УНИВЕРЗИТЕТ У БЕОГРАДУ

МАТЕМАТИЧКИ ФАКУЛТЕТ

ТЕОРИЈА КОДИРАЊА ПОДАТАКА У СРЕДЊОШКОЛСКОЈ НАСТАВИ МАТЕМАТИКЕ

Мастер рад

Ивана Живанић

Београд,

Октобар 2015.

УНИВЕРЗИТЕТ У БЕОГРАДУ

МАТЕМАТИЧКИ ФАКУЛТЕТ

МАСТЕР РАД

Тема: Теорија кодирања података у средњошколској настави математике

Кандидат: Ивана Живанић

Ментор: др Небојша Икодиновић, Математички факултет у Београду

Чланови комисије:

проф. др Зоран Петровић, Математички факултет у Београду

др Филип Марић, Математички факултет у Београду

Датум одбране: _____

Садржај

УВОД

Често изговарана реченица, али ретко кад поткрепљена „Математика има огромне примене”, била је почетна идеја овог рада. Наставници излажући тешко и сувопарно математичко градиво често се зауставе на овој реченици, не дајући ученицима ни наговештај њене примене. Нарочито су данашње генерације ученика жељне занимљивих лекција, али су уједно и смеле да питају „За шта ће то мени у животу?“

Тема које је обрађена у овом раду као ни добар део математичког апарата још увек нису део градива гимназија, али је тема актуелна и ученици су се сигурно сусретали са њом кроз свакодневни живот. Мада, део изложеног (бројевне репрезентације, представљање података у рачунару) представља информатичко градиво. Може се успоставити корелација између ова два наставна предмета, па су за поједине алгоритме дати псеудокодови како би се показало да се могу испрограмирати и олакшати бесмислено рачунање. Рад је такође покушај спајања царства бројева са виртуелним светом у коме ће на систематичан начин бити издвојене целине које се као такве могу уводити у наставу, било као занимљиво поткрепљење часовима математике или као проширење математичког знања. Циљ рада није да детаљно и строго математички изложи одговарајуће садржаје, већ да кроз интересантне задатке и примере приближи ученицима материју, и да истовремено покаже значај и снагу математике.

Било да су наставници заговорници увођења информационих технологија у наставу или не, овај рад представља могућност да се открије делић математичке основе „виртуелног света”. Са друге стране рад отвара видике ученицима и проширује свест да уколико желе да се баве информационим технологијама, у њиховом оснаживању или слабљењу, неопходно је да познају математику.

Рад се састоји из пет поглавља, од којих прво и пето поглавље могу бити испричани најширем кругу ученика као занимљива прича са лаким примерима и задацима који не захтевају велико математичко предзнање. Такође, пето поглавље садржи више мотивационих прича које могу послужити наставницима као идеја шта све може бити обрађено на часовима математике. Почетно поглавље говори о криптографији кроз историју и једноставним методама шифровања. У овом, и наредна три, обрађени су примери и задаци који илуструју изложен материјал. У другом поглављу је реч о бројевним системима, прво кроз излагање математичке теорије, а потом као примена бројевних система показано је представљање података у рачунару. Треће поглавље почиње са модуларном аритметиком и конгруенцијама, а завршава се применом изложеног кроз бар-кодове и математичка шифровања. Четврто поглавље је математички најзахтевније и у њему је обрађена Ојлерова функција која се користи у РСА алгоритму, који се нарочито у употреби за заштиту података на интернету (сигурносни протоколи, системи електронског пословања укључујући и електронски потпис итд) .

I КРАТКА ИСТОРИЈА КРИПТОГРАФИЈЕ

Криптографијом се може обогатити настава математике. Многи примери из криптографије могу послужити да би се разбиле предрасуде о томе да је сваки математички проблем решив помоћу праве формуле или доброг рачунара. Наиме, сигурност криптосистема лежи управо у нашој немогућности да брзо и ефикасно решимо одговарајући проблем из подручја алгебре, теорије бројева или комбинаторике. Занимљиве методе шифровања, нарочито откривања метода за разбијање система за шифрирање, омогућиће ученицима да осете снагу и лепоту математике.

1. Основно о криптографији

Замислимо да је потребно да две особе размењују информације на даљину, али тако да буду сигурни да нико трећи неће доћи до тих информација (некада су се те информације размењивале у облику писама, а данас углавном у електронској форми, али је суштина заштите остала иста). Аналогија за безбедно слање инфрмација јесте: Лазар ставља своје писмо у кутију која ће бити закључана катанцем и тако закључану кутију шаље ка Милици. Милица добивши кутију, зна шифру за откључавање катанца тако да ће моћи да отвори кутију и прочита писмо. Ако на путу од Лазара до Милице кутија заврши у рукама треће особе, она ће бити лишена информација јер не зна шифру за отварање катанца. Процес закључавања кутије односно информација назива се шифровање (енкрипција), док се процес отварања кутије назива дешифровање (декрипција). Криптографија закључава поверљиве информације виртуелним катанцом и неке од једноставнијих техника биће показане у овом поглављу.

Следе дефиниције (не сасвим формалне) основних појмова из криптографије.

Криптографија је вештина и наука чувања безбедности порука. Назив криптографија потиче од грчке речи *kriptos*, што значи скривен. Она омогућава пошиљалац сигурно пошаље своју поруку примаоцу, тако да непозвана трећа страна, нападач не може да дође до њеног садржаја.

Порука која се шаље назива се још и **отворени текст** и представља информацију у било ком облику (текстуални документ, низ битова, дигитални запис итд.).

Класична криптоанализа се односи на алгоритме који шифрују текстуалне поруке.

Шифровање је процес маскирања поруке, који има за циљ скривање њене садржине. Шифрована порука се назива **шифрат**. **Дешифровање** је процес враћања шифроване поруке у отворени текст.

Криптоанализа је наука разбијања и читања шифрованих порука. Покушај криптоанализе назива се напад, а успешна криптоанализа назива се **декриптовање**. Циљ криптоанализе јесте проналажење слабости дате криптографске шеме у циљу њеног разбијања. [1]

2. Криптографија кроз историју

Криптографија датира скоро од самих почетака цивилизације, а свеприсутна је и данас. Некада се шифровало помоћу једноставних направа, потом помоћу механичких машина, а данас помоћу рачунара. У овом поглављу биће показано неколико занимљивих метода шифрирања које су постојале кроз историју, а данас се не користе, али могу послужити као врло занимљив увод и као приказ развоја математичке мисли кроз криптографију.

