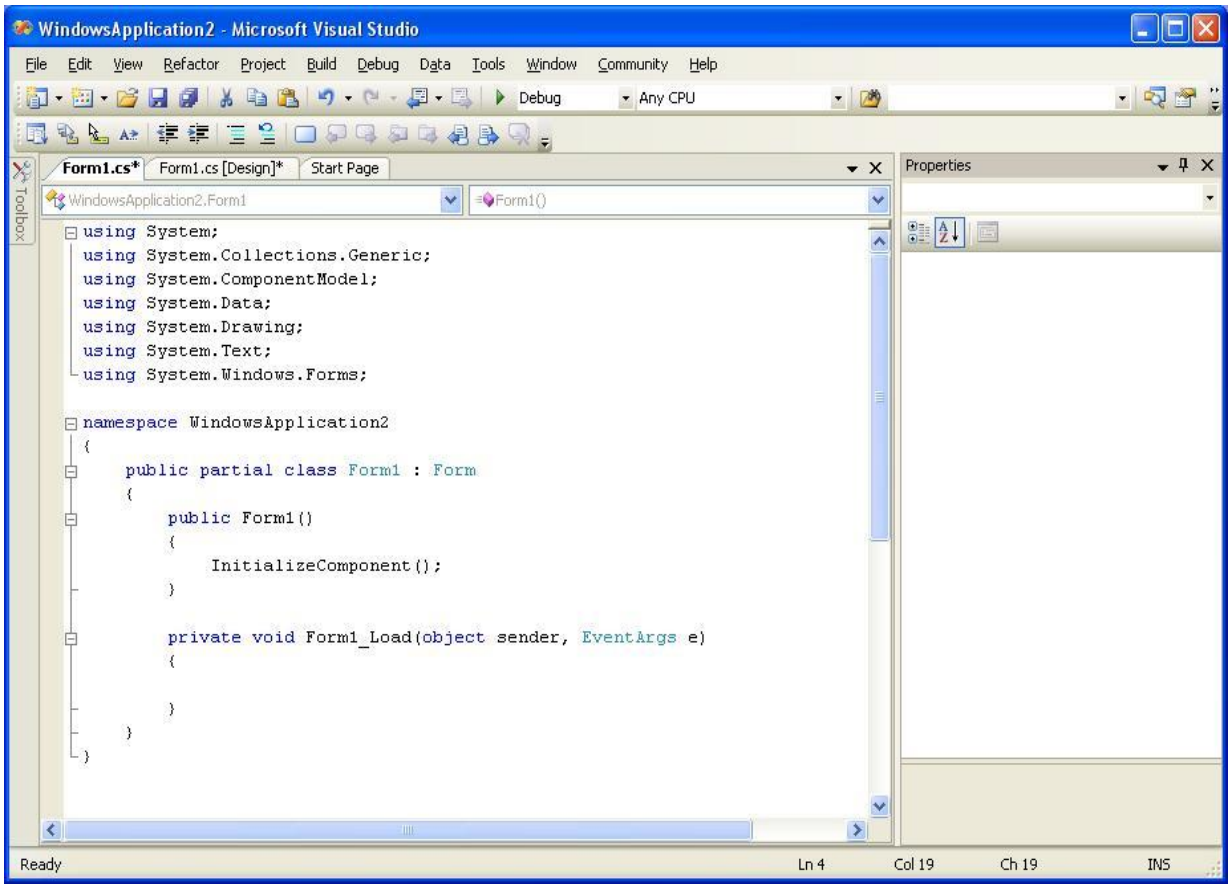
Setup (ўрнатиш пакети) хосил қилиш ва х.к. лар имконини беради. Visual Studio.NET да проект қандай яратилишини кўриш учун менюдан File/New/Project… Шундан сўнг экранда қуйидаги 1-расмда тасвирланган ойна чиқади:



1-расм. Янги проект яратиш ойнаси.

Бу ерда керакли тил танланади, ушбу битирув малакавий иши C# да ёзилганлиги учун C#ни танлаб кўрамиз. Ойнанинг ўнг томонида яратилаётган проектнинг типи кўрсатилиши лозим. Улар қуйидагилар: (Windows Application), (ASP.NET), (Console Application) ва баъзи бошқалар. Шунингдек, проектга ном ва жойлашадиган каталокка йўл кўрсатиш мумкин. ОК тугмасини босганингиздан сўнг 2- расмда тасвирланган ойна пайдо

бўлади:

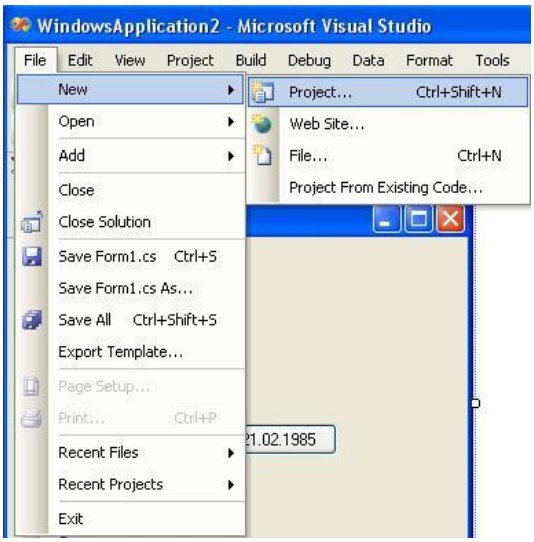


2- расм. Проект яратишнинг асосий висуал қисмлари.

Марказда визуал формалар яратиш ва кодлар ёзиш ойнаси жойлашган. Ўнг томонда эса Solution Explorer, Class View ва Properties Explorer ойналари жойлашиши мумкин.

Visual Studio.NET нинг асосий менюси

Visual Studio.NET нинг менюси ойнанинг юқори қисмида жойлашган. Менюда Visual Studio.NET проект элементлари устида ишлашга мўлжалланган барча буйруқлар мавжуд. Меню пунктлари буйруқли ва гуруҳли (бошқа пунктларга эга) бўлади. Менюдаги ҳар бир гуруҳли пунктининг номи унинг таркибидагиларга боғлиқ. Масалан, File менюси проект файллари билан ишлашга мўлжалланган буйруқларни ўз ичига олади. Бу 3- расмда тасвирланган:



3-расм. Асосий меню.

1-Жадвал. File менюси

Буйруқ номи	**Тугмалар**	**Буйруқнинг вазифаси**
New > Project	**Ctrl+Shift+N**	Янги проект яратиш
File	**Ctrl+N**	Янги файл яратиш
Blank Solution	-	Янги қарор яратиш
Open> Project	-	Олдин яратилган проектни очиш
Project For Web...	**Ctrl+Shift+O**	Олдин яратилган проектни тармоқ бўйича очиш
File	**Ctrl+O**	Алоҳида файлларни очиш
File For Web...		Алоҳида файлларни очиш(проектга тегишли ва у билан тармоқ орқали боғланмаган)
Close	-	Жорий проектни ёпиш
Add New Item..	**Ctrl+Shift+A**	Проектга янги элемент қўшиш
Add Existing Item...	**Alt+Shift+A**	Проектга мавжуд элементни қўшиш
Open Solution...		Йечимнинг мавжуд проектини очиш
Close Solution		Йечимнинг мавжуд проектини ёпиш
Save Selected Items...	**Ctrl+S**	Фаол саҳифани сақлаш

4-Жадвал. Debug менюси

Буйруқ номи	**Тугмалар**	**Буйруқнинг вазифаси**
Windows\Breakpoints	**Ctrl+Alt+B**	Параметрлар ойнасини очиш
Start	**F5**	Проектни ишга тушуриш
Restart	**Ctrl+F5**	Дастурни бошидан ишга тушуриш
Break All	**Ctrl+Shift+Break**	Дастур бажарилишини вақтинча тўхтатиш
Continue	**F5**	Дастур бажарилишини давом эттириш
Stop Debugging	**Shift+F5**	Дастур бажарилишини тўхтатиш
Detach All	**Shift+F5**	Проектга бириктирилган барча жараёнларни бекор қилиш
Start Without Debugging	**Ctrl+F5**	Дастурни Debug сиз ишга тушуриш
Processes	**Ctrl+F5**	Барча жараёнлар рўйхати
Step Into	**F11**	Дастурнинг қадамли бажарилиши
Step Over	**F10**	Дастурнинг курсоргача бажарилиши

Оператор ва буйруқлари

C# (Си шарп деб ўқилади) – бу Microsoft компаниясининг янги дастурлаш тилидир. У Висуал Студио ва Visual Studio.NETнинг янги версиясига киради. C# дан ташқари Visual Studio.NET- га Visual Basic.NET ва C++ ҳам киради. Бундан ташқари Борландфирмаси шуни маълум қилдики, C++ Builder ва Delphiнинг кейинги версиялари .NET платформани қўллаб қувватлайди. Microsoft компаниясининг янги тил яратишига сабаблардан бири – бу янги .NET платформа учун компонентга йўналтирилган тил яратиш эди. Бошқа тиллар .NET платформа яратилишидан олдин яратилган эди. C# тили эса шу платформа асосида яратилган. Лекин бу тил янги платформа учун ягона дегани эмас. Microsoft компаниясининг янги тил яратишига сабаблардан яна бири – бу

Java тили учун алтернатива яратиш эди. Бизга маълумки, Javani Microsoft да реализация қилишлицензияланмаган эди. Сун компанияси (Java нинг эгаси) Microsoft ни судга беради, ва бу судда Microsoft ютқазади. Шундан сўнг Microsoft Java дан умуман воз кечишга ва Java га ўхшаш тил яратишга қарор қилди. Ўша тил C# номини олди. C# чиққандан сўнг Javaга нима бўлади – бу ҳали номаълум. C# дан мақсад Java билан рақиблик қилиш бўлса ҳам бу тиллар ҳозирда ишлатилмоқда.

Ўзгарувчилар

C# нинг ҳар бир маълумот типи учун SRL да (Common Language Runtime) мос маълумот типлари мавжуд. Бу шундан далолат беради, ҳар бир тип икки номга эга – тўлиқ ва қисқартирилган. Қисқартирилган ном C# да қўлланилади. Қанақа ном қўллаш дастурчига боғлиқ. Қисқартирилган ном қисқа ва қулайдир. Асосий маълумотлар типлари қуйидаги жадвалда берилган:

5-Жадвал. C# тили ўзгарувчилари

C# типи	**SRL типи**	**Байтлардаги ҳажми**	**Изоҳ**
int	Int32	4	Бутун (белги билан)
float	Single	4	Ҳақиқий сон
char	Char	-	Символ (Unicode)
bool	Boolean		Мантиқий тип
short	Int16	2	Қисқа бутун (белги билан)
long	Int64	8	Узун бутун (белги билан)
String	String		Қатор
Byte	Byte	1	Байт
decimal	Decimal	8	Аниқ ҳақиқий сон

Агар сиз узун ном ишлатмоқчи бўлсангиз, у ҳолда System.Int32 каби ёзишингиз керак ва х.к. (Яъни SLR ном олдидан System сўзини ёзиш керак). Ёки дастур бошида қатор қўшса ҳам бўлади using System; Шундай қилиб к ўзгарувчисининг 3 хил эълон қилиниши бир хил кучга эга:

int к;

using System;

int32 k;

Ва System.Int32 k;

Ўзгарувчини эълон қилиш инициализация (бошланғич қиймат бериш) билан бирга олиб борилиши мумкин:

int z=88;

C# операторлар тўплами стандартдир: +, -, *, / - бошқа ихтиёрий тилларда ҳам ишлатилади. Шуни таъкидлаш керакки, / (бўлиш белгиси) бутун сонларга қўлланганда бўлинманинг бутун қисмини беради:

```
int k=100999, n=1000, c;
  c=k/n;
  Console.WriteLine(c.ToString());
```

Ушбу фрагмент экранга 101 эмас 100ни чиқаради, ҳеч қандай айланиш содир бўлмайди.

Яна бир % оператори борки, у бўлинманинг қолдиғини беради. Қуйидаги фрагмент экранга 999 ни чиқаради:

```
int k=100999, n=1000, c;
  c=k%n;
  Console.WriteLine(c.ToString());
```

С га ўхшаш тиллардаги каби C# да ҳам инкримент ва декримент операторлари мавжуд.

Қуйидаги фрагментда k 1 га ортади, n эса 1га камаяди:

```
k++;
n--;
```

Мантиқий операторлар

С га ўхшаш тиллардаги каби C# да ҳам қуйидаги мантиқий операторлар мавжуд:

6- Жадвал. C# тили ўзгарувчилари

Оператор	**Таъриф**	**Мисол**
&&	Мантиқий **ВА.** Агар иккала операнда ҳам	(x==8) &&

	TRUE бўлса, натижа TRUE бўлади	(y==5)
||	Мантиқий **ЁКИ**. Агар иккала операнда ҳам FALSE бўлса, натижа FALSE бўлади	(y>8) || (y<5)
!	Инкор.Мантиқий қийматни тескарисига алмаштиради	if(!(a==b))...

Бу операторларнинг ҳаммаси bool типли натижага эга. Эътибор беринг юқорида 2та баробар белгиси (==) ишлатилмоқда. 1та баробар белгиси (=) ўзлаштириш учун ишлатилади. 2та баробар белгисига жуфт белги ҳам бор, бу - !=(“тенг эмас”). Шундай қилиб юқоридаги **!** оператори учун мисолни қуйидагича ўзгартириш мумкин:

IF (!(a==б))**...**

Таъкидлаш жоизки C# да бошқа дастурлаш тилларидан фарқли ҳолда FALSE

ўрнига 0 ёки TRUE ўрнига ихтиёрий 0 дан бошқа сон қўйиш мумкин эмас. Шунга кўра қуйидаги фрагмент хато ҳисобланади:

```
Int k;
...
if (k) //Хато!
...
```

C# да класслар.