2.1. Спартанска шифра Скитале

Спартанци су први увели систем војне криптографије. Наиме, конструисали су уређај **скитале** (пети век п.н.е.) који се заснивао на једној врсти шифре премештања. Скитале је дрвени штап око кога је обмотана трака од коже. Пошиљалац исписује поруку дуж штапа, а затим одмотава траку која наизглед носи бесмислени низ слова. Порука је шифрована, јер ју је једино могуће прочитати ако се делови текста поставе у правилном редоследу.

Слика 1. Скитале

У овом примеру кључ који закључава поруку јесте дебљина штапа око кога се намотава трака по којој ће бити исписана порука.

2.2. Цезарова шифра

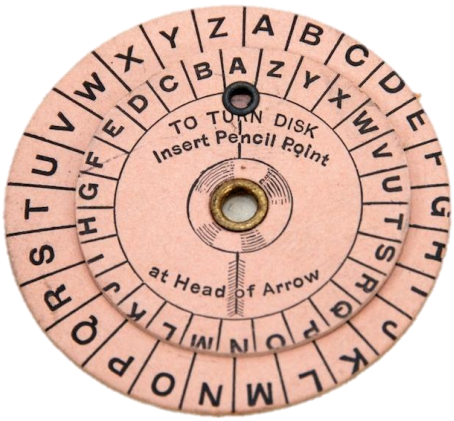
Прва шифра, призната у данашњем смислу је шифра замене и она долази од Римљана, названа по чувеном императору Цезару – Цезарова шифра. То је тип шифре где се свако слово отвореног текста мења одговарајућим словом азбуке, помереним за одређени број места, с тим што се претпоставља се да се алфабет циклично наставља, тј. након последњег слова Z долазе слова A, B, C… Цезар је користио шифру са кључем (померајем) три код које се слова отвореног текста мењају словима која се налазе три места даље од њих у алфабету ($A \to D, B \to E$ итд.). Цезарова изрека VENI VIDI VICI шифрована кључем који представља померај три била би ZHRM ZMGM ZMFM (подразумева се да је скуп знакова латински алфабет од 24 слова).

Приликом шифровања поруке, за различите кључеве потребно је утврдити како ће изгледати нова азбука, што се може урадити исписивањем азбуке без помераја и азбуке са померајем у табели од два реда.

Табела 1. Табела азбуке и азбуке шифровање кључем (померајем) 3

А	Б	В	Г	Д	Ђ	Е	Ж	З	И	Ј	К	Л	Љ	М	Н	Њ	О	П	Р	С	Т	Ћ	У	Ф	Х	Ц	Ч	Џ	Ш
Д	Ђ	Е	Ж	З	И	Ј	К	Л	Љ	М	Н	Њ	О	П	Р	С	Т	Ћ	У	Ф	Џ	Ц	Ч	Џ	Ш	А	Б	В	Г

Исто се може одредити коришћењем једноставне направе која се састоји од два кружна прстена од којих се унутрашњи помера, тако да је помоћу ње шифровање врло једноставно (*Слика 2*). Точак за шифровање може се врло једноставно направити од картона, што ученици могу самостално даураде.



Слика 2. Точак за шифровање

Следе примери шифровања порука Цезаровим методом.

Задатак 1. Шифровати, односно дешифровати Цезаровим методом са кључем $k = 3$ речи дате у табелама:

а) Шифровати

О	Б	Л	А	К

б) Дешифровати

Ф	Ч	Р	А	Ј

Решење:

На основу табеле 1. или уз помоћ точка за шифровање може се закључити да слову О одговара слово Т шифроване азбуке, слову Б слово Ђ итд. На основу чега решење гласи:

О	Б	Л	А	К
Т	Ђ	Њ	Д	Н

Аналогно се решава други део задатка, само што се полази од шифроване азбуке. Шифрованом слову Ф одговара слово С, слову Ч одговара слово У итд. Долази се до речи:

С	У	Н	Ц	Е
Ф	Ч	Р	А	Ј

Задатак 2. Дешифровати одговор загонетке.

1. Пуна школа ђака ниоткуда врата. (Кључ је 3)

Њ	Ч	Ђ	?	Р	Љ	А	?

2. Гураво прасе све пуље опасе. (Кључ је 10)

А	Е	И	Р

Решење:

1. На исти начин као и првом задатку долази се до

Л	У	Б		Н	И	Ц	
Њ	Ч	Ђ		Р	Љ	А	

Остатак речи се наслућује или ко зна одговор загонетке лако може рећи која су преостала слова. Дакле, одговор је: ЛУБЕНИЦА.

2. Да би се решио овај део задатка потребно је направити табелу шифровања са померајем 10 (или још лакше коришћењем точка за шифровање). Односно, решење је:

К	О	С	А
А	Е	И	Р

У то време сматрало се да је због великог броја могућих кључева шифра замене нерешива, што је током дугог низа векова било тачно. Међутим, захваљујући Арапима који су унапредили своје познавање математике, лингвистике и статистике, и који се сматрају оснивачима криптоанализе, Цезарова шифра (подразумева се шифра чији је кључ било који померај) је лако разбијена. Наизглед бескорисно проучавање учесталости појављивања неких слова довело је до првог великог открића у криптоанализи – **анализе учесталости слова неког природног језика**. Данас, захваљујући анализи учесталости, није потребно проверавати све могуће кључеве, већ је садржај поруке моуће открити на основу учесталости појављивања слова.

Један од метода дешифровању без познатог кључа јесте употреба „грубе силе“, односно испитивање свих могућих кључева редом све док се не добије неки смислени текст, што може бити веома напорно.

Пример 1. Нека шифровани текст гласи: ОЦЕКЦ, тада дешифровањем за различите вредности кључева добијамо следеће речи:

ПШЖЉШ за $k = 1$,

РЏЗЛЏ за $k = 2$,

СЧИМЧ за $k = 3$,

ТАЈНА за $k = 4$,

ЋБКЊБ за $k = 5$,

УВКОВ за $k = 6$,

ФГЛПГ за $k = 7$,

ХДЉРД за $k = 8$,

ЦЂМСЂ за $k = 9$,

ЧЕНТЕ за $k = 10$,

ЏЖЊЋЖ за $k = 11$,

ШЗОУЗ за $k = 12$,

АИПФИ за $k = 13$,

БЈРХЈ за $k = 14$,

ВКСЦК за $k = 15$,

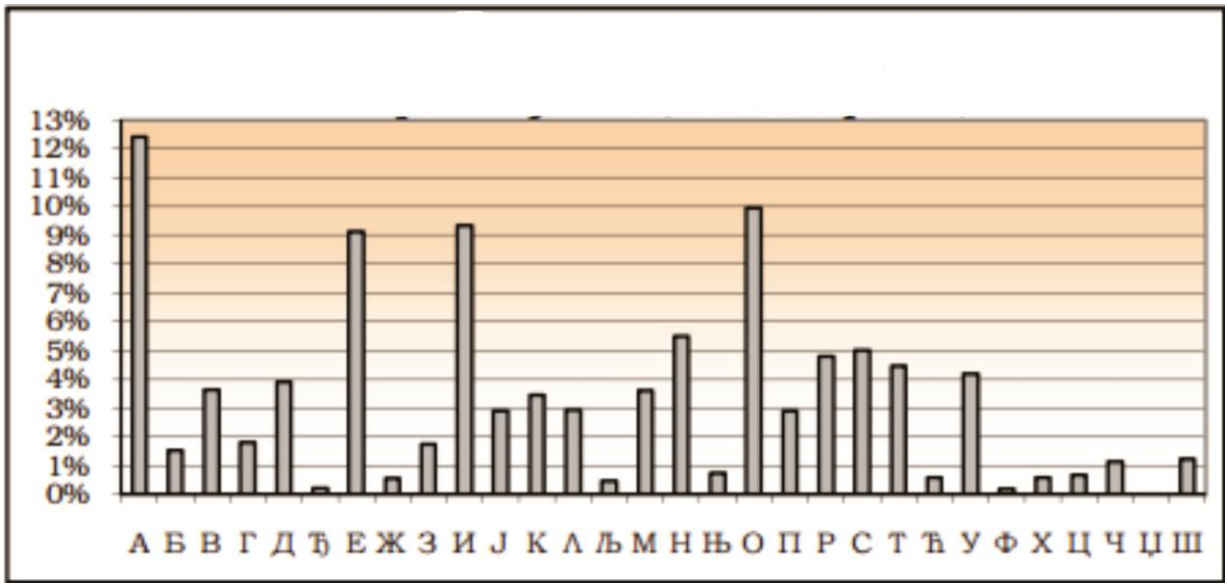
ГЛТЧЛ за $k = 16$,

ДЉЋЏЉ за $k = 17$,

ЂМУШМ за $k = 18$...

на основу чега закључујемо да је тражена реч ТАЈНА.

Други начин подразумева коришћење анализе учесталости појављивања слова у природном језику. Да би се шифра разбила коришћењем другог метода мора бити познато на ком језику је написан шифровани текст. На слици 3. графички је представљена релативна учесталост слова у српском језику.



Слика 3.Учестанолост појављивања слова у текстовима на српском језику *[1]*

Следећи задатак биће решен помоћу фреквенцијске анализе слова српског језика.

Задатак 3. Дешифровати реченицу написану српским језиком и шифровану Цезаровим методом.

ЏМЂСЏМЂУХМ РС ЏГМЧУЈМ У ДЏЖОМ.

Решење:

Пошто је најфреквентније слово у српском језику слово А, претпоставимо да је најфреквентније слово шифрованог текста М управо слово А. То би значило да је реченица шифрована кључем са вредношћу 14, односно да оригинална порука гласи:

МАТЕМАТИКА ЈЕ КРАЉИЦА И СЛУГА.

У случају да претпоставка о томе које је најфреквентније слово шифрованог текста била погрешна, требало би покушати са следећим по учесталости појављивања словом српског језика (И, Е, О, У, итд).

2.3. Нематематичка шифровања[1]

Поступци који не укључују директно математику, а шифровања врше на занимљиве начине: помоћу кључне речи, квадрата, датума, названа су нематематичким шифровањем.

2.3.1. Шифровање кључном речи

Прво се одабере кључна реч (било која састављена од слова која чине нешифровану поруку). Кључна реч се напише испод низа слова који чине азбуку, а затим се испод осталих слова азбуке дописују редом слова изузев оних која су искоришћења у кључној речи.

Пример 2. Милица жели да пошаље поруку ОЧИ ЗБОРЕ ШТО ИМ СРЦЕ ВЕЛИ Лазару са кључном речи ЗЕМЉА.

Сада се прави шифрована азбука на следећи начин:

Азбука:	А	Б	В	Г	Д	Ђ	Е	Ж	З	И	Ј	К	Л	Љ	М	Н	Њ	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Џ	Ш
Шифрована азбука:	З	Е	М	Љ	А	Б	В	Г	Д	Ђ	Ж	И	Ј	К	Л	Н	Њ	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Џ	Ш

На основу добијеног шифрованог писма, порука ће гласити:

ОЧЂ ДЕОРВ ШТО ЂЛ СРЦВ МВЈЂ

Да би Лазар могао да дешифрује поруку потребно је да зна кључну реч. Процес дешифровања би изгледао:

Формира се шифрована азбука на основу добијене кључне речи, а затим се испод ње исписује азбука како би се одредили парови слова шифроване азбуке и азбуке

[1] Термин преузет из [13]

Шифрована азбука: З Е М Љ А Б В Г Д Ђ Ж И Ј К Л Н Њ О П Р С Т У Ф Х Ц Ч Џ Ш

Азбука: А Б В Г Д Ђ Е Ж З И Ј К Л Љ М Н Њ О П Р С Т У Ф Х Ц Ч Џ Ш

Сада се види којим словима азбуке одговарају слова шифроване азбуке, односно послата порука гласи

ОЧИ ЗБОРЕ ШТО ИМ СРЦЕ ВЕЛИ

2.3.2. Шифровање методом квадрата

Шифровање се врши тако што се порука пише у квадратној табели (или више њих), што ће на примеру изгледати:

Задатак 4. Поруку У ДОБРУ ЈЕ ЛАКО ДОБАР БИТИ НА МУЦИ СЕ ПОЗНАЈУ ЈУНАЦИ шифровати методом квадрата.

У	Д	О	Б	Р	У	Ј
Е	Л	А	К	О	Д	О
Б	А	Р	Б	И	Т	И
Н	А	М	У	Ц	И	С
Е	П	О	З	Н	А	Ј
У	Ј	У	Н	А	Ц	И
А	А	А	А	А	А	А

Порука је редом спакована у кадрат, а у поља која су остала празна уписно је слово А.

Шифрована порука се добија редом исписивањем слова једне по једне колоне, одоздо нагоре.