Аввало, класслар ўзи нима эканлиги ҳақида икки оғиз. Тасаввур қилинг сизда бирор бир ҳусусият ёнида характерланадиган бирор объектингиз бор. Масалан, бирор фирма ишчисини олайлик. Унинг фамилияси, ёши, стажи ва х.к. хусусиятлари бор. Бу ерда ҳар бир ишчини мустақил ўзгарувчилар ёнида эмас, балки Worker типли битта ўзгарувчи ичида таърифлаш қулайроқ. Шу Worker ичида фамилия, ёш, стаж ўзгарувчилари жойлашади. Worker типи C# маълумот типлари орасида йўқ, аммо бу муаммо

Мисол:

```
int[] k = {-5, 4, 55};
```

Берилган конструксия ларни нафақат int типига балки 3 ўлчовли массив учун ҳам қўлласа бўлади. С ва С++ даги каби С# да ҳам массив элементлари нумерацияси 0 дан бошланади. Шунга кўра бизнинг мисолда массивнинг бошланғич элементи – бу к [0], охиргиси эса – к [2]. к [3] элемент эса йўқ. Энди кўп ўлчамли массивларга ўтамиз. Икки ўлчовли массив қуйидагича берилади:

int[,] к = new int [2,3];

Эътибор беринг, квадрат қавс фақат 1 марта ишлатилган. Мисолимиздаги 6 (=2*3)массивда ҳам (к[0,0] - биринчи, к[1,2] - иккинчи) Шу сингари кўп ўлчовли массивларни ҳам киритишимиз мумкин. Қуйида уч ўлчовли массивга мисол берилган:

int[,,] к = new int [10,10,10];

Қуйидаги йўл билан кўп ўлчовли массивларни бирданига инициализация қилиш мумкин:

int[,] к = {{2,-2},{3,-22},{0,4}};

Юқорида келтирилган массивларга мисоллар тўғрибурчакли деб аталади. Агар уларни жадвал кўринишида берсак, у ҳолда массив тўғрибурчак ҳосил қилади. Тўғрибурчакли массивлар билан бир қаторда поғонали массивлар ҳам мавжуд. Мисол:

```
//2 ўлчовли поғонали массивни аниқлаймиз
  int[][] к = new int [2][];
//поғонали массивнинг 0-елементини аниқлаймиз
  //Бу массивда 3 та элемент мавжуд
  к[0]=new int[3];
// поғонали массивнинг 1-елементини аниқлаймиз
  // Бу массивда 4 та элемент мавжуд
  к[1]=new int[4];
```

к[1][3]=22; //Массивнинг охирги элементига 22 ни ёзамиз

Эътибор беринг,поғонали массивларда бир неча жуфт квадрат қавслар ишлатилган(массивларнинг ўлчамига қараб). Шунингдек массивэлементларини ўқишимиз, ёзишимиз мумкин ва х.к. Поғонали массивларнинг энг муҳим ва қизиқ имконияти – бу уларнинг "тўғрибурчакли" эмаслигидир. Юқорида берилган мисолдаги к массивнинг биринчи қаторида 3 та бутун сон, иккинчисида эса 4 та мавжуд.

If ва switch операторлари

If оператори дастурнинг икки йўналишда шоҳланиб кетиши учун хизмат қилади. Агар бирор шарт қўлланса дастур бир томонга кетади, агар қўлланмаса бошқа томонга кетади. Қуйида фойдаланувчи тоқ ёки жуфт сон киритилганлигини аниқлайдиган мисол келтирамиз:

```
classClass1
{
...
    static void Main(string[] args)
    {
        int к = Int32.Parse(Console.ReadLine());
        if(b)
        {
            Console.WriteLine("Жуфт сон");
        }
        else
        {
            Console.WriteLine("Тоқ сон");
        }
        Console.ReadLine();
    }
}
```

For ва foreach цикллари

For циклига мисол билан бошлаймиз:

```
int к = Int32.Parse(Console.ReadLine());
int sum=0;
for(int i=1; i<=к; i++){
   sum+=i;
}
Console.WriteLine(sum);
```

Бу мисол 1 дан фойдаланувчи киритган к гача бўлган сонлар суммасини ҳисоб-лайди. Сумма SUM ўзгарувчисига ёзилади ва экранга чиқарилади. Кўп ҳолларда цикллар массивлар устида бир неча амаллар бажариш учун ишлатилади. Массив элементлари нумерацияси 0 дан бошлангани учун, типик цикл қуйидагича кўринишга эга бўлади:

```
int[] a = {-5, 4, 55};
int sum=0;
for(int i=0; i<3; i++){
   sum+=a[i];
}
```

Бу мисолда цикл счетчигининг бошлъанғич қиймати 0 га тенг, ва цикл давомидаги шартга кичик белгиси қўямиз (<), сўнг массив элементлари сони қўйилади. Агар циклда фақат битта оператор бажарилса, у ҳолда фигурали қавс қўйиш шарт бўлмайди. Энди foreach циклига мисол кўрамиз:

```
int[] m = {-5, 4, 10};
int sum=0;
foreach(int i in m){
```

While цикли

While цикли икки – while ва do-while кўринишда бўлади. Иккала цикл ҳам, қоидага кўра, цикл неча марта бажарилиши номаълум бўлган ҳолларда ишлатилади. Масалан, фойдаланувчи томонидан парол

киритилаётган ёки бирон нимани чекли аниқликда ҳисоблаш пайтида. Иккала цикл ҳам while сўзидан кейинги қавс ичидаги шартTRUE бўлгунча бажарилади. Шарт FALSE га тенг бўлиши билан цикл бажарилиши тўхтайди. While ва do-while ўртасидаги энг муҳим фарқ шики, do-while кўпи билан бир марта бажарилган ҳолда while бирор марта ҳам бажарилмаслиги мумкин. Уларни қўллашга мисоллар:

```
string password;
do{
   password=Console.ReadLine();
}while(password!="Саттаров");
int к=0; //қадамлар сони
//Ихти рий сонларнинг нги кетма-кетлигини киритамиз
Random rnd=new Random(112); // Ихти рий параметр
while(rnd.Next(1, 6)!=5)
{
   k++;
};
Console.WriteLine("С "+(k+1).ToString()+"-марта 5 тушди");
```

Биринчи мисолда цикл фойдаланувчи тўғри паролни (саттаров) киритмагунча айла-нади,иккинчи мисолда эса қандайдир ихтиёрий сон 5 га тенг бўлгунча такрорла-нади. Агар сон бошиданоқ 5 чиқса цикл бирор марта ҳам бажарилмайди.

Сатрлар (System.String класси)

C# да сатрлар бу – SystemString классининг нусхаларидир. Умуман олганда, C# да string типи бор, лекин SystemString кўпроқ ривожланган, шунинг учун уни ишлатиш анча осон ва қулай. Бу класс бир қанча метод ва ҳусусиятларга эга, улардан баъзилари қуйида санаб ўтилган:

Length хоссаси сатр узунлигини ифодалайди. Қўллаш учун мисол:

```
  String c="қққ";
```

нусха яратилаверишини истамасак, у ҳолда System.String класси ўрнига

StringBuilder классини қўллашимиз керак.

Константалар

Константалар - дастурда ўзгармайдиган катталикларда ишлатиш учун қулайдир.Константалардан фойдаланиш дастурнинг ихтиёрий ерида баъзи катталикларни бир зунда ўзгартириш имконини беради. Константалар const калит сўзи билан аниқланади.

Мисол:

```
classMyClass
   {
// константани эълон қилиш.
      public const int SomeValue=20;
      ...
   }
```

Константалар класс ичида аниқланади. Кўпинча дастурда баъзи ёрдамчи класс киритилади. Ундан мақсад, дастурнинг барча константаларини бир жойда сақлашдир. Масалан:

```
abstract classConstans
   {
      public const int SomeValue1=20;
      public const int SomeValue2=100;
      public const double SomeValue3=0.35;
   }
```

Бу класснинг нусхасини яратиш имкони йўқлиги учун бу классни абстракт ҳолда аниқладик. У фақат константаларни сақлаш учун ишлатилади. Масалан:

```
classApp
   {
      static void Main()
```

C# да киритиш/чиқариш (System.IO)

Киритиш-чиқариш операциялари учун System.IO номлар фазоси

қўлланилади. Шу номлар фазосининг энг муҳим класслари қуйида санаб ўтилган:

- **BinaryReader** – файлдан турли типдаги маъкумотларни ўқиш имконини беради (бутун, ҳақиқий, мантиқий ва х.к.)

- BinaryWriter - файлга турли типдаги маъкумотларни ёзиш имконини беради (бутун, ҳақиқий, мантиқий ва х.к.)

- Directory– папкалар билан ишлайдиган статик методли класс

- DirectoryInfo – маълум бир папка билан ишлаш учун класс

- File - файллар билан ишлайдиган статик методли класс

- FileInfo - маълум бир файл билан ишлаш учун класс

- Path – файл йўллари билан ишлайдиган класс

- FileAttributes – файл аттрибутлари

- FileMode – файлни очишнинг мавжуд воситалари

- FileAccess - файлнинг ҳусусиятларини кўрсатувчи константаларни ўз ичига олади

- FileSystemWatcher - файл тизимидаги ўзгаришларни кузатиш учун класс

- NotifyFilters - файл тизимидаги ўзгаришларни бажарадиган параметрлар

- WatcherChangetypes - файл тизимида қандай ўзгаришлар кузатилмоқда

System.Security.Cryptography – номлар фазоси

System.Security.Cryptography – номлар фазоси маълумотларни шифрлаш ва дешифрлаш ҳамда бошқа бир нечта ҳешлаш, тасодифий сонлар генерацияси ва ҳабарларни ҳақиқийликка текшириш каби криптографик хизматларни тақдим этади

AsymmetricSignatureFormatter - синфи

AsymmetricAlgorithm - синфи

Ассимметрик шифрлаш алгоритмларини реализация қилишда керак бўладиган абстракт синфлар тўплами.

Меъросийлик ийерархияси

System.Object

System.Security.Cryptography.AsymmetricAlgorithm

System.Security.Cryptography.DSA

System.Security.Cryptography.ECDiffieHellman

System.Security.Cryptography.ECDsa

System.Security.Cryptography.RSA

конструкторлар

Номи	вазифаси
AsymmetricAlgorithm	AsymmetricAlgorithm синфининг янги нусхасини шакллантиради

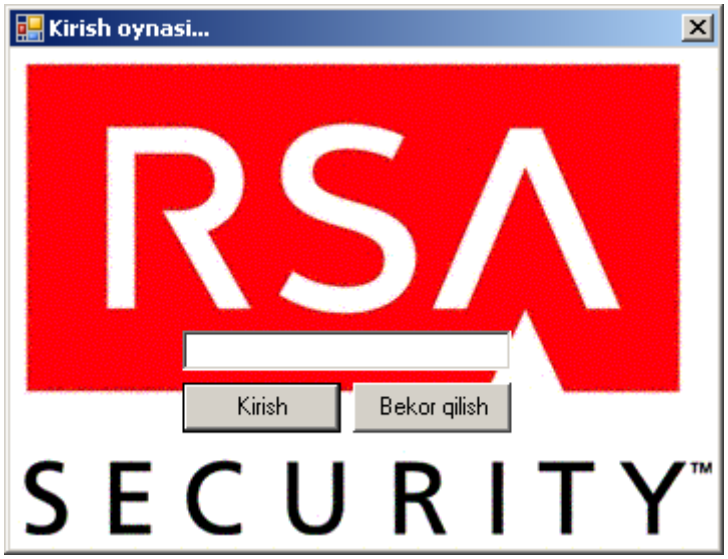
Хоссалари

Номи	вазифаси
KeyExchangeAlgorithm	Жорий синфда Калитлар алмашиш алгоритми номини аниқлайди
KeySize	Жорий ассимметрик шифрлаш алгоритми калитининг модули узунлигини (битларда) қабул қилади
LegalKeySizes	ассимметрик шифрлаш алгоритми учун калит ўлчамини қабул қилади
SignatureAlgorithm	Имзо алгоритми номини қабул қилади

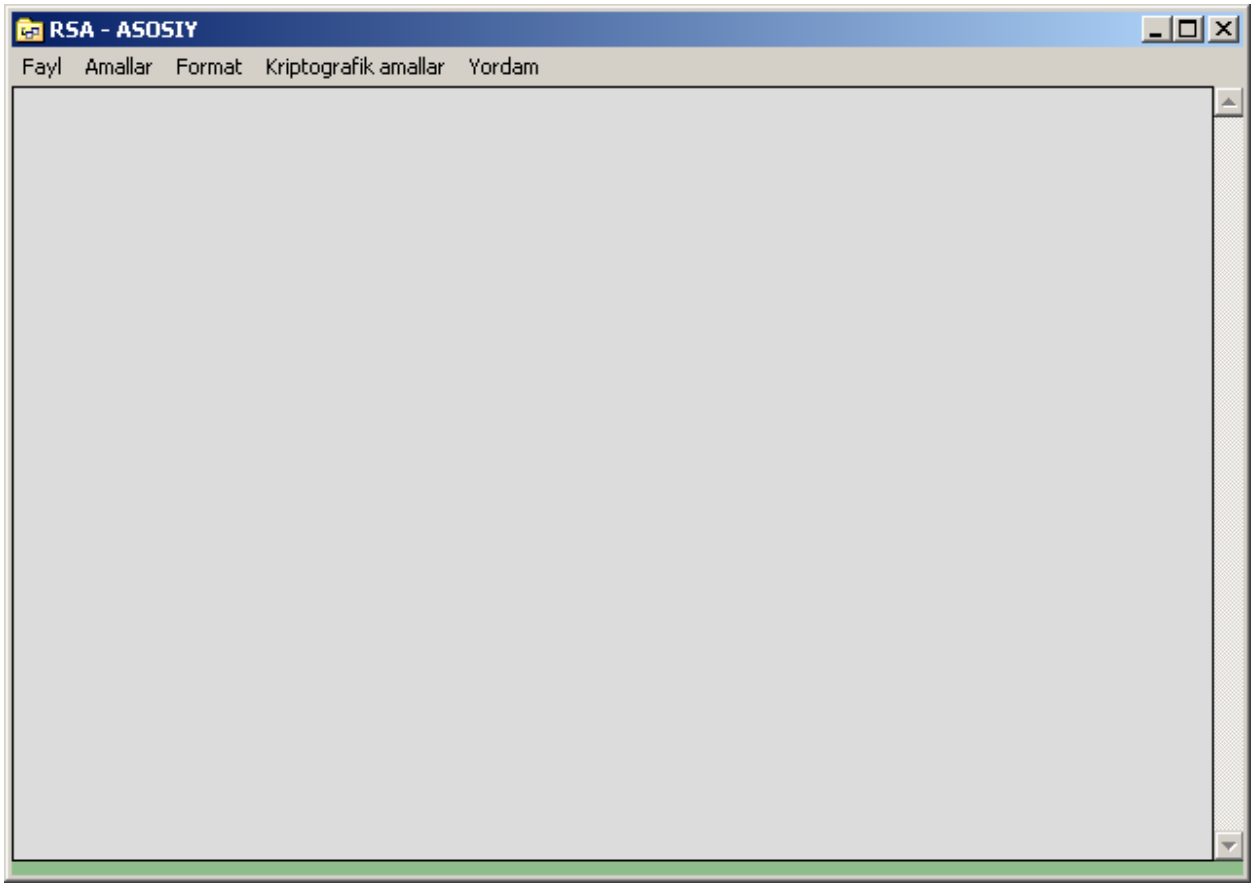
2.3 RSA шифрлаш дастуридан фойдаланиш йўриқномаси

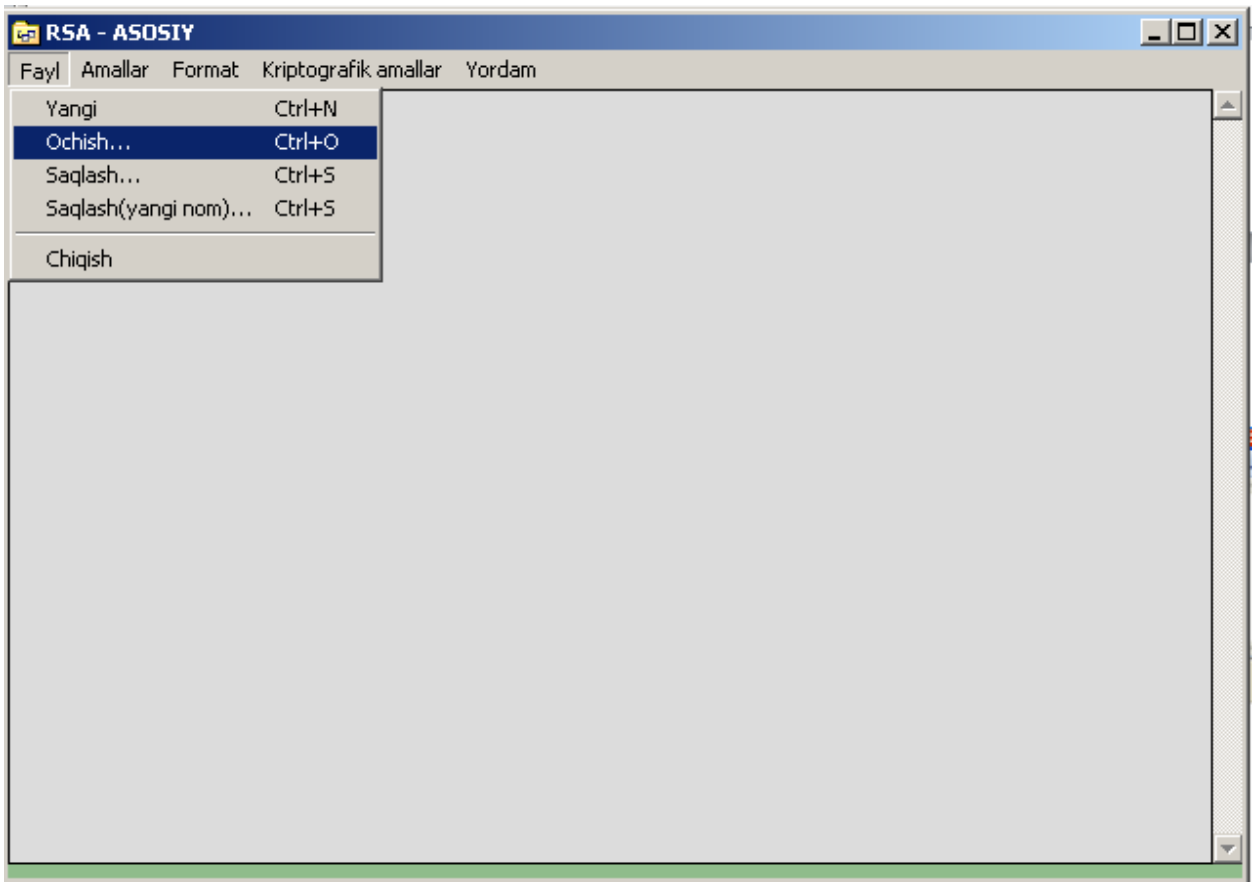
RSA шифрлаш дастури умумий кўриниши қуйидагича. Дастлаб дастурни ишга тушириш учун “RSA.exe” файли ишга туширилади ва бунинг натижасида кириш ойнаси очилади. Хосил бўлган ойнадан дастурга кириш учун паролни киритади ва кириш тугмаси босилади.

Кириш ойнаси



RSA асосий ойнаси

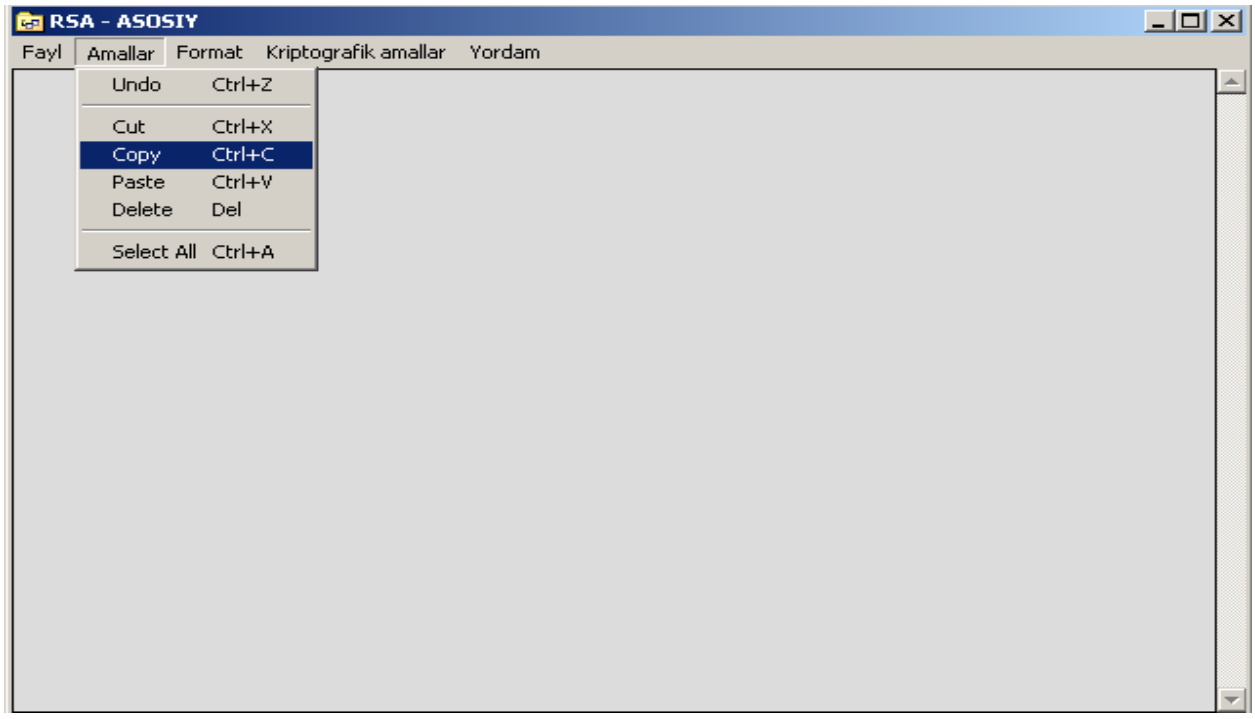




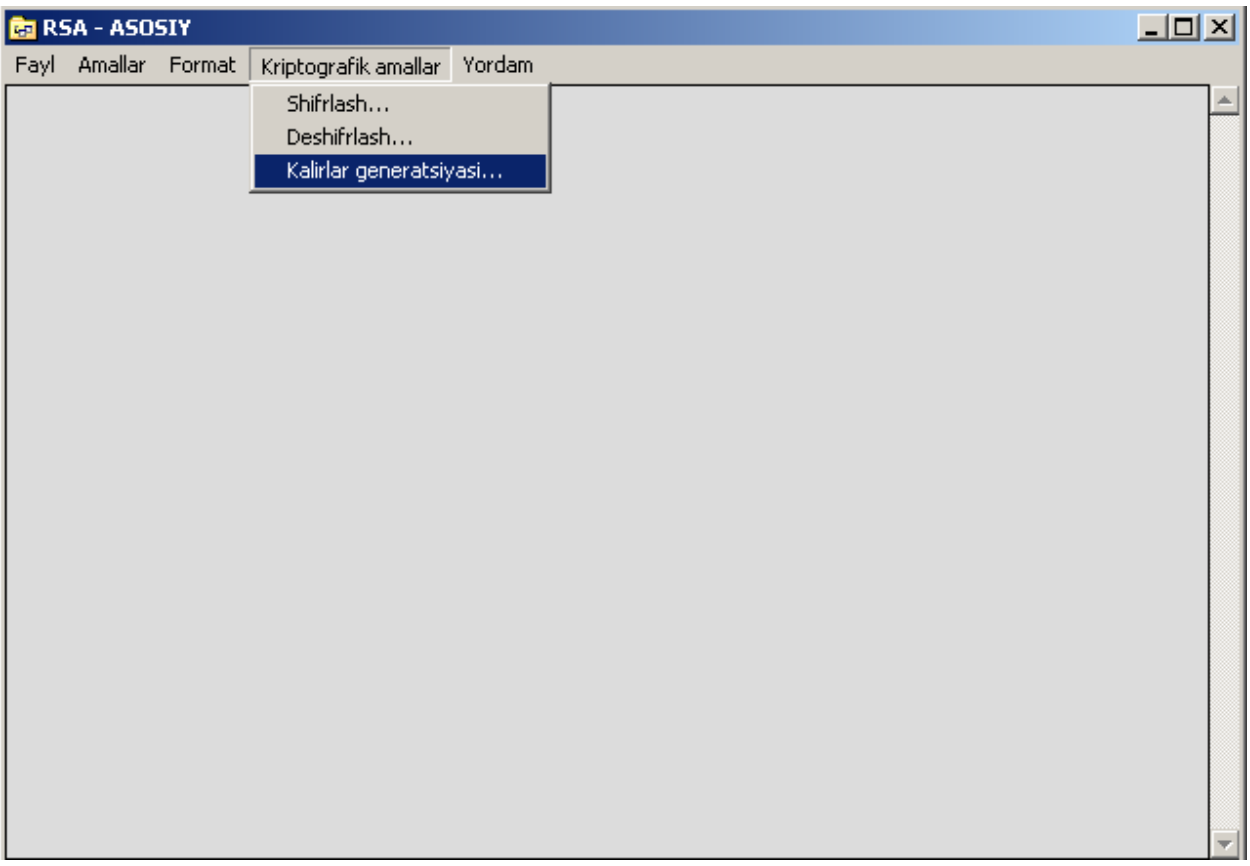
Файл менюси

Файл менюси таркиби блокнот дастури каби содда бўлиб, фойдаланувчи учун қулай

Амаллар менюси

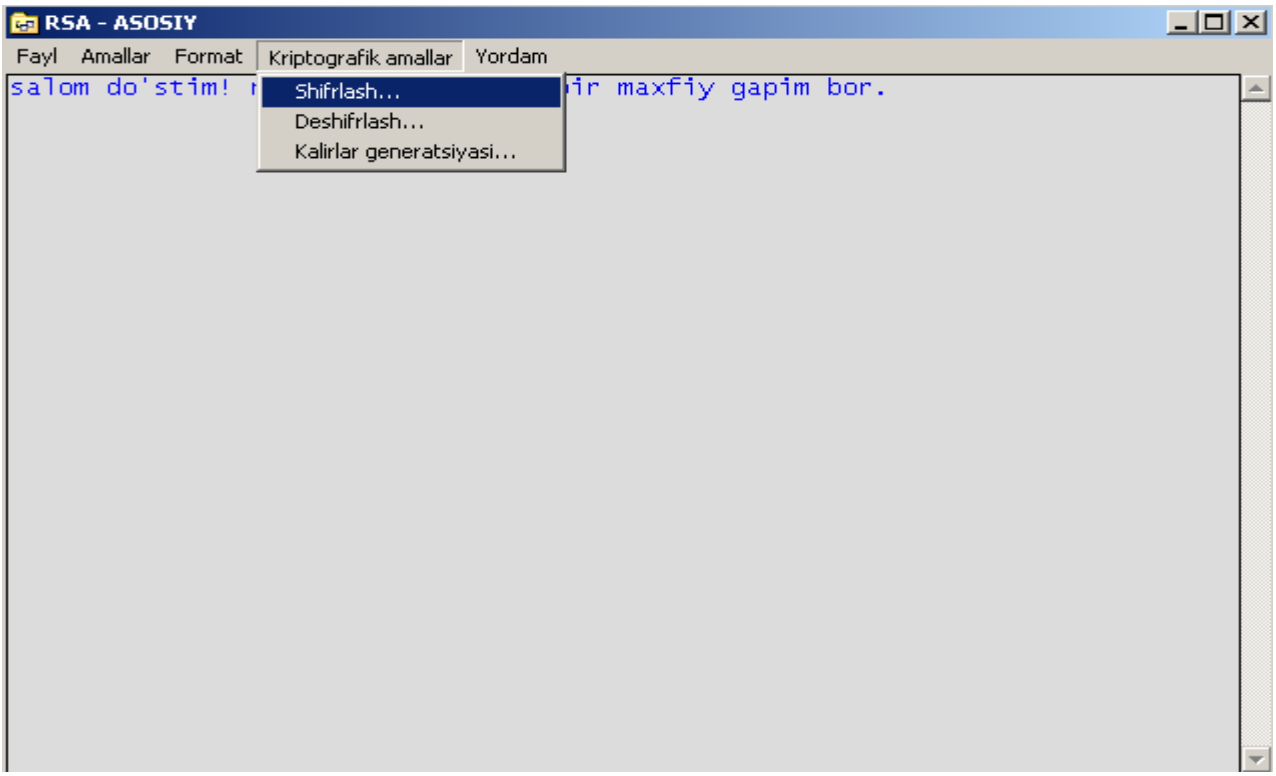


Криптографик амаллар менюси



Криптографик амаллар менюсида жойлашган буйруқлар дастурнинг энг асосий қисми ҳисобланади**.** Калитлар генерацияси буйруғи орқали очиқ ва ёпиқ калитлар генерация қилинади. Бу ойнада калит узунлигини ҳам белгилаш мумкин