АУЕНБЕУАЈПАЛДАУОМРАОАНЗУБКБААНЦИОРАЦАИТДУАИЈСИОЈ

Лазар је могао да испише слова редоседом колона којим је желео, једино што Милица треба да зна да би дешифровала поруку јесте величина квадрата и редослед којим су записиване колоне.

Порука се може уписати и у правоугаоник или неку другу геометријску слику, суштина шифровања ће остати иста. Као што је Задатку 4. приказано уколико нема довољно слова како би се попунила геометријска слика, у том случају се празна поља попуњавају неким од карактера по договору (у задатку 4. то је било слово А).

2.3.3. Шифровање датумом

Као што сам назив каже порука је одређена датумом, и то баш датумом када је шифрована.

Пример 4. Лазат шаље поруку ШТО ЉУДИ ХОЋЕ ТО И ВРЕМЕ ТРПИ Милици дана 30. августа 2008. Прво што је потребно јесте да испише датум, понављајући га колико је потребно, а испод њега текст који се шифрује.

3	0	0	8	2	0	0	8	3	0	0	8	2	0	0	8	3	0	0	8	2	0	0
Ш	Т	О	Љ	У	Д	И	Х	О	Ћ	Е	Т	О	И	В	Р	Е	М	Е	Т	Р	П	И

Следи додавање бројева који се налазе изнад слова на нумеричку вредност сваког слова (слово А-1, Б-2, итд.). Тако, на пример, нумеричка вредност слова Ш биће $3 + 30 = 33$, али пошто нема 33 слова одузима се од добијеног резултата 30, тако да је крајњи резултат 3. Овакво сабирање представља сабирање по модулу 30, о чему ће касније бити речи.

На основу претходног добија се низ бројева:

3	22	18	22	26	5	10	4	21	23	7	30	20	10	3	28	10	15	7	30	22	18	10

Сада се уместо нумеричке вредности испишу слова која ће чинити шифровану поруку. Дакле, прво слово шифроване поруке је треће слово азбуке, а то је В, следеће је двадесет друго слово азбуке – Т итд. Најзад, шифрована порука гласи:

ВТОСХДЧГСЋЕШРИВЧИМЕШТОИ

Дешифровање се врши једноставно уколико се зна датум када је порука шифрована. Креће се од добијене поруке испод које се испишу нумеричке вредности слова. Након тога се испише цео ред са датумом када је порука послата и од првог реда нумеричких вредности одузимају се вредности датума. Добијени бројеви представљају нумеричке вредности слова оригиналне поруке.

3	22	18	22	26	5	10	4	21	23	7	30	20	10	3	28	10	15	7	30	22	18	10
3	0	0	8	2	0	0	8	3	0	0	8	2	0	0	8	3	0	0	8	2	0	0

Што је када се одузме:

30	22	18	13	24	5	10	26	18	23	7	22	18	10	3	20	7	15	7	22	20	18	10

На крају се превођењем нумеричке форме у слова добија:

ШТОЉУДИХОЋЕТОИВРЕМЕТРПИ

Остало је још да се објасни како је $3 - 3 = 30$ и $4 - 8 = 26$. Пошто нумеричке вредности слова крећу од 1, а $3 - 3 = 0$, онда се позајмљује 30 (толико има слова у азбуци). Исто важи за други случај. Пошто би се добио негативан број, позајмљивањем се рачун своди на $34 - 8 = 26$.

Ако би се са математичке стране размотрили изложени једноставни шифарски системи могло би се доћи до одређених закључака. Зато ће бити уведене неке ознаке.

Нека је P скуп могућих отворених текстова. То може да буде скуп слова, као што је био у разматраним примерима, и нека је C скуп могућих шифрата. **Шифарска трансформација** је једнозначна функција која пресликава P у C (f не сме да пресликава два различита отворена текста у исти шифрат). Пар трансформација $f: P \longrightarrow C$ и $f^{-1}: C \longrightarrow P$ је шифарски ситем.

Најједноставнија трансформација је **транслација.** Нека P означава слово отвореног текста (односно одговарајући број) $\text{А} = 1, \text{Б} = 2, \ldots \text{Ш} = 30$. Шифровање се описује једнакошћу $C \equiv (P + b)\ mod\ n$, где је b кључ, а n број слова у азбуци; због тога дешифровање описује једнакост $P \equiv (C - b)\ mod\ n$. Цезарова шифра важи за $b = 3$ и $n = 30$. Шифровање датумом и кључном речи такође спадају у транслацију, само што се различитим функцијама пресликавају слова отвореног текста. Функције су код ових шифрирања задате датумом односно кључном речи.

Шифровање методом квадрата може се класификовати у шифарски систем који ради са појединачним словима, где се слова замењују другим словима, при чему је коришћен квадрат ради пермутације. Такође, може бити задата и произвољна премутација, рецимо $\text{А} \longrightarrow \text{Ф}$, $\text{Б} \longrightarrow \text{Л}$, ... Затим, могуће је пемутацију задати таблицом или изразом који описује процес шифровања, односно дешифровања.

Ако се отворени текст и шифрат представе као низ слова, псеудокод за шифрирање са померајем (транслацију) би могао да изгледа:

begin

$sifrat\ = '';$

for $i = 0$ ***to*** $duzina(otvoreniTekst)$

begin

$slovo = brojevnaReprezentacija(otvoreniTekst[i]);$

$sifrovanoSlovo\ = (slovo\ +\ pomeraj)\ mod\ 30;$

$sifrat = sifrat\ +\ sifrovanoSlovo;$

(овде операција сабирања представља надовезивање речи)

end

end

II БРОЈЕВНИ СИСТЕМИ

У оквиру првог дела овог поглавља биће изложена основна теорија о бројевним системима, како би се у другом делу разумео начин на који се кодирају подаци у рачунару.

1. Основно о бројевним системима

Људи најчешће користе бројевни систем са основом десет. Највероватније је десет као основа система настала на основу анатомије људске шаке, јер су користили својих десет прстију за бројање. Свакодневно се користи и систем са основом дванаест – дванаест месеци у години и дванаест подеока на часовнику. Често се броји и по шездесет – сваки пут када се мери неки угао ($1° = 60', 1' = 60''$) и сваки пут када се мери време, јер један сат има шездесет минута, а један минут шездесет секунди.

Неки народи броје по четири јер постоје четири размака међу прстима једне шаке. Други броје по осам из истог разлога уколико укључе и другу шаку [2].

Компоненте било ког позиционог бројног система су **основа – база** и **цифре**. Бројевни запис облика $(a^n a^{n-1} \dots a^0)_b$, где је b природан број већи од 1, који се назива основа, и a^i бројеви такви да је $1 \leq a^i \leq b$, који се називају цифре, има вредност $a^n b^n + a^{n-1} b^{n-1} + \cdots + a^1 b^1 + a$.

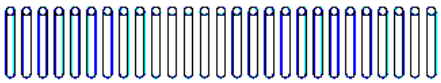
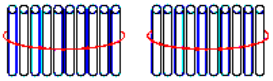
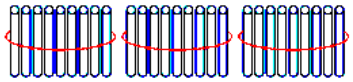
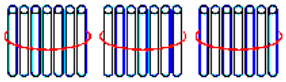
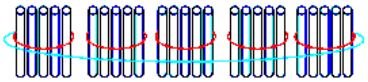
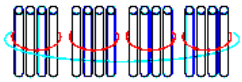
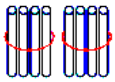
Цифре су симболи које могу имати вредност од нуле до вредности основе умањена за један. Треба обратити пажњу да системи са основом већом од 10 захтевају употребу слова (или неких других симбола) јер за представљање бројева у тим системима није довољно десет цифара. На пример, слово A у хексадецималном бројевном систему представља број 10, односно у овом систему цифру десет. Следи део који је углавном преузет из [3], где се говори о представљању бојева у различитим бројевним системима.

Могућност да представимо бројеве у позиционом систему дугујемо теореми о остатку. Декадни бројни систем заснован је на јединственом приказивању бројева збиром производа цифара и степена броја 10 који у овом случају представља основу система. На пример,

$$7\,203 = 7 \cdot \mathbf{10}^3 + 2 \cdot \mathbf{10}^2 + 0 \cdot \mathbf{10}^1 + 3 \cdot \mathbf{10}^0 .$$

Уместо броја 10, за базу бројног система се може изабрати било који други природан број већи од 1.

Табела 1. Представљање бројева у различитим базама

Број: 				Запис броја у датом систему
База b	b^2	b^1	b^0	
$b=10$				27
$b=9$				30
$b=7$				36
$b=5$				102
$b=4$				123

Записи у последњој колони табеле имају редом следећа значења

$b = 10$: $27 = 2 \cdot \mathbf{10}^1 + 7 \cdot \mathbf{10}^0$

$b = 9$: $30 = 3 \cdot \mathbf{9}^1 + 0 \cdot \mathbf{9}^0$

$b = 7$: $36 = 3 \cdot \mathbf{7}^1 + 6 \cdot \mathbf{7}^0$

$b = 5$: $102 = 1 \cdot \mathbf{5}^2 + 0 \cdot \mathbf{5}^1 + 2 \cdot \mathbf{5}^0$

$b = 4$: $123 = 1 \cdot \mathbf{4}^2 + 2 \cdot \mathbf{4}^1 + 3 \cdot \mathbf{4}^0$

За записивање бројева у систему са базом b обично се користи b цифара (цифре заправо представљају ознаке за бројеве $0, 1, \ldots, b-1$) и сваки број се на јединствен начин може приказати као збир производа цифара и степена броја b.

Табела 2. Цифре различитих бројевних система

ОСНОВА	**ЦИФРЕ**
9	**0, 1, 2, 3, 4, 5, 6, 7, 8**
8	**0, 1, 2, 3, 4, 5, 6, 7**
7	**0, 1, 2, 3, 4, 5, 6**
6	**0, 1, 2, 3, 4, 5**
5	**0, 1, 2, 3, 4**
4	**0, 1, 2, 3**
3	**0, 1, 2**
2	**0, 1**

Теорема 1. Нека је $n > 1$. Тада за сваки природан број a постоје и јединствено су одређени бројеви p из $\boldsymbol{N}$ и $a_0, a_1, a_2, \dots, a_n$ из $\boldsymbol{N}_0$ такви да је $a = a_p n^p + \cdots + a_2 n^2 + a_1 n^1 + a_0$ (1), при чему је $0 \le a_0, a_1, a_2, \dots, a_{p-1} < n$ и $1 \le a_p < b$.

Доказ

Егзистенција. Нека су са M означени сви природни бројеви за које важи теорема. Очигледно је да важи $1 \in M$, јер 1 има облик (1) за $p = 0, a_0 = 1$. Треба доказати да $a \in M \Rightarrow a + 1 \in M$. Будући да a има облик (1), то $a + 1$ има један од облика

$$(1 + a_0) + a_1 n + \cdots + a_p n^p \ (1 + a_0 < n);$$
$$(1 + a_1) n + \cdots + a_p n^p \ (1 + a_0 = n, 1 + a_1 < n);$$
$$(1 + a_2) n^2 + \cdots + a_p n^p \ (1 + a_0 = 1 + a_1 = n, 1 + a_2 < n);$$
$$\vdots$$
$$(1 + a_p) n^p (1 + a_0 = 1 + a_1 = \cdots = 1 + a_{p-1} = n, 1 + a_p < n);$$
$$n^{p+1} \ (1 + a_0 = 1 + a_1 = \cdots = 1 + a_p = n)$$

Будући да је и $a + 1$ облика (1), то и сваки природан број има бар један од облика (1).

Јединственост. Рецимо да је

$$(2) \begin{cases} a_0 + a_1 n + \cdots + a_p n^p = b_0 + b_1 n + \cdots + b_q n^q \\ (a_i, b_j \in \boldsymbol{Z}_n, a_p \neq 0, b_q \neq 0). \end{cases}$$

Из (2) следи да су бројеви $a_0 - b_0, b_0 - a_0$ дељиви са n. Бар један од тих бројева је позитиван и мањи од n; Дакле $a_0 = b_0$. Сада (2) прелази у

$$a_1 n + \cdots + a_p n^p = b_1 n + \cdots + b_q n^q$$

одакле се на исти начин добија $a_1 = b_1$. Тим поступком добија се $p = q$ и $a_i = b_i (i = 0,1, \dots p)$■[4]

Да би се истакла база система у којем је неки број записан, често се запис ставља у заграде, а у индекс се записује основа система. Индекс изостављамо када је база 10 или када је јасно о којој је основи реч.

Пример 1. Записом $(2305)_7$ представљен је број у систему са основом 7. Приказ овог броја у декадном систему (на који смо навикли) једноставно се одређује рачуном који произилази из значења самог записа:

$$(2305)_7 = 2 \cdot 7^3 + 3 \cdot 7^2 + 0 \cdot 7^1 + 5 = 838$$

Поступак превођења декадног записа неког природног броја у систем било које основе b илустрован је у примеру 2.