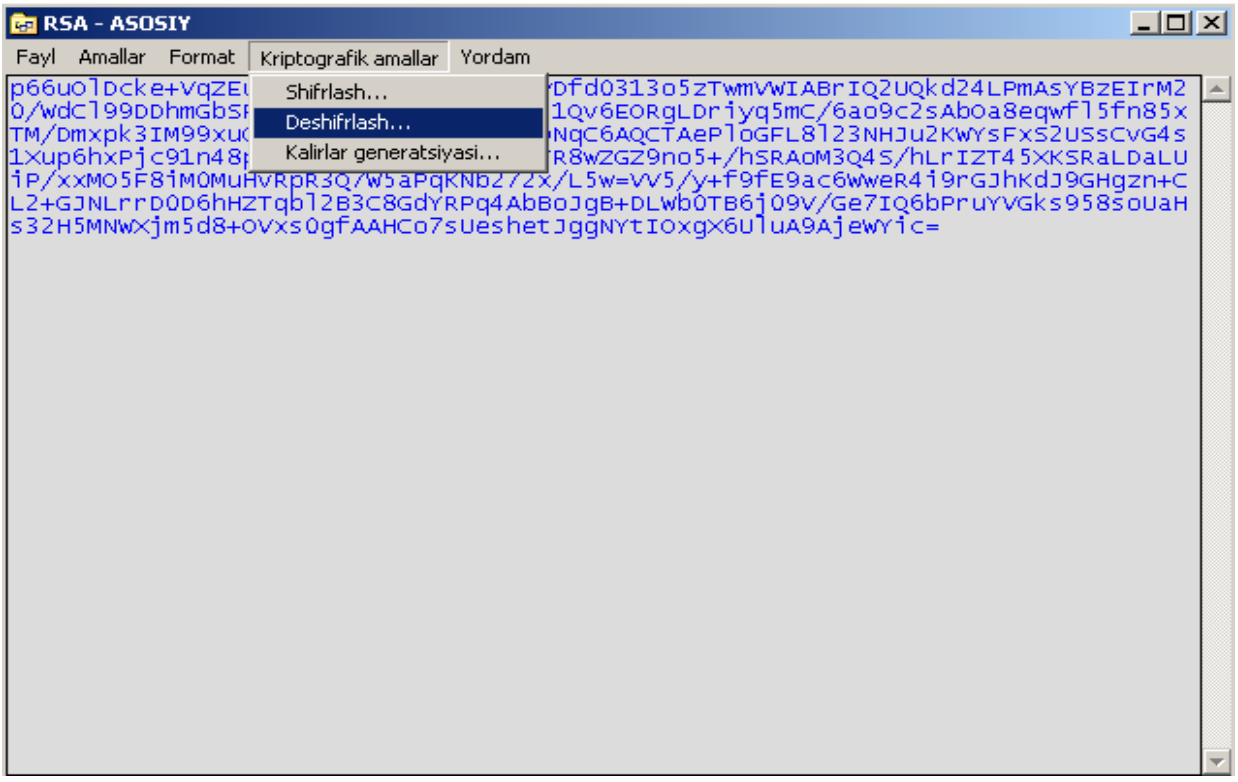
калитлар генерацияси



Шифрлаш буйруғи

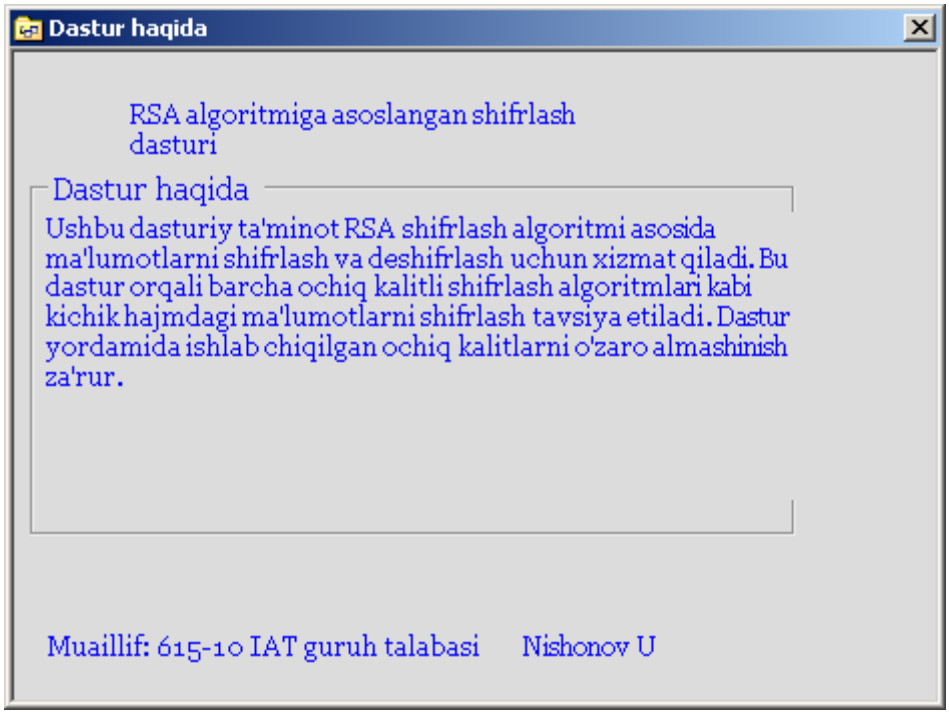
Шифрлаш амалини бажариш учун матн терилади ёки файл менюсидан очиш амали танланади ва керакли матн файли танланади. Акс ҳолда хатолик кўрсатади. Сўнгра шифрлаш буйруғИ танланади ва очиқ калит файли танланади. Шундан сўнг қуйидагича шифрланган матнга эга

бўламиз



Дешифрлаш буйруғи орқали эса ёпиқ калит ёрдамида шифрматн дастлабки матнга ўгирилади

Дастур ҳақида ойнаси



2.4 Қисм бўйича хулоса

Бу бўлимда RSA очиқ калитли шифрлаш алгоритми, RSA шифрлаш алгоритми учун дастурий таъминот яратишда C# дастурлаш тилининг аҳамияти ҳамда RSA шифрлаш дастуридан фойдаланиш йўриқномаси ҳақида тўхталиб ўтдим.

3.1 Маълумотларни шифрлаш алгоритмларининг аҳамияти

Юқорида ўрнига қўйиш ва ўрин алмаштиришга асосланган шифрлаш алгоритмларини, уларнинг асосидаги акслантиришларни математик моделларининг асосий хусусиятлари кўриб ўтилди.

Ўрнига қўйишга асосланган шифрлаш жараёнида очиқ маълумотни ташкил этувчи алифбо белгиларини айрим (алоҳида) олинган ҳолда, шифрмаълумот алифбосининг айрим (алоҳида) олинган белгиларига алмаштириш ёки ўрин алмаштиришга асосланган шифрлаш жараёнида очиқ маълумотни ташкил этувчи алифбо белгиларини айрим (алоҳида) олинган ҳолда ўринларини алмаштириш амалга оширилган бўлсин. Бундай ҳолатда шифрлаш жараёни алгоритмининг криптобардошлилигини ошириш учун калит узунлиги шифрланиши керак бўлган маълумот узунлиги даражасида бўлиши зарур бўлади. Мисол учун, шартли равишда, бирор алифбода тузилган ушбу “$x_1 x_2 \dots x_N$” – очиқ маълумотдан, уни ташкил этувчи алифбо белгиларининг ўринларини алмаштириш натижасида “$x_{i_1} x_{i_2} \dots x_{i_N}$” – шифрмаълумот ҳосил қилинган бўлса, у ҳолда калитни ифодаловчи $1 \to i_1, 2 \to i_2, \dots, N \to i_N$ - ўрин алмаштиришлар сони очиқ маълумотни ташкил этувчи алифбо белгиларининг сони билан тенг. Худди шу каби, ўрнига қўйишга асосланган шифрлаш алгоритмларидан фойдаланишда очиқ маълумот частотавий хусусиятларининг шифрмаълумотга кўчмаслигини таъминлаш учун кўп алифболи шифрлаш алгоритмларидан фойдаланилади, бунга эришиш учун эса, юқорида кўрилганидек шифрлаш жараёни босқичларида бир хил белгиларни ҳар хил белгиларга алмаштириш, яъни калит узунлигини ошириш зарурати туғилади. Шифрланиши керак бўлган маълумот ҳажмининг ортиши билан, шифрлаш жараёнини амалга оширишда қўлланиладиган алгоритм калити узунлигининг мос равишда ортиб бориши, криптобардошлиликни таъминлаш нуқтаи назаридан самарали бўлсада, бундай ҳолат алгоритмларнинг амалда қўлланишлари нуқтаи назаридан: калитларни сақлашда, уларни тарқатишда, аппарат-техник таъминотларни

амалга оширишда ва бошқа шу каби ҳолатларда ноқулайликлар туғдиради. Шунинг учун шифрланиши керак бўлган маълумотни, уни ташкил этувчи алифбо белгиларининг маълум бир узунликдаги бирикмалари (блоклари) бирлашмаси (конкатенацияси) кўринишда ифодалаб, ана шу блокларнинг алоҳида-алоҳида самарали ва криптобардошли шифрланишини амалга ошириш масаласи келиб чиқади. Бу масала симметрик блокли шифрлаш алгоритмлари орқали амалга оширилди. Симметрик блокли шифрлаш алгоритмларининг асосини очиқ маълумот блокларини юқори даражада *аралаштириш* ва *тарқатиш* (ёйилиш, таралиш) хоссаларига эга бўлган акслантиришлар ташкил этади [13, 59-60]. *Самарали аралаштириш* берувчи ($\oplus$, $\bmod 2^n$, *ўрин алмаштириш жадваллари*, *циклик суришлар* ва ҳоказо) амаллар *корреляцион иммунстлик* – шифрланиши керак бўлган ёки калит блокларини ташкил этувчи алифбо белгиларидан бирининг ўзгариши, акслантириш натижасида олинган шифрблокни ташкил этувчи алифбо белгиларининг фақат биргина мос белгиси ўзгаришига таъсир қилиб, бошқа қисмига таъсир этмаслигини таъминловчи ўрин алмаштиришга асосланган шифрлаш акслантиришларидан иборат. *Самарали тарқатиш* берувчи бир алифболи ва кўп алифболи ўрнига қўйиш акслантиришларга асосланган S блок акслантиришлари *чизиқсизликни* - шифрланиши керак бўлган ёки калит блокларини ташкил этувчи алифбо белгиларидан бирининг ўзгариши, акслантириш натижасида олинган шифрблокни ташкил этувчи алифбо белгиларининг икки ва ундан ортиқ қисмига таъсир этишини таъминловчи ўрнига қўйишга асосланган шифрлаш алгоритмлари акслантиришларидан иборат.

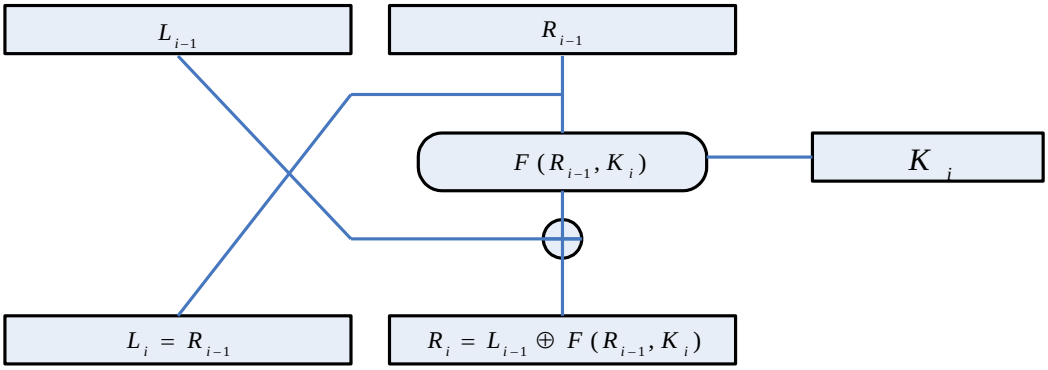
Аралаштирувчи акслантиришлар очиқ маълумот ва унга мос келувчи шифрмаълумот блокларининг частотавий (статистик) ва аналитик боғлиқлик хусусиятларини ўрнатишни мураккаблаштирса, тарқатувчи акслантиришлар очиқ маълумот блоки битта белгисининг ўзгаришини мос шифрмаълумот блокининг кўп белгилари ўзгаришига таъсир қилишини юзага келтириб,

очиқ маълумотнинг частотавий (статистик) хусусиятларини шифрмаълумотга кўчмаслигини таъминлайди.

Симметрик блокли шифрлаш алгоритмлари бир нечта босқичлардан (раундлардан) иборат бўлиб, ҳар бир раунд аралаштирувчи ва тарқатувчи акслантиришлардан тузилган. Бундай асосда тузилиш тамойили, ҳар бир раунд шифрлаш жараёнини ҳар хил калитлар билан бир хил турдаги акслантиришларни амалга оширишга ҳамда дешифрлаш жараёнини раунд акслантиришлари ва калитларини тескари тартибда қўллашнинг самарали имконини беради. Алгоритм асосини ташкил этувчи, раунд шифрлаш жараёнини амалга оширувчи, аралаштириш ва тарқатиш хусусиятларига эга бўлган функциялар *асосий акслантиришлар* дейилади. *Асосий акслантириш*ларнинг аппарат-техник жиҳатдан қулай қўлланиш модели сифатида тескари боғлиқликка эга бўлган силжитиш регистларини келтириш мумкин [13, 59-60]. Бунда тарқатувчи акслантириш тескари боғлиқликни таъминловчи функция билан, аралаштирувчи акслантириш эса, регистрдаги маълумотларни силжитиш билан амалга оширилади.

Шифрланиши керак бўлган маълумот блокини силжитиш регистрларига киритиб (юклаб), регистрдаги маълумотни шартли равишда чап ва ўнг қисм блок векторларига бўлиб, улар устида ҳар хил калитлар билан бир хил турдаги акслантиришларни босқичма-босқич амалга оширишга асосланган – *Фейстел (Фейштел) тармоғи* деб аталувчи шифрлаш жараёни функционал қурилмасига асосланган алгоритмлар кенг тарқалган. Булар жумласига DES ва ГОСТ 28147-89 киради.

Файстел тескариси мавжуд криптобардошли акслантиришларни тадқиқ қилмай, бундай акслантиришлар қатнашмаган криптобардошлилиги юқори бўлган шифрлаш тизимларини топиш масаласининг ечимига киришган. У бу масаланинг ечимини қуйидагича ҳал этган. Шифрланадиган блок иккита L_0, R_0 қисмларга ажратилади. Фейстел тармоғи i – раунди итератив блокли шифрлаш қуйидаги схема бўйича аниқланади (10-расм).



10-расм. Фейстел тармоғи i – раунди

Бу ерда $X_i = (L_{i-1}, R_{i-1})$ – i -раунд учун L_{i-1} ва R_{i-1} қисмларга ажратилган кирувчи маълумот, $Y_i = (L_i, R_i)$ эса X_i ни i – раунд калити K_i билан F акслантириш натижасида ҳосил бўлган шифрмаълумот.