Уколико је $2 \le b \le 10$ за записивање бројева у систему са основом b користе се арапске цифре – наравно, само оне које означавају бројеве мање од b.

Пример 2. Прикажимо број 876 (записан у декадном систему) у системима за различит избор базе b.

База	**Поступак превођења**	**Превод**
$b = 7$	$876 = 125 \cdot 7 + 1$ $125 = 17 \cdot 7 + 6$ $17 = 2 \cdot 7 + 3$ $2 = 0 \cdot 7 + 2$	$876 = (2361)_7$ Провера: $876 = 2 \cdot 7^3 + 3 \cdot 7^2 + 6 \cdot 7 + 1$ $876 = (1554)_8$
$b = 8$	$876 = 109 \cdot 8 + 4$ $109 = 13 \cdot 8 + 5$ $13 = 1 \cdot 8 + 5$ $1 = 0 \cdot 8 + 1$	
$b = 2$	$876 = 438 \cdot 2 + 0$ $438 = 219 \cdot 2 + 0$ $219 = 109 \cdot 2 + 1$ $109 = 54 \cdot 2 + 1$ $54 = 27 \cdot 2 + 0$ $27 = 13 \cdot 2 + 1$ $13 = 6 \cdot 2 + 1$ $6 = 3 \cdot 2 + 0$ $3 = 1 \cdot 2 + 1$ $1 = 0 \cdot 2 + 1$	$876 = (1101101100)_2$

Вредност броја записаног у неком бројевном систему може се израчунати и коришћењем **Хорнерове шеме**

$$(a_n a_{n-1} \dots a_1 a_0)_{\boldsymbol{b}} = (((a_n \cdot b + a_{n-1}) \cdot b + a_{n-2}) + \cdots + a_1) \cdot b + a_0,$$

где су $a_n, \dots, a_0$ цифре датог броја, а b је основа.

Задатак 1. Коришћењем Хорнерове шеме одредити вредности бројева $(3127)_8$ и $(3129)_{16}$.

$$(3127)_8 = \Big(\big((3 \cdot 10) + 1\big) \cdot 10 + 2\Big) \cdot 10 + 7 = (1623)_{10}$$
$$(3129)_{16} = \Big(\big((3 \cdot 10) + 1\big) \cdot 10 + 2\Big) \cdot 10 + 9 = (12585)_{10}$$

Ако је $(a_n a_{n-1} \dots a_1 a_0)_b$ број чију вредност треба израчунати и x његова вредност, псеудокод за Хорнерову шему би изгледао:

begin
    $x \coloneqq a_n;$
    for $i = 1$ *to* n
        $x \coloneqq x \cdot b + a_{n-1};$
end

За разлику од људи, рачунари уместо прстију поседују прекидаче који им омогућавају рачунање и сваки од њих може бити укључен или искључен. Укључен прекидач је представљен јединицом, а искључен нулом, што за последицу има да рачунари користе бројевни систем са основом два (бинарни бројевни систем), чије су цифре 0 и 1. (Наравно, овакав приказ рачунарског система је упрошћен у односну на стварни). У рачунарству се такође користе систем са основом осам (октални бројевни систем) и систем са основом шеснаест (хексадекадни бројевни систем). Ова два система имају много компактнији запис од бинарног, а уједно се веома лако преводе у бинарни систем и обратно. Задатак 2. илуструје једноставност превођења међу овим системима.

Задатак 2. Број бинарни 11101, превести у октални, па у хексадецимални систем.

С обзиром да постоји веза међу основама ова три система, односно да се основа једног система може представити као степен друге, превођење између ових система је много лакшше него међу системима где не постоји оваква веза. Основа бинарног система је 2, основа окталног је $8 = 2^3$ и основа хексадецималног је $16 = 2^4$. Ово повлачи да ће три цифре бинарног представљати једну цифу окталног система, односно четири цифре бинарног једну цифру хексадециманог.

Да би се број 11101 превео у октални, потребно је да се од његових цифара формирају групе од по три, а након тога сваку групу бинарних цифара превести у једну окталну цифру. Односно, важи:

$$11101$$
$$101 \rightarrow 1 \cdot 2^0 + 0 \cdot 2^1 + 1 \cdot 2^2 = (5)_8$$

11 је када се дода нула на почетак $011 \rightarrow 1 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 = (3)_8$
што значи да је вредност броја $(11101)_2$ преведеног у октални 35.

Превођење у хексадецимални је слично, само што се бинарне цифре групишу по четири, односно:

$$11101$$
$$1101 \rightarrow 1 \cdot 2^0 + 0 \cdot 2^1 + 1 \cdot 2^2 + 1 \cdot 2^3 = (13)_{16} = D_{16}$$
$$(10 = A, 11 = B, 12 = C, 13 = D, 14 = E \text{ и } 15 = F,$$
$$\text{цифре хексадецималног система које су веће од 9})$$
$$0001 \rightarrow 1 \cdot 2^0 + 0 \cdot 2^1 + 0 \cdot 2^2 + 0 \cdot 2^3 = (1)_{16}$$

што на крају износи: $(11101)_2 = (1D)_{16}$

Ова превођења могу бити моного бржа уклико се напамет занају вреднсоти цифара у различитим системима. Веза бинарних и хексадецималних цифара

$$0000 \rightarrow 0 \;\; 1000 \rightarrow 8$$

$$0001 \rightarrow 1 \;\; 1001 \rightarrow 9$$

$$0010 \rightarrow 2 \;\; 1010 \rightarrow A$$

$$0011 \rightarrow 3 \; 1011 \rightarrow B$$

$$0100 \rightarrow 4 \; 1100 \rightarrow C$$

$$0101 \rightarrow 5 \;\; 1101 \rightarrow D$$

$$0110 \rightarrow 6 \;\; 1110 \rightarrow E$$

$$0111 \rightarrow 7 \;\; 1111 \rightarrow F$$

Пример 3. Превести број $(1A)_{16}$ у октални и бинарни систем.

Најлакше је прво превести у бинарни, па потом из бинарног у октални систем.
Цифра 1 је у хексадецималном систему 0001, док је цифра $\mathrm{A} = 1010$, тј. $(1A)_{16}$ је у бинарном систему 00011010. Превођење из бинарног у октални би изгледало

$$010 \rightarrow 2$$
$$011 \rightarrow 3$$
$$000 \rightarrow 0$$

Односно $(00011010)_2 = (32)_8$.

2. Бројеви и технологија

Свет технологије почива на систему са основом два. Рачунари читају податке као електричне импулсе који имају високу или ниску волтажу. Компакт дискови чувају податке користећи минијатурна удубљења на површини диска, па се и овде разликују два стања – постоји удубљење и не постоји удубљење. Та два стања могу бити представљена цифрама бинарног система, цифром нула се представља једно стање, а цифром један друго стање. Бинарна цифра, било да је она 0 или 1, назива се **бит** (енглески **bit*).*** Реч бит потиче од енглеских речи ***bi****nary* (бинарни) и *digi****t*** (цифра).

2.1.Представљање података у рачунару

С обзиром да су за представљање података у рачунару на располагању само два стања, односно две цифре, 0 и 1, сваки податак и инструкција који се уносе у рачунар морају бити представљени као запис састављен од нула и јединица. Такав запис се назива **бинарни запис.** За податке и инструкције представљене помоћу цифара 0 и 1 каже се да су бинарно представљени, а свака цифра тог записа (нула или јединица) се назива **бит,** док се група од осам битова назива **бајт.**

Бит је веома мала количина информација, јер се једим битом могу представити само цифра 0 и цифра 1. Да би се представила цифра 2 потребна су два бита (јер је $2 = 10_2$). Да би се кодирала било која смисленија информација (једно слово неког текста), потребно је употребити неколико битова. Стога су битови груписани у бајтове и сваки бајт има свој редни број, тзв. адресу у меморији. Више о овој теми може се наћи у [12].

2.1.1. Представљање текста у рачунару

Рачунари и њихови периферни уређаји користе бинарне цифре за читање, чување и уписивање информација.

Текст је у рачунару представљен као **низ карактера. Карактери** су сви елементи из којих се састоји текст (свако слово алфабета, специјални карактер, интерпукцијски знаци). Да би се текст представио бројевима врши се **кодирање карактера**. Кодирање сваком карактеру придружује јединствени број (његов **код**). Како би текст унет у једном рачунару на једном програму био исти на другом рачунару, мора да постоји стандардизовано кодирање и да сви произвођачи рачунарских система користе стандардна кодирања. Из тог разлога састављене су стандардне таблице у којима су пописани карактери и њихови кодови. Овакве таблице се називају **кодне стране.**[12]

Када рачунар чита или шаље информације, оне су обрађене у некој кодној страни. Када су информације послате до уређаја, као на пример штампача, штампач дешифрује (декодира) бинарну информацију и преводи у одговарајући карактер.

Пре одређивања бројевних кодова за сваки појединачни карактер, потребно је одредити који се скуп карактера кодира (различита писма, спрецијални карактери итд.). С обзиром на то да су први електронски рачунари настали у Сједињеним Америчким Државама, рани рачунари допуштали сус само кодирање карактера за енглеско говорно подручје. Почетком 1960-тих, стандардизована је ASCII (American Standard Code for Information Interchange) таблица, која кодира 128 карактера који се користе на енглеском говорном подручју.[12]

Примери ASCII кода за велика слова енглеске абецеде дата су у *Табели 3.* [5]

Табела 3. ASCII код за слова енгеске абецеде

СЛОВО	**ASCII код**	**СЛОВО**	**ASCII код**	**СЛОВО**	**ASCII код**
A	0100 0001	J	0100 1010	S	0101 0011
B	0100 0010	K	0100 1011	T	0101 0100
C	0100 0011	L	0100 1100	U	0101 0101
D	0100 0100	M	0100 1101	V	0101 0110
E	0100 0101	N	0100 1110	W	0101 0111
F	0100 0110	O	0100 1111	X	0101 1000
G	0100 0111	P	0101 0000	Y	0101 1001
H	0100 1000	Q	0101 0001	Z	0101 1010
I	0100 1001	R	0101 0010	размак	0010 0000

Како су рачунари почели да се користе у различитим говорним подручјима, развила се потреба за кодирањем другачијих карактера, односно потреба за дугим кодним странама. Тако је креиран јединствени стандард који је обухватао све карактере свих језика. Данас се у ове сврхе користи **Unicode**, који је настао 1990-тих. Unicode је направљен са намером да буде универзалан – покрива све савремене језике са писмом, јединствен – нема дуплирања карактера јер се кодирају писма, а не језици, и униформан – сваки карактер се кодира истим бројем битова. [12]

Unicode у својој основној вишејезичкој равни садржи 65536 карактера (чији се кодови записују помоћу два бајта) и који су довољни за запис већине живих језика. Првих 128 карактера у таблици поклапа се са таблицом ASCII, за којима следе карактери западноевропских латиница, након тога остали латинични карактери, па ћирилица, грчки алфабет итд. Једна од модификација Unicode кодирања која се назива **UTF-8** (енгл. *Unicode Transformation Format*). Идеја ове врсте кодирања јесте да се карактери кодирају раличитим бројем бајтова: ASCII карактери се кодирају једним бајтом, остали латинични карактери, грчка слова и ћирилични карактери кодирају се помоћу два бајта, док се кинески карактери кодирају помоћу три бајта. [12]

Све ово се обрађује у оквиру предмета Рачунарство и информатика у првом разреду гимназије. Циљ овог поглавља јесте да се успостави корелација између математике и рачунарства и информатике, као и да се решавањем примера нове технологије сагледају из математичког угла, односно из правца одакле су настале. Једноставности ради примери показани у овом раду тичу се ASCII кодирања.

Задатак 3. Записати ASCII код речи **BROJ**.
Решење:

B=01000010
R=01010010

O=01011111

J=01001010

Односно, ASCII код речи BROJ је 01000010010100100101111101001010.

Задатак 4. Шта се добија када се дешифрује следећи код:

01000001010011100100000 1?

Решење:

Пошто се један карактер записује помоћу осам битова, може се закључити да првих осам битова представљају слово A, наредних осам слово N и последњих осам поново слово A.

Низ бинарних цифара је тежак за читање, писање и памћење, па су кодови обично записани у окталном систему (систем са основом 8) или у хексадецималном систему (систему са основом 16).