Фейстел тармоғи i – раунди шифрлаш жараёнинг математик модели қуйидагича ифодаланади:

$$\begin{cases} L_i = R_{i-1}, \\ R_i = L_{i-1} \oplus F(R_{i-1}, K_i). \end{cases}$$

Бундай тармоққа асосланган алгоритмлар бир неча итерациядан ташкил топган K_i калитларда шифрланадиган акслантиришлардан (функциялардан) ташкил топган.

Фейстел тармоғи акслантиришларининг асосий хоссаси F -раунд функцияси тескариси мавжуд бўлмаса ҳам, Фейстел тармоғи бу акслантиришларининг тескарисини топиш имконини беради. Ҳақиқатан ҳам, шифрлаш жараёни i – раунд математик моделидаги $\oplus$ - модуль 2 бўйича иккилик санок тизимида қўшиш амали хоссасидан фойдаланган ҳолда қуйидаги тенгликка эга бўлинади:

$$\begin{cases} R_{i-1} = L_i, \\ L_{i-1} = R_i \oplus F(L_i, K_i). \end{cases}$$

Бу тенгликлар Фейстел тармоғи асосида қурилган шифрлаш алгоритмларини дешифрлашнинг математик моделини ифодалайди.

3.2 AQSH va O’zbekistonning shifrlash bo’yicha davlat standartlari va ularning mazmuni

АҚШ давлат стандарти **AES FIPS-197** алгоритми раундларининг шифрлаш жараёнлари: *SubBytes* –берилган жадвал асосида байтларни алмаштириш, *ShiftRows* – берилган жадвал асосида байтларни циклик суриш, *MixColumns* –тескариси мавжуд бўлган берилган матрица бўйича байтларни аралаштириш, *AddRoundKey* –раунд калитлари блоки битларига мос блоклар битларини *XOR* амали билан қўшиш акслантиришларида иборат бўлиб, бу акслантиришларнинг биттаси, яъни *AddRoundKey* акслантириши бир томонлама ҳисобланади. Чунки раунд калити блоки ва унга *XOR* амали билан қўшилувчи мос блок номаълум бўлиб, бу акслантириш натижаси маълум бўлганда унга мос келувчи блокни аниқлаш учун раунд калитини топиш керак бўлади. Бундай ҳолат эса раунд калитларининг барча мумкин бўлган қийматларини танлаб чиқишни талаб этади. Раунд калит узунлигининг қанчалик катта бўлиши ва раундлар сонининг кўп бўлиши алгоритм криптобардошлилигини ифодалайди. **AES FIPS-197** алгоритми раунд калитининг энг кичик узунлиги 128 бит бўлиб, барча мумкин бўлган қийматлари сони 2^{128} та, бу узунликдаги барча мумкин бўлган ҳолатларни танлаб чиқишни бугунги ҳисоблаш техника ва технологиялари имкониятларидан самарали фойдаланилганда ҳам мумкин қадар қисқа вақт ичида амалга ошириш иложиси мавжуд эмас. Алгоритм 10 раунддан иборат.

O‘z DSt 1105:2009 «Ахборот технологияси. Ахборотнинг криптографик муҳофазаси. Маълумотларни шифрлаш алгоритми» стандарти электрон маълумотларни муҳофаза қилиш учун мўлжалланган криптографик алгоритмни ифодалайди. Маълумотларни шифрлаш алгоритми (МША) - симметрик блокли шифр бўлиб, ахборотни шифрматнга ўгириш ва дастлабки матнга ўгириш учун фойдаланилади. МША *256* bit узунликдаги маълумотлар блокини шифрматнга ўгириш ва шифрматнни дастлабки матнга ўгириш учун *256* ёки *512* bit узунликдаги криптографик калитдан фойдаланиши мумкин.

O‘z DSt 1105:2009 криптоалгоритмининг математик асоси

МШАда модуль арифметикасининг диаматрицалар алгебрасидан фойдаланилади, бунда ҳисоблашнинг қийинлик даражаси матрицалар алгебрасидаги сингари бажарилади.

Шифрматнга ўгириш ва дастлабки матнга ўгириш процедураларида фойдаланиладиган диаматрицалар алгебрасининг асосий амали диаматрицани p модуль бўйича диаматрицага тескарилаш амали ҳисобланади. Бу амалларда икки ўлчамли сеанс калити массивининг махсус тузилмали $4x4$ тартибли квадрат диаматрица билан акс эттирилувчи қисмлари иштирок этади; махсус тузилмали диаматрица учун барча диагонал элементлар бир хиллиги, *1*-сатрдаги нодиагонал элементлар, шунингдек *2*-сатрнинг боши ва охиридаги элементлар ҳам бир хиллиги хосдир.

Махсус тузилмали диаматрицанинг муҳим хоссаси диаматрицанинг диааниқловчисини ҳисоблаш формуласининг соддалигидир, бу эса диаматрицани тескарилаш шартларини текшириш ишларини соддалаштиради. Махсус тузилмали диаматрицага нисбатан тескари диаматрица ўзининг дастлабки тузилмасини сақлайди.

$4x4$ тартибли махсус тузилмали диаматрица 10 та ҳар хил элементлар $a_0, \ldots, a_9$ дан тузилган бўлиб, унинг диааниқловчиси диагонал элемент a_7 ни учта йиғиндига кўпайтмаси сифатида топилади, бу йиғиндилардан ҳар бири диагонал элемент билан битта сатрда жойлашган унга ўнгдан қўшни элемент билан устун элементларининг йиғиндисини ифодалайди.

Махсус диаматрица учун диааниқловчи a қуйидагича топилади:

$$d \equiv a_7 \times (a_7 + a_0 + a_8 + a_3 + a_5) \times (a_7 + a_1 + a_8 + a_9 + a_6) \times (a_7 + a_2 + a_8 + + a_9 + a_4) \pmod p.$$

Махсус тузилмали диаматрицани тескарилаш шартларини текшириш МША параметрларига қўйиладиган асосий талаб ҳисобланади. У диагонал элементнинг қийматларини ва айтиб ўтилган кўпайтмаларни *2* модули бўйича ноль билан таққослашга келтирилади. Бу ҳар қандай шифрлаш калити ва функционал калитдан тескари диаматрицани шакллантиришга имкон беради.

МШАда, шунингдек бутун сонларни параметрли кўпайтириш, тескарилаш ва даражага ошириш деб аталган параметрли группа амалларидан ҳам фойдаланилади.

МША учун босқич (раунд)лар сони $e=8$ қилиб белгиланган.

Маълумотларни шифрлаш алгоритмининг параметрлари ва функциялари

МША қуйидаги параметр ва функциялардан фойдаланади:

a) k – *256* ёки *512* bit узунликдаги шифрлаш калити;

b) k_f – *256* bit узунликдаги функционал калит;

c) K_e – *8*x*4 (*ёки *4*x*8)* тартибли икки ўлчамли массив шаклидаги босқич калити;

d) b – *256* bit ли кириш блоклари сони;

e) e – босқичлар сони, $e=8$;

f) $p, (p+1)$ – модуль, $p=256$;

g) *Aralash()* – оддий шифралмаштириш бўлиб, дастлабки матнни шифрматнга ва тескари йўналишда алмаштириш учун диаматрицавий қисмлар устида амалга оширилади; мазкур шифралмаштириш кириши *Holat* массивининг диаматрицавий қисмлари ҳамда K_1 ва K_2 массивлари бўлиб, чиқиши *Holat* массивидир;

h) *BaytAlmash()* – оддий шифралмаштириш бўлиб, дастлабки матнни шифрматнга ва тескари йўналишда *Holat* массиви элементларини алмаштириш массиви элементлари билан байт сатҳида алмаштириш учун фойдаланилади; мазкур шифралмаштириш кириши байт сатҳида *Holat* массиви, алмаштириш массиви чизиқли массив B_{sA} *[256]* ёки B_{sAD} *[256]* бўлиб, чиқиши байт сатҳида *Holat* массивидир;

i) *Sur()* – *Holat* массиви элементларини янада яхшироқ аралаштириш учун дастлабки матнни шифрматнга ва тескари йўналишда алмаштиришда фойдаланилади; мазкур алмаштириш кириши байт сатҳида *Holat* массиви, чиқиши устун бўйлаб шифрлашда пастга ва сатр бўйлаб ўнгга ёки шифрни

очишда устун бўйлаб юқорига ва сатр бўйлаб чапга сурилган байт сатҳида *Holat* массивидир;

j) *ShaklSeansKalitBayt()* – сеанс учун калит шакллантириш бўлиб, дастлабки матнни шифрматнга ва тескари йўналишда алмаштиришда *BaytAlmash()* шифралмаштиришини бажариш учун фойдаланилади; мазкур шифралмаштириш кириши шифрлаш калити *k* ва функционал калит k_f бўлиб, чиқиши байт сатҳида чизиқли массивлар B_{sA} *[256]* ва B_{sAD} *[256]*;

k) *ShaklSeansKalit()* – сеанс учун калитни шакллантириш бўлиб, дастлабки матнни шифрматнга ва тескари йўналишда алмаштиришда *Aralash()* шифралмаштиришни бажариш учун фойдаланилади; мазкур шифралмаштириш кириши байтли элементлардан таркиб топган чизиқли массив K_{st}*=[32]* бўлиб, чиқиши махсус тузилмали диаматрицалардан ташкил топган (K_{1t}, K_2) ёки (K_1, K_{2t}) массивлар жуфтликларидир;

l) *ShaklBosqichKalit()* – сеанс давомида сеанс-босқич калитидан босқич калитини шакллантириш бўлиб, дастлабки матнни шифрматнга ва тескари йўналишда алмаштиришда *Qo'shBosqichKalit()* алмаштиришини бажариш учун фойдаланилади; мазкур алмаштириш кириши чизиқли сеанс-босқич калити массиви k_{se}, чиқиши байт сатҳида берилган икки ўлчамли K_e*[8,4]* массивидир;

m) *Qo'shBosqichKalit()* – оддий шифралмаштириш бўлиб, дастлабки матнни шифрматнга ва тескари йўналишда *Holat* ва босқич калити массиви K_e элементлари устида истисноли ЁКИ (*2* модули бўйича битлаб қўшиш) амалини бажаришдан иборат; мазкур шифралмаштириш кириши байт сатҳида *Holat* массиви, K_e массиви бўлиб, чиқиши байт сатҳида *Holat* массивидир;

n) *Qo'shHolat()* – оддий шифралмаштириш бўлиб, шифрлаш блоклари устида амалга ошириладиган электрон код китоби режимидан бошқа режимларда дастлабки матнни шифрматнга ва тескари йўналишда *XOR* амали иштирокида фойдаланиладиган алмаштириш.

МША белгилаб қўйилган икки хил - *256* ва *512* бит узунликдаги калитлар ёрдамида амалга оширилади.

Биринчи ҳолатда, шифрлаш криптографик модулига *256* битли калит киритилади. Бу калит тўлалигича шифрлаш калити k сифатида олинади, дастлабки сеанснинг k_f функционал калити эса, шифрлаш калитининг хэш-функцияси қиймати сифатида ҳисоблаб топилади.

Иккинчи ҳолатда, шифрлаш криптографик модулига *512* битли калит киритилади. Бу калитнинг *256* битли биринчи ярми, шифрлаш калити k сифатида олинади, унинг *256* битли иккинчи ярми биринчи сеанснинг функционал калити k_f сифатида олинади.

Учинчи ҳолатда, шифрлаш криптографик модулига ҳеч қандай янги калит киритилмайди. Шифрлаш калити k сифатида олдинги сеансда ишлатилган шифрлаш калити олинади, функционал калит k_f сифатида эса олдинги сеансда ишлатилган функционал калит k_{f-1} нинг шифрлаш калити k дан фойдаланиб хэшланган қиймати олинади.

3.3 Қисм бўйича хулоса

Бу бўлимда маълумотларни шифрлаш алгоритмларининг аҳамияти АҚШ ва Ўзбекистоннинг шифрлаш бўйича давлат стандартлари ва уларнинг мазмуни ҳақида тўхталиб ўтдим

МЕҲНАТ МУХОФАЗАСИ

Микроиқлимнинг инсон организмига таъсири

Ишлаб чикариш биносининг микроиклими ходимга катта таъсир курсатади. Тавсия этиш мазмундан микроиклимнинг айрим улчамларининг четга чиқиши мехнатга лаёкатни сустлаштиради, ходимнинг хиссиётини ёмонлаштиради ва касбий касалликларга олиб келиши мумкин.

Хаво харорати. Паст харорат организмнинг совуб кетишга хамда шамоллаш касалликлари чиқишига сабаб бўлади.

Юқори хароратда – организм кизиб кетади, жуда куп микдорда терлайди, мехнатга лаёкат сустлашади. Ишчи эътибори сустлашиб, бахциз ходисага олиб келиши мумкин.

Хавонинг Юқори намлиги тери ва упканинг устки кисмидан намликнинг бугланишини кийинлаштиради ва огир – оқибатда организмнинг терморегуляцияси бузилишига, инсон ахволининг ёмонлашуви, мехнатга лаёкатлиликнинг сустлашувига олиб келади. Паст намликда (< 20%) – Юқори нафас юлларининг шиллик пардалари кўриб қолиши кўзатилади.

Хаво харакати тезлиги. Инсон в к□0,15 м/сек.да хаво харакатини сеза бошлайди. Хаво оқимининг харакати унинг хароратига боғлиқ. 36^0 С > т да оқим инсонга салқинлатувчи таъсир, 40^0 С < т да нокулай, ёмон, салбий таъсир кўрсатади.

Микроиқлим кўрсаткичлари

Микроиклим ишчи худудда ишчиларнинг доимий ва ёки вақтинча турган жойидан 2 м баландликда бахоланади.

Енг кулай шароитлар – терморегуляция механизмлари кучланишисиз организмнинг нормал исиклик ахволини таъминловчи хамда узоқ ва мунтазам инсонга таъсир килувчи микроиклим улчамларининг йигиндиси. Улар мехнатга кобилиятлиликнинг юксак савияси учун шарт – шароит яратади ва иссик - кулай комфорт сезувчанликни таъминлайди.

Инсонга узоқ мунтазам таъсир этишда терморегуляция механизмлари-кучланиши билан давом этадиган организмнинг иссиклик холатида дархол нормаллашувчи узгаришлар чакирадиган микроиклим улчамлари йигиндиси юл қўйиладиган иклим шароитлари деб каралади. Бундай холда организмга шикаст етмайди ёки саломатликнинг ахволига зарар булмайди, бирок дискомфорт иссикликни сезиш, инсон узини ёмон хис қилиши ва мехнатга лаёкати пасайиши (сустлашиши) мумкинлиги кўзатилади.

Хонани танлаш

Хона кенг, меоёрида ёритилган ва ҳавоси осон алмаштириладиган бўлиши керак. Ёрқин қуёш нурлари мониторга салбий таъсир этади.Қоронғи хонада фақат иш жойини ёритиш ҳам мақсадга мувофиқ эмасдир.Столни шундай жойлаштиринг-ки, дераза ойнаси қаршингизда бўлмасин. Агар бунинг иложи бўлмаса, у ҳолда қалин парда ёки жалюзи сотиб олинг, шунда деразадан тушаётган ёруғлик сизга халал бермайди. Агар ойна ён томонда бўлса, яна парда ёки жалюзи жонингизга оро киритади. Чанг ва иссиқлиқ саломатликка путур етказибгина қолмай, техникага ҳам ёмон таъсир ўтказади, шунинг учун хонага кондиционер ўрнатган маъқул.

Инсон организмига электр токининг таъсири

электр курилмаларини ишлатишда изоляция шкастланиши натижасида машина корпуси кучланиш остида колиб,одам унга тегиб кетгинида электр токи уради.

Одам танаси оркали утган электр токи термик, электр ва биологик таъсир курсатади.

Токнинг термик таъсири терининг айрим жойларини қуйишида, кон томирлари, кон, юрак, мия ва бошка аозоларининг юкори хароратгача кизишида номоён булади.

Токнинг электр таъсири кон ва бошка органик суюкликларнинг парчаланишида намоён булади. Окибатда уларнинг физик – кимёвий таркиби бузилади.

Токнинг биологик таъсири организмнинг тирик тукималари яллиланиши ва асабийлашида намоён булади.

Бунда мушаклар, шу жумладан, юрак ва упка мушаклари ихтиёрсиз равишда тортишиб коладию, натижада организмда хар – хил бузилишлар руй бериши, масалан, нафас олиш ва кон айланиш органларининг иши бузилиши ёки хатто батамом тўхтаб колиши мумкин.

электр токи таъсирининг бу турлари шкастланишининг икки турини келтириб чиқаради.электр токи шкастланиши ва электр токи уриши.

электр токи шкастланиши бу, электр токи ёйи таъсири этиши натижасида организмнинг айрим жойларидаги тукималарнинг яккол шикастланишидир. электр токи шкастланишнинг куйдаги турлари билан фаркланади: электр токидан қуйиш, электр излари, таринининг металланиши ва механик шикастланишлар.

электр излари ток таъсир этган одамнинг танаси сиртида аник куриниб турадиган кулранг ёки оч сарик рангдаги доглардир.

Излар, тирналишлар, кичик жарохатлар кесиклар ёки латлар куринишида бўлади. Терининг шкастланган кисми кадок сингари каттиқлашиб колади.

Теринг маталланиши электр ёйи таъсирида эриган металл майда заррачаларнинг терининг устки катламига кириб қолишидир.

Бу ходиса, масалан, киска товушларда, кучланиш остида булган ажратгич ва рубилиникларни тармокдан узатаётганда руй беради.

Механик шкастланишлар одам оркали ўтаётган ток таъсирида мушакларнинг ихтиёрсиз равишда кескин тортишиб колиши оқибатида юз беради. Натижада тери, кон томирлари ва асаб тўқималари узилиши,шунингдек бўғинлар чиқиши ва хатто суяклар синиши мумкин.

электр токи уриши деганда, организм оркали электр токи утганида тирик тукималарнинг асабийлашиши натижасида мушакларнинг ихтиёрсиз равишда тортишиб колиши тушунилади.

Одам организми электр токининг таъсири кандай оқибатларга олиб келишига караб, электр токи уришининг шартли равишда куйдаги тўрт даражага ажратиш мумкин:

1. Даража одамнинг мушаклари тортишиб колади, аммо у хушидан кетмайди;

2 Даража одамнинг мушаклари тортишиб колади, у хушидан кетади, лекин у нафас олади, ва юраги ишлайди;

3 Даража одамнинг мушаклари тортишиб, юрагининг ишлаши ёки нафас олиши бузилади, (ёки икиилови баравар руй беради);

4-- даража клиник ўлим юз беради, яъни нафас олиш ва қон айланиши тўхтайди.

Клиник ўлим хаёт билан ўлим ўртасидаги холат бўлиб, юрак ва ўпка ишлашдан тухтаган пайитдан бошланади.Клиник ўлим холатида бўлган одамда хеч кандай хаёт белгилари булмайди: у нафас олмайди, юраги ишламайди, орикни сезмайди, куз корачиги кенгаяди ва ёруғликни сезмайди.Аммо бу давр организмида хаёт хали бутунлай сўнмаган бўлади, чунки унинг туқималари дарров ўлмайди ва турли аъзолари хали ишлаб туради. Гарчи бу жараён энди жуда суст, одатдагидан фаркли равишда кечсада,аммо энг кичик хаёт фаолияти учун етарли бўлади.

Биринчи навбатда кислород етишмаслигига жуда сезгир бўлган бош мия кобиғининг хужайралари ўла бошлайди.Онг ва таффакур ана шу хужайраларнинг фаолиятига боғлик.Шу сабабли клиник ўлимнинг давом этиш вакти юракнинг ишлаши ва нафас олиш тухтаган пайтда то бош мия хужайралари ула бошлайдиган пайтга кадар ўтадиган вакт билан аникланади.Куп холларда бу вакт 4—6 минут, соғлом кишиларда тасодифан электр токи уриши натижасида улганда эса 7—8 минутни ташкил этади.

Биологик (хакикий) улимни кайтариб булмайди, ходиса булиб, бунда организм хужайралари ва тукималарида биологик жараёнлар тухтайди.

Ток тўғридан тўғри юрак мускулларига таъсир кили бунинг тзхтаб қолишига ёки мускулларга таъсири қилиб унинг тўхтаб қолишига ёки

фибрациясига сабаб бўлади. Бундай шароитда юрак насос сингари ишлай олмайди. Натижада қон айланиши тўхтайди ва организм ўлади.

Одам танасидаги хар хил туқималар электр токига турлича қаршилик кўрсатади.Масалан,тери, унинг эпидермис деб аталадиган ташқи катламлари нинг калинлиги 0,1 –0,5 мм булади ва асосан жонсиз котиб кетган хужайралардан ташкил топади. Бу катламнинг қаршилиги катта, бўлиб одам танасининг умумий қаршилигини белгилайди.Одам танаси ички туқималарнинг қаршилиги ---- 300 --- 500 Ом ни ташкил этади.Одам танасининг қаршилиги 3000 дан 1000000 Ом гача ўзгариб туради.Шикастланган тананинг қаршилиги энг паст 300 --- 500 Ом бўлади. Ток катталашиши ва унинг танадан утиб туриш вакти ортиши билан тер чикиши купайиши ва бошка омиллар туфайли тананинг қаршилиги пасаяди. қаршиликни хисоблашда одам танасинрининг уртача қаршилиги 1000 Ом га тенг қилиб олинади.

Шикастланиш даражаси кўп даражада токнинг тури ва частотасига боғлик.20--- 1000 Гц частотали узгарувчан ток энг хавфлидир. Частотаси 20 Гц дан кичик ёки 1000 Гц дан катта булганда токнинг хавфлилиги анча пасаяди.

Одамларни электр токидан шкастланишининг асосий сабаблари қуйидагалардан иборат:

1.Кучланиш остида булган ток ўтказувчи кисмларга тасодифан тегиб кетиш, ток ўтказувчи кисмларда кучланиш борлигини билмай қолганда юз бериши мумкин.

2.электр курилмасининг одатдаги шароитда кучланиш остида булмайдиган, аммо тасодифан кучланиш остида колган маталл кисмларига тегиб кетганда.

3.Одам турган ер кадам кучланишининг пайдо булиш.Бу хол симнинг ерга туташиб колиши, потенциал чикиб кетиши, химояловчи ерга улаш ускунисининг, ноллаш симнинг бузилганлиги ва бошка сабаблар туфайли юз беради.

Одамнинг ток занжирига уланиб колиш схемаси турлича булиши мумкин: одатда икки фазага тегиб кетиш; ва бир фазага тегиб кетиш.

Икки фазага тегиб кетиш одатда, хавфлирокдир, чунки бундай одам танасига ушбу тармокдаги энг катта кучланиш ----- линия кучланиши таъсир килади ва шу сабабли одам оркали энг катта кийматли ток ўтади:

$$I_0 = \frac{U_л}{R_0} = \frac{1,7U_ф}{R_0}$$

Бу ерда: I_0— одам танаси оркали утаётган ток, А;

$U_л$--- линия кучланиши, яони тармокнинг фаза симлари уртасидаги кучланиш, В;

$U_ф$ – фаза кучланиш, яони битта чулгамнинг боши билан охири уртасидаги (ёки фаза сими билан нолинчи сим орасидаги) кучланиш, В.

Одам танасининг қаршилиги R_0= 1000 Ом булганда линия кучланиши $U_л$= 380 В (бинобарин фаза кучланиши $U_ф$= 220 В) булган тармокда қуйидагига тенг булади:

$$I_0 = \frac{1,73 * 220}{1000} = 0,38\ А = 380\ мА$$

Бундай ток одамни халока килади.

Хатто одам ердан ишончли тарзда изоляцияланган яони резина калиш ёки боти кийган ёхуд изоляцияловчи тахта ёки электр ўтказмайдиган пойандозда турган булса хам, агар у икки фазага тегиб кетса, токдан шикастланиш хавфли камаймайди.

Бундан ташкари, одам турган полнинг қаршилиги, оёьидаги пойабзалнинг қаршилиги ва баози бошка омиллар хам таъсир курсатади.

Нейтрали ерга уланган тармокда одам танасининг қаршилиги R_0 билан кетма-кет тарзда пойабзалнинг қаршилиги R_{ey} полнинг қаршилиги R_0 ва ток манбаи нейтралини ерга улаш симининг қаршилиги R_{ey} уланиб колади.

Ушбу қаршиликларни хисобга оладиган бўлсак, одам орқали утаётган ток қуйидагига тенг булади:

$$I_0 = \frac{U_\phi}{P_0 + R_{па} + R_n + R_{ey}} \qquad (2)$$

Шундай бир ходисани куриб чикамиз. Фараз килайлик, оёьига ток ўтказадиган пойабзал кийиб олган одам зах ерда металл устида турган булсин. Бу холда $R_{па}=0$ ва $R_п=0$ деб олиш мумкин. Бундан ташқари, нейтрални ерга улаш симининг қаршилиги R_{ey} одатда 10 Ом дан катта бўлмаганлиги учун уни хисобга олмас хам бўлади.

Натижада (2) тенглама ушбу кўринишни олади:

$$I_0 = \frac{U\phi}{R_0} \qquad (3)$$

(2) ва (3) тенгламаларни таккослаб икки фазага тегиб кетиш хавфлирок эканлигига яна бир бор ишонч хосил киламиз, чунки бунда одам танасидан утувчи ток нокулай шароитда бир фазага тегиб кетгандагига караганда деярли икки баравар катта булади.

Бирок бундай шароитда бир фазага тегиб кетиш хам, ток кичик булишига карамасдан жуда хавфли хисобланади. Масалан: фаза кучланиши $U_ф=220$ В ва $R_0=1000$Ом булганда (3) га мувофик одам оркали ўтадиган ток кучи: $I_0=\frac{220}{1000}=0{,}22$А$=220$мА. Бундай ток одамни халок этади.

Агар одам оёғига ток ўтказмайдиган, масалан: резина пойабзал кийиб олган ва ток ўтказмайдиган тахта пол устида турган бўлса, у холда $R_{па}=50000$ Ом ва $R_0=60000$ Ом деб олиб, (2) га мувофик ушбу формулага эга бўламиз:

$$I_0 = \frac{220}{1000 + 50000 + 60000} + 0{,}002\text{А} = 2\text{мА}$$

бундай ток одам учун хавфли эмас.

Нейтрали изоляцияланмаган тармокда одам танамида ўтадиган ток симларнинг катта қаршиликка эга булган изоляцияси оркали ток манбаига кайтиб келади.

ХУЛОСА.

Ҳозирги замоннинг устувор йўналишларидан бири бўлган ахборотлаштириш асрида яшаётган ҳар бир жамият аъзоси ахборот хавфсизлигини таъминлашни ва мустақиллигимиз шарофати муносабати ривожланиб келаётган техника тараққиёти даражасини чуқур англаб етиш, унинг яратилиш структураси билан танишиш, ишлаб чиқариш жараёнида содир бўлаётган янгиликларни таҳлил қилиш, ундан ташқари киритилаётган янги технологияларининг афзалликларини ва қулайлик тарифларини юқори савияда таҳлил этиш, унинг келиб чиқиш сабабларини, илдизларини ўрганишни ўз олдимизга қўйилган долзарб вазифалардан бири деб билмоғимиз даркор.

Бугунги даврда бутун жаҳонда ривожланиб бораётган ахборот технологияларидаги ўзгаришлар, дастурлаш тилларининг янги версияларини яратилиши, бундан ташқари мамлакатимизда ахборот технологиялари йўналиши бўйича йўлга қўйилаётган халқаро муносабатлар туфайли биз ёшларни чет эл мамлакатларига бориб ўқиб келишимиз, ўқиш давомида у ерда эгаллаган амалий кўникмаларимизни янги технологияларда синаб кўриш, мустақил Ватанимизни ривожланиш тараққиётига ўзимизнинг ҳиссамизни қўшишни асосий вазифамиз деб билишимиз керак.

Берилган мавзу бўйича топшириқни бажаришда шуни англаб етдимки, Криптографик алгоритмлар учун дастурий таъминот яратиш жараёнида C# дастуридан фойдаланиш мақсадга мувофиқ экан. Яратилган дастурдан тармоқ орқали ҳабарларни хавфсиз алмашиниш жараёнида ҳамда “Ахборот хавфсизлиги” фанини талабаларга ўқитиш жараёнларида фойдаланиш мумкин. Меҳнат муҳофазаси бўлимида микроиқлимнинг инсон организмига таъсири ва электр токидан муҳофаза қурилмалари ва ҳисоботлари хакида тўхталиб ўтдим.

Адабиётлар рўйҳати:

1. И.А.Каримов Жаҳон молиявий иқтисодий инқирози. Ўзбекистон шароитида уни бартараф этишнинг йўллари ва чоралари. – Т.: Ўзбекистон, 2009. – 56
2. Ахмедова О.П. Параметрлар алгебраси асосида носимметрик криптотизимлар яратиш усули ва алгоритмлари // Номзодлик диссертация иши, Тошкент-2007.
3. Бабаш А.В., Шанкин Г.П. История криптографии. Част И. – Москва: Лори Гелиос АРВ, 2002. – 240 с.
4. Арипов М.М., Пудовченко Ю.Е. Основи криптологии – Ташкент: 2004. – 136 с.
5. Ўз ДСт 1109:2006 «Ахборот технологияси. Ахборотнинг криптографик муҳофазаси. Атамалар ва таърифлар».
6. Акбаров Д.Е. Ахборот хавфсизлигини таъминлашнинг криптографик усуллари ва уларнинг қўлланишлари. Тошкент. ”Ўзбекистон маркаси “, 2009. – 432
7. Хасанов П.Ф., Исаев Р.И., Хасанов Х.П., Назарова М.Х. Ахмедова О.П. Ахборотнинг криптографик муҳофазаси тарихи (Илмий криптография даври) // Алоқа дунёси. – Тошкент, 2005, №2 (5). – 47-53 бетлар.
8. Венбо Мао. Современная криптография. Теория и практика. – Москва - Санкт-Петербург - Киев: Лори Вилямс, 2005. – 768 с.
9. Петцолъд Ч. Программирование для Мисрософт Виндовс на С#.Издателско-торговий дом «Русская Редакция», 2002.- 576 с
10. Троелсен. Е. С# и платформа .НЕТ. Библиотека программиста.Питер, 2004. —796 с
11. www.google.com
12. www.wikipedia.org
13. www.intuit.ru

ИЛОВА

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.IO;
using System.Runtime.Serialization;
using System.Runtime.Serialization.Formatters.Binary;
using System.Security.Cryptography;
using System.Text;
using System.Threading;
using System.Windows.Forms;
using System.Xml.Serialization;

namespace RSACryptoPad
{
	public class MainForm : Form
	{
		private FontDialog fontDialog;
		private IContainer components;
		private Label processingLabel;
		private MenuItem aboutMenuItem;
		private MenuItem copyMenuItem;
		private MenuItem cutMenuItem;
		private MenuItem decryptMenuItem;
		private MenuItem deleteMenuItem;
		private MenuItem editMenuItem;
		private MenuItem encryptionMenuItem;
		private MenuItem encryptMenuItem;
		private MenuItem exitMenuItem;
		private MenuItem fileMenuItem;
		private MenuItem fontMenuItem;
		private MenuItem formatMenuItem;
		private MenuItem generateKeyPairMenuItem;
		private MenuItem helpMenuItem;
		private MenuItem newFileMenuItem;
		private MenuItem openFileMenuItem;
		private MenuItem pasteMenuItem;
		private MenuItem saveAsFileMenuItem;
```

```
        private MenuItem saveMenuItem;
        private MenuItem selectAllmenuItem;
        private MenuItem separator1;
        private MenuItem separator2;
        private MenuItem separator3;
        private MenuItem undoMenuItem;
        private MenuItem wordWrapMenuItem;
        private MainMenu mainMenu;
        private OpenFileDialog openFileDialog;
        private Panel panel;
        private PictureBox pictureBox;
        private SaveFileDialog saveFileDialog;
        private TextBox inputTextBox;

        public static int currentBitStrength = 0;
        private bool cleanForm = true;
        private string currentFileName = "RSA";

        public delegate void FinishedProcessDelegate();
        public delegate void UpdateBitStrengthDelegate( int bitStrength );
        public delegate void UpdateTextDelegate( string inputText );


public MainForm()
        { InitializeComponent(); }
        public MainForm( string fileName )
        {
                InitializeComponent();
                if( File.Exists( fileName ) )
                {
                        currentFileName = fileName;
                        StreamReader streamReader = new StreamReader( fileName, true );
                        SetText( streamReader.ReadToEnd() );
                        streamReader.Close();
                        this.Text = GetFileName( fileName ) + " RSA";
                        cleanForm = true;
                }
        }

        protected override void Dispose( bool disposing )
        {
                if( disposing )
```

```
                {
                    if( components != null )
                    { components.Dispose(); }
                }
                base.Dispose( disposing );
            }

            private void InitializeComponent()
            {
        this.components = new System.ComponentModel.Container();
        System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(MainForm));
        this.mainMenu = new System.Windows.Forms.MainMenu(this.components);
        this.generateKeyPairMenuItem = new System.Windows.Forms.MenuItem();
        this.helpMenuItem = new System.Windows.Forms.MenuItem();
        this.aboutMenuItem = new System.Windows.Forms.MenuItem();
        this.inputTextBox = new System.Windows.Forms.TextBox();
        this.openFileDialog = new System.Windows.Forms.OpenFileDialog();
        this.saveFileDialog = new System.Windows.Forms.SaveFileDialog();
        this.fontDialog = new System.Windows.Forms.FontDialog();
        this.panel = new System.Windows.Forms.Panel();
        this.processingLabel = new System.Windows.Forms.Label();
        this.pictureBox = new System.Windows.Forms.PictureBox();
        this.panel.SuspendLayout();
        ((System.ComponentModel.ISupportInitialize)(this.pictureBox)).BeginInit();
        this.SuspendLayout();
        //
        // mainMenu
        //
        this.mainMenu.MenuItems.AddRange(new System.Windows.Forms.MenuItem[] {
        this.fileMenuItem,
        this.editMenuItem,
        this.formatMenuItem,
        this.encryptionMenuItem,
        this.helpMenuItem});
        //
        // fileMenuItem
        //
        this.fileMenuItem.Index = 0;
        this.fileMenuItem.MenuItems.AddRange(new System.Windows.Forms.MenuItem[] {
        this.newFileMenuItem,
        this.openFileMenuItem,
```

```
this.saveMenuItem,
this.saveAsFileMenuItem,
this.separator1,
this.exitMenuItem});
this.fileMenuItem.Text = "Fayl";
//
// newFileMenuItem
//
this.newFileMenuItem.Index = 0;
this.newFileMenuItem.Shortcut = System.Windows.Forms.Shortcut.CtrlN;
this.newFileMenuItem.Text = "Yangi";
this.newFileMenuItem.Click += new System.EventHandler(this.newFileMenuItem_Click);
//
// openFileMenuItem
//
this.openFileMenuItem.Index = 1;
this.openFileMenuItem.Shortcut = System.Windows.Forms.Shortcut.CtrlO;
this.openFileMenuItem.Text = "Ochish...";
this.openFileMenuItem.Click += new System.EventHandler(this.openFileMenuItem_Click);
//
// saveMenuItem
//
this.saveMenuItem.Index = 2;
this.saveMenuItem.Shortcut = System.Windows.Forms.Shortcut.CtrlS;
this.saveMenuItem.Text = "Saqlash...";
this.saveMenuItem.Click += new System.EventHandler(this.saveMenuItem_Click);
//
// saveAsFileMenuItem
//
this.saveAsFileMenuItem.Index = 3;
this.saveAsFileMenuItem.Shortcut = System.Windows.Forms.Shortcut.CtrlS;
this.saveAsFileMenuItem.Text = "Saqlash(yangi nom)... ";
this.saveAsFileMenuItem.Click += new System.EventHandler(this.saveAsFileMenuItem_Click);
//
// separator1
//
this.separator1.Index = 4;
this.separator1.Text = "-";
//
// exitMenuItem
//
this.exitMenuItem.Index = 5;
```

```
this.exitMenuItem.Text = "Chiqish";
this.exitMenuItem.Click += new System.EventHandler(this.exitMenuItem_Click);
//
// editMenuItem
//
this.editMenuItem.Index = 1;
this.editMenuItem.MenuItems.AddRange(new System.Windows.Forms.MenuItem[] {
this.undoMenuItem,
this.separator2,
this.cutMenuItem,
this.copyMenuItem,
this.pasteMenuItem,
this.deleteMenuItem,
this.separator3,
this.selectAllmenuItem});
this.editMenuItem.Text = "Amallar";
//
// undoMenuItem
//
this.undoMenuItem.Index = 0;
this.undoMenuItem.Shortcut = System.Windows.Forms.Shortcut.CtrlZ;
this.undoMenuItem.Text = "Undo";
this.undoMenuItem.Click += new System.EventHandler(this.undoMenuItem_Click);
//
// separator2
//
this.separator2.Index = 1;
this.separator2.Text = "-";
//
// cutMenuItem
//
this.cutMenuItem.Index = 2;
this.cutMenuItem.Shortcut = System.Windows.Forms.Shortcut.CtrlX;
this.cutMenuItem.Text = "Cut";
this.cutMenuItem.Click += new System.EventHandler(this.cutMenuItem_Click);
//
// copyMenuItem
//
//
this.aboutMenuItem.Index = 0;
this.aboutMenuItem.Text = "Dastur haqida";
this.aboutMenuItem.Click += new System.EventHandler(this.aboutMenuItem_Click);
```

```
            //
            // inputTextBox
            //
            this.inputTextBox.AcceptsReturn = true;
            this.inputTextBox.AcceptsTab = true;
            this.inputTextBox.AllowDrop = true;
            this.inputTextBox.BackColor = System.Drawing.Color.Gainsboro;
            this.inputTextBox.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;
            this.inputTextBox.Font = new System.Drawing.Font("Lucida Console", 9.75F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.inputTextBox.ForeColor = System.Drawing.Color.Blue;
            this.inputTextBox.Location = new System.Drawing.Point(0, 0);
            this.inputTextBox.Multiline = true;
            this.inputTextBox.Name = "inputTextBox";
            this.inputTextBox.ScrollBars = System.Windows.Forms.ScrollBars.Vertical;
            this.inputTextBox.Size = new System.Drawing.Size(634, 427);
            this.inputTextBox.TabIndex = 1;
            this.inputTextBox.TabStop = false;
            this.inputTextBox.DragDrop += new
System.Windows.Forms.DragEventHandler(this.inputTextBox_DragDrop);
            this.inputTextBox.DragEnter += new
System.Windows.Forms.DragEventHandler(this.inputTextBox_DragEnter);
            this.inputTextBox.TextChanged += new System.EventHandler(this.inputTextBox_TextChanged);
            //
            // fontDialog
            //
            this.fontDialog.Font = new System.Drawing.Font("Arial", 8.25F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.fontDialog.FontMustExist = true;
            this.fontDialog.ShowEffects = false;
            //
            // panel
            //
            this.panel.BackColor = System.Drawing.Color.Black;
            this.panel.BorderStyle = System.Windows.Forms.BorderStyle.Fixed3D;
            this.panel.Controls.Add(this.processingLabel);
            this.panel.Controls.Add(this.pictureBox);
            this.panel.ForeColor = System.Drawing.Color.Lime;
            this.panel.Location = new System.Drawing.Point(-1, 411);
            this.panel.Name = "panel";
            this.panel.Size = new System.Drawing.Size(200, 82);
            this.panel.TabIndex = 0;
```

```
                Application.DoEvents();
                        streamWriter.Close();
                }
        }

        private string openFile( string title, string filterString )
        {
                openFileDialog.FileName = "";
                openFileDialog.Title = title;
                openFileDialog.Filter = filterString;
                if( openFileDialog.ShowDialog() == DialogResult.OK )
                {
                        if( File.Exists( openFileDialog.FileName ) )
                        {
                                StreamReader streamReader = new StreamReader(
openFileDialog.FileName, true );
                                string fileString = streamReader.ReadToEnd();
                                streamReader.Close();
                                if( fileString.Length >= inputTextBox.MaxLength )
                                {
                                        MessageBox.Show( "XATO: \nSiz ochmoqchi bo'lgan fayl
matn muharririga sig'maydi.\nIltimos kichikroq fayl oching!\nAmal bekor qilindi!" );
                                        return null;
                                }
                                if( fileString != null )
                                {
                                        this.Text = GetFileName( openFileDialog.FileName ) + " -
RSA";
                                        currentFileName = openFileDialog.FileName;
                                }
                                return fileString;
                        }
                }
                return null;
        }

        private bool saveFile()
        {
                saveFileDialog.Title = "Save As";
                saveFileDialog.FileName = "*.txt";
                saveFileDialog.Filter = "Text Document( *.txt )|*.txt|All Files|*.*";
                if( saveFileDialog.ShowDialog() == DialogResult.OK )
```

```
        }

        private void mainForm_Load( object sender, EventArgs e )
        {
                if( File.Exists( "Settings.bin" ) )
                {
                        Settings settings = GetSettings();
                        this.Location = settings.Location;
                        this.Width = settings.Width;
                        this.Height = settings.Height;
                        inputTextBox.Font = settings.Font;
                        inputTextBox.WordWrap = settings.Wrapping;
                        wordWrapMenuItem.Checked = ( settings.Wrapping == true ) ? true : false;
                }
                this.Text = GetFileName( currentFileName ) + " - ASOSIY";
                inputTextBox.Focus();
                this.Resize += new System.EventHandler( this.mainForm_Resize );
                this.Move += new System.EventHandler( this.MainForm_Move );
                inputTextBox.Height = this.Size.Height - 53;
                inputTextBox.Width = this.Size.Width - 7;
        }

        private void mainForm_Resize( object sender, EventArgs e )
        {
                inputTextBox.Height = this.Size.Height - 53;
                inputTextBox.Width = this.Size.Width - 7;
                SaveSettings( "SIZE" );
        }

        private void MainForm_Move( object sender, EventArgs e )
        { SaveSettings( "LOCATION" ); }

        private void mainForm_FormClosing( object sender, FormClosingEventArgs e )
        {
                this.Resize -= new EventHandler( this.mainForm_Resize );
                this.Move -= new EventHandler( this.MainForm_Move );
                if( !cleanForm )
                {
        string dialogText = " " + currentFileName + " faylidagi matn  o'zgartirildi." + Environment.NewLine +
Environment.NewLine + "Saqlashni xohlaysizmi?";
```

```
                                switch( MessageBox.Show( dialogText, "RSA",
MessageBoxButtons.YesNoCancel ) )
                                {
                                        case DialogResult.Yes:
                                                {
                                                        if( !saveFile() ) { e.Cancel = true; }
                                                        break;
                                                }
                                        case DialogResult.Cancel:
                                                {
                                                        e.Cancel = true;
                                                        break;
                                                }
                                }
                        }
                }

                private void inputTextBox_TextChanged( object sender, EventArgs e )
                {
        if (currentFileName.Equals("RSA"))
                        { cleanForm = ( !inputTextBox.Text.Equals( "" ) ) ? false : true; }
                        else
                        { cleanForm = false; }
                }

                private void inputTextBox_DragEnter( object sender, DragEventArgs e )
                { e.Effect = DragDropEffects.All; }

                private void inputTextBox_DragDrop( object sender, DragEventArgs e )
                {
                        bool clean = false;
                        if( !cleanForm )
                        {
            string dialogText = " " + currentFileName + "faylidagi matn  o'zgartirildi." + Environment.NewLine +
Environment.NewLine + "Saqlashni xohlaysizmi?";
                                switch( MessageBox.Show( dialogText, "RSA",
MessageBoxButtons.YesNoCancel ) )
                                {
                                        case DialogResult.Yes:
                                                {
                                                        if( saveFile() ) { clean = true; }
                                                        break;
```

```
                                        }
                                case DialogResult.No:
                                        {
                                                clean = true;
                                                break;
                                        }
                                }
                        }
                        else
                        { clean = true; }
                        if( clean )
                        {
                                if( e.Data.GetDataPresent( DataFormats.FileDrop, true ) == true )
                                {
                                        string[] fileNames = ( string[] )e.Data.GetData( DataFormats.FileDrop,
true );
                                        StreamReader streamReader = new StreamReader( fileNames[ 0 ], true
);
                                        string fileContents = streamReader.ReadToEnd();
                                        streamReader.Close();
                                        currentFileName = fileNames[ 0 ];
                                        this.Text = GetFileName( fileNames[ 0 ] ) + " - RSA";
                                        SetText( fileContents );
                                        cleanForm = true;
                                }
                        }
                }

                private void newFileMenuItem_Click( object sender, EventArgs e )
                {
                        if( !cleanForm )
                        {
            string dialogText = " " + currentFileName + " faylidagi matn  o'zgartirildi." + Environment.NewLine +
Environment.NewLine + "Saqlashni xohlaysizmi?";
                                switch( MessageBox.Show( dialogText, "RSA",
MessageBoxButtons.YesNoCancel ) )
                                {
                                        case DialogResult.Yes:
                                                {
                                                        if( saveFile() )
                                                        {
                                                                SetText( "" );
```

```
                currentFileName = "RSA";
                                                this.Text = "RSA";
                                                cleanForm = true;
                                        }
                                        break;
                                }
                        case DialogResult.No:
                                {
                                        SetText( "" );
        currentFileName = "RSA";
                                        this.Text = "RSA";
                                        cleanForm = true;
                                        break;
                                }
                        case DialogResult.Cancel:
                                { break; }
                        default:
                                { break; }
                        }
                }
                else
                {
                        SetText( "" );
    currentFileName = "RSA";
                        this.Text = "RSA";
                }
        }

        private void openFileMenuItem_Click( object sender, EventArgs e )
        {
                if( !cleanForm )
                {
    string dialogText = " " + currentFileName + " faylidagi matn  o'zgartirildi." + Environment.NewLine +
Environment.NewLine + "Saqlashni xohlaysizmi?";
                        switch( MessageBox.Show( dialogText, "RSA",
MessageBoxButtons.YesNoCancel ) )
                        {
                                case DialogResult.Yes:
                                        {
                                                if( saveFile() ) { SetText( "" ); }
                                                else
                                                { return; }
```

```
                {
                        try
                        {
                                StreamWriter streamWriter = new StreamWriter( currentFileName );
                                streamWriter.Write( inputTextBox.Text );
                                streamWriter.Flush();
                                streamWriter.Close();
                                cleanForm = true;
                        }
                        catch( Exception Ex )
                        { MessageBox.Show( "ERROR: \n" + Ex.Message ); }
                }
        }

        private void exitMenuItem_Click( object sender, EventArgs e )
        {
                if( !cleanForm )
                {
        string dialogText = " " + currentFileName + " faylidagi matn  o'zgartirildi." + Environment.NewLine +
Environment.NewLine + "Saqlashni xohlaysizmi?";
                        switch( MessageBox.Show( dialogText, "RSA",
MessageBoxButtons.YesNoCancel ) )
                        {
                                case DialogResult.Yes:
                                        {
                                                if( saveFile() ) { Dispose( true ); }
                                                break;
                                        }
                                case DialogResult.No:
                                        {
                                                Dispose( true );
                                                break;
                                        }
                                case DialogResult.Cancel:
                                        { break; }
                                default:
                                        { break; }
                        }
                }
                else
                { Dispose( true ); }
```

```
        }

        private void undoMenuItem_Click( object sender, EventArgs e )
        { inputTextBox.Undo(); }

        private void cutMenuItem_Click( object sender, EventArgs e )
        {
                try { inputTextBox.Cut(); }
                catch( Exception Ex )
                { Console.WriteLine( Ex.Message ); }
        }

        private void copyMenuItem_Click( object sender, EventArgs e )
        {
                try { inputTextBox.Copy(); }
                catch( Exception Ex ) { Console.WriteLine( Ex.Message ); }
        }

        private void pasteMenuItem_Click( object sender, EventArgs e )
        {
                try { inputTextBox.Paste(); }
                catch( Exception Ex ) { Console.WriteLine( Ex.Message ); }
        }

        private void deleteMenuItem_Click( object sender, EventArgs e )
        {
                if( ( inputTextBox.Text.Length != 0 ) & ( inputTextBox.SelectionStart !=
inputTextBox.Text.Length ) )
                {
                        if( !inputTextBox.SelectedText.Equals( "" ) )
                        {
                                try
                                {
                                        int cursorPosition = inputTextBox.SelectionStart;
                                        inputTextBox.Text = inputTextBox.Text.Remove(
cursorPosition, inputTextBox.SelectedText.Length );
                                        inputTextBox.SelectionStart = cursorPosition;
                                }
                                catch( Exception Ex ) { Console.WriteLine( Ex.Message ); }
                        }
                        else
                        {
```

```
                                        try
                                        {
                                                int cursorPosition = inputTextBox.SelectionStart;
                                                inputTextBox.Text = inputTextBox.Text.Remove(
cursorPosition, 1 );
                                                inputTextBox.SelectionStart = cursorPosition;
                                        }
                                        catch( Exception Ex ) { Console.WriteLine( Ex.Message ); }
                                }
                        }
                }

                private void selectAllmenuItem_Click( object sender, EventArgs e )
                {
                        try { inputTextBox.SelectAll(); }
                        catch( Exception Ex ) { Console.WriteLine( Ex.Message ); }
                }

                private void wordWrapMenuItem_Click( object sender, EventArgs e )
                {
                        if( wordWrapMenuItem.Checked == true )
                        {
                                wordWrapMenuItem.Checked = false;
                                inputTextBox.WordWrap = false;
                        }
                        else
                        {
                                wordWrapMenuItem.Checked = true;
                                inputTextBox.WordWrap = true;
                        }
                        SaveSettings( "WRAPPING" );
                }

                private void fontMenuItem_Click( object sender, EventArgs e )
                {
                        if( fontDialog.ShowDialog() == DialogResult.OK )
                        {
                                inputTextBox.Font = fontDialog.Font;
                                SaveSettings( "FONT" );
                        }
                }
```

```
        private void decryptMenuItem_Click( object sender, EventArgs e )
        {
            if( inputTextBox.Text.Length != 0 )
            {
                openFileDialog.FileName = "";
                openFileDialog.Title = "Open Private Key File";
                openFileDialog.Filter = "Private Key Document( *.kez )|*.kez";
                string fileString = null;
                if( openFileDialog.ShowDialog() == DialogResult.OK )
                {
                    if( File.Exists( openFileDialog.FileName ) )
                    {
                        StreamReader streamReader = new StreamReader(
openFileDialog.FileName, true );
                        fileString = streamReader.ReadToEnd();
                        streamReader.Close();
                        if( fileString.Length >= inputTextBox.MaxLength )
    { MessageBox.Show("XATO: \nSiz ochmoqchi bo'lgan fayl matn muharririga sig'maydi.\nIltimos
kichikroq fayl oching!\nAmal bekor qilindi!"); }
                    }
                }
                if( File.Exists( openFileDialog.FileName ) )
                {
                    string bitStrengthString = fileString.Substring( 0, fileString.IndexOf(
"</BitStrength>" ) + 14 );
                    fileString = fileString.Replace( bitStrengthString, "" );
                    int bitStrength = Convert.ToInt32( bitStrengthString.Replace(
"<BitStrength>", "" ).Replace( "</BitStrength>", "" ) );
                    Point point = new Point( ( inputTextBox.Size.Width / 2 ) - (
panel.Size.Width / 2 ), ( inputTextBox.Size.Height / 2 ) - ( panel.Size.Height / 2 ) );
                    panel.Location = point;
                    panel.Visible = true;
                    this.Refresh();
                    fileMenuItem.Enabled = false;
                    editMenuItem.Enabled = false;
                    formatMenuItem.Enabled = false;
                    encryptionMenuItem.Enabled = false;
                    helpMenuItem.Enabled = false;
                    string tempStorage = inputTextBox.Text;
                    if( fileString != null )
                    {
```

```
                                        FinishedProcessDelegate finishedProcessDelegate = new
FinishedProcessDelegate( FinishedProcess );
                                        UpdateTextDelegate updateTextDelegate = new
UpdateTextDelegate( UpdateText );
                                        try
                                        {
                                                EncryptionThread decryptionThread = new
EncryptionThread();
                                                Thread decryptThread = new Thread(
decryptionThread.Decrypt );
                                                decryptThread.IsBackground = true;
                                                decryptThread.Start( new Object[] { this,
finishedProcessDelegate, updateTextDelegate, inputTextBox.Text, bitStrength, fileString } );
                                        }
                                        catch( CryptographicException CEx )
                                        { MessageBox.Show( "ERROR: \nOne of the following has
occured.\nThe cryptographic service provider cannot be acquired.\nThe length of the text being encrypted is greater
than the maximum allowed length.\nThe OAEP padding is not supported on this computer.\n" + "Exact error: " +
CEx.Message ); }
                                        catch( Exception Ex )
                                        {
                                                MessageBox.Show( "ERROR:\n" + Ex.Message );
                                                SetText( tempStorage );
                                        }
                                }
                        }
                }
                else
                { MessageBox.Show( "XATO: Siz mavjud bo'lmagan ma'lumotni deshifrlay olmaysiz!!!"
); }
        }

        private void encryptMenuItem_Click( object sender, EventArgs e )
        {
                if( inputTextBox.Text.Length != 0 )
                {
                        openFileDialog.FileName = "";
                        openFileDialog.Title = "Open Public Key File";
                        openFileDialog.Filter = "Public Key Document( *.pke )|*.pke";
                        string fileString = null;
                        if( openFileDialog.ShowDialog() == DialogResult.OK )
                        {
```

```
                                        if( File.Exists( openFileDialog.FileName ) )
                                        {
                                                StreamReader streamReader = new StreamReader(
openFileDialog.FileName, true );
                                                fileString = streamReader.ReadToEnd();
                                                streamReader.Close();
                                                if( fileString.Length >= inputTextBox.MaxLength )
                { MessageBox.Show("XATO: \nSiz ochmoqchi bo'lgan fayl matn muharririga sig'maydi.\nIltimos
kichikroq fayl oching!\nAmal bekor qilindi!"); }
                                        }
                                }
                                if( fileString != null )
                                {
                                        FinishedProcessDelegate finishedProcessDelegate = new
FinishedProcessDelegate( FinishedProcess );
                                        UpdateTextDelegate updateTextDelegate = new UpdateTextDelegate(
UpdateText );
                                        string bitStrengthString = fileString.Substring( 0, fileString.IndexOf(
"</BitStrength>" ) + 14 );
                                        fileString = fileString.Replace( bitStrengthString, "" );
                                        int bitStrength = Convert.ToInt32( bitStrengthString.Replace(
"<BitStrength>", "" ).Replace( "</BitStrength>", "" ) );
                                        Point point = new Point( ( inputTextBox.Size.Width / 2 ) - (
panel.Size.Width / 2 ), ( inputTextBox.Size.Height / 2 ) - ( panel.Size.Height / 2 ) );
                                        panel.Location = point;
                                        panel.Visible = true;
                                        this.Refresh();
                                        fileMenuItem.Enabled = false;
                                        editMenuItem.Enabled = false;
                                        formatMenuItem.Enabled = false;
                                        encryptionMenuItem.Enabled = false;
                                        helpMenuItem.Enabled = false;
                                        if( fileString != null )
                                        {
                                                try
                                                {
                                                        EncryptionThread encryptionThread = new
EncryptionThread();
                                                        Thread encryptThread = new Thread(
encryptionThread.Encrypt );
                                                        encryptThread.IsBackground = true;
```

```
                                                encryptThread.Start( new Object[] { this,
finishedProcessDelegate, updateTextDelegate, inputTextBox.Text, bitStrength, fileString } );
                                                }
                                                catch( CryptographicException CEx )
                                                { MessageBox.Show( "ERROR: \nOne of the following has
occured.\nThe cryptographic service provider cannot be acquired.\nThe length of the text being encrypted is greater
than the maximum allowed length.\nThe OAEP padding is not supported on this computer.\n" + "Exact error: " +
CEx.Message ); }
                                                catch( Exception Ex )
                                                { MessageBox.Show( "ERROR: \n" + Ex.Message ); }
                                        }
                                }
                        }
                        else
        { MessageBox.Show("XATO: Siz mavjud bo'lmagan ma'lumotni shifrlay olmaysiz!!!"); }
                }

                private void generateKeyPairMenuItem_Click( object sender, EventArgs e )
                {
                        KeyPairGeneratorForm generator = new KeyPairGeneratorForm();
                        if( generator.ShowDialog() == DialogResult.OK )
                        {
                                RSACryptoServiceProvider RSAProvider = new RSACryptoServiceProvider(
currentBitStrength );
                                string publicAndPrivateKeys = "<BitStrength>" + currentBitStrength.ToString()
+ "</BitStrength>" + RSAProvider.ToXmlString( true );
                                string justPublicKey = "<BitStrength>" + currentBitStrength.ToString() +
"</BitStrength>" + RSAProvider.ToXmlString( false );
                                if( saveFile( "Save Public/Private Keys As", "Public/Private Keys Document(
*.kez )|*.kez", publicAndPrivateKeys ) )
                                { while( !saveFile( "Save Public Key As", "Public Key Document( *.pke
)|*.pke", justPublicKey ) ) { ; } }
                        }
                }

                private void aboutMenuItem_Click( object sender, EventArgs e )
                {
                        AboutForm aboutRSACryptoPad = new AboutForm();
                        aboutRSACryptoPad.ShowDialog( this );
                }
        }
}
```