2.1.2. Представљање слике у рачунару

За разлику од аналогне фотографије која успоставља неку врсту аналогије између онога што се фотографисало, слике на негативу и коначне слике, дигитално записана фотографија је заснована на другачијим принципима. Простор фотографије је дискретизован и издељен на квадратиће, који се називају **пиксели** (енгл. *pixel – picture element*). Што је пиксела више, слика је квалитетнија. С обзиром на то да се светлост било које боје може разложити на светлост црвене, зелене и плаве боје. На пример, љубичаста је комбинација црбене и зелене. Интензитет сваке светлосне компоненете може се записати бројем па се боја сваког пиксела може представити помоћу три броја. То би значило да се слика посматра као низ пиксела, а сваки пиксел се представља бројевима, онда се и цела слика посматра као низ бројева. Фото-ћелија дигиталног апарата дели простор на пикселе, мери количину сваке светлосне компоненте и добијене бројеве записује на меморијску картицу фото-апарата. [12]

Најједноставнија од свих **јесте једнобитна слика** (енгл. *one-bit image*), где је сваки пиксел представљен помоћу једног бита. Ако је бит пиксела 0, онда је тај пиксел обојен у бело, а ако је 1 тада је обојен у црно. Једнобитна слика је сачињена само од црне и беле боје, без сиве.

Код **слике сачињене од сивих тонова** (енгл. *grayscale*), сваки пиксел је обично представљен са осам бита (осам бита представљају један бајт), односно бројем који има осам цифара у бинарном запису. Најмањи осмобитни број је $(00000000)_2 = 0$ и пиксел коме је придружена нула обојен је белом бојом. Највећи осмобитни број је $11111111_2 = 1 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 255$ и пиксел коме је придружен декадни број 255 обојен је у црно. Пиксели којима су придружени бројеви између 0 и 255 су представљени неком нијансом сиве.

Код **RGB** слике сваком пикселу су придружена три различита броја величине једног бајта. Један од та три броја одређује количину црвене (R-red), други зелене (G-green) и трећи плаве (B-blue) које ће заједно чинити једну боју. Сваки пиксел RGB слике садржи 256 различитих нијанси црвене, 256 различитих нијанси плаве и 256 различитих нијанси зелене.

Ако је слика у **CMYK** формату, најчешће су сваком пикселу су придружена четири броја величине једног бајта. Четири броја одређују количину цијан (C-cyan), магента (M-magenta), жуте (Y-yellow) и црне (K-black). Сваком од та четири броја додељен је један бајт, тако да сваки пиксел CMYK слике може да прикаже 256 нијанси сваке од ове четири боје.

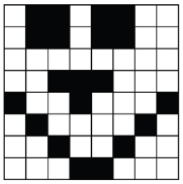
Слика 1. Боје слика у CMYK формату [2]

Различити формати слика захтевају различит број битова потребних да би се ускладиштиле информације о боји сваког пиксела. Дубина бита је број бита који се користи да би сачувао информацију о боји сваког пиксела, па тако дубина боје за **one-bit** слику је један, за **grayscale** је осам, за **RGB** је $3 \cdot 8 = 24$, а за **CMYK** је $4 \cdot 8 = 32$.

(а) (б) (в)

Слика 2. Слике различитих формата: (а) ***CMYK*** (б) ***Grayscale*** (в) ***one-bit*** *[2]*

Слика као битови



Слика 3. Слика представљена помоћу бита

Мрежа на којој су неки квадратићи обојени у црно представља насмејано лице. Једине боје које су коришћене су црна и бела, што се врло лако преводи у језик читљив свим рачунарима. Ако беле квадратиће представимо цифром 1, а црне цифром 0, тада ће цела слика бити представљена бројем

1001001110010011111111111100011101101110101111011101101111100111

Зна се да је лице нацртано на мрежи од осам вертикалних и осам хоризонталних квадратића. Сваки квадратић може бити црн или бео, па због тога низ од осам квадратића може на 256 начина да буде обојен. Сви црни квадратићи ће имати вредност 0, а сви бели квадратићи ће имати вредност 255.

Може се уочити да уколико се уместо јединице запишу нуле и обратно добиће се слика инверзна претходној, односно

01101100011011000000000000111000100100010100001000100100000011000

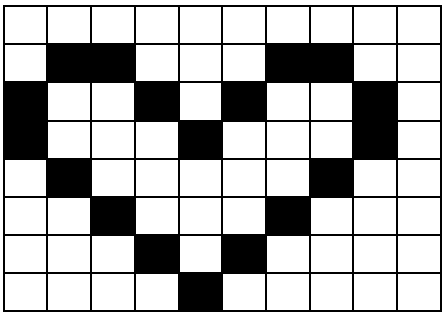
даје следећу слику.

Слика 4. Заменом битова добија се инверзна слика

Овим је показано како се једноставне слике представљају низом бинарних цифара.

Задатак 5. Нацртати нешто на папиру са квадратном мрежом. Затим користећи бинарну аритметику превести слику у бинарни код, а након тога у хексадецимални запис. На крају хексадецимални број проследити неком другом ко би покушао да реконструише послату слику.

Следи реконструкција како би могао да изгледа ток часа приликом израде овог задатка. Нека је прва група нацртала следећу слику



Следеће што је потребно јесте да кодира (преведе) ову слику у бинарни запис – бели квадратићи биће представљени нулом, а црни јединицом. Записивање ће кренути слева надесно, од првог реда. Дакле, слика у бинарном облику изгледа овако:

0000000000 0110001100 1001010010 1000100010 0100000100 0010001000 0001010000 0000100000

Сада се цртеж из бинарног облика преводи у хексадецимални.

Са превођењем се креће слева надесно узимајући групе од по четири цифре

$$0000 = 0 \;\; 0000 = 0 \;\; 0001 = 1 \;\; 1000 = 8 \;\; 1100 = C$$

$$1001 = 9 \;\; 0100 = 4 \;\; 1010 = A \;\; 0010 = 2 \;\; 0010 = 2$$

$$0100 = 4 \;\; 0001 = 1 \;\; 0000 = 0 \;\; 1000 = 8 \;\; 1000 = 8$$

$$0001 = 1 \;\; 0100 = 4 \;\; 0000 = 0 \;\; 0010 = 2 \;\; 0000 = 0$$

Коначно, цртеж преведен у хексадецимални запис изгледа: $0018C94A224108814020$. Након успешног превођења прва група би послала другој групи хексадецималну вредност цртежа, док би друга група добивши број покушала да на основу њега нацрта првобитну слику. Поступак декодирања поруке би био следећи: прво се добијени број преводи у бинарни тако што се свака цифра хексадецималног система представи са четири цифре бинарног система (на пример $0_{16} = 0000_2, C_{16} = 1100_2$ итд). Када се број из хексадецималног система преведе у бинарни систем, онда се приступа цртању по квадратној мррежи и на крају би друга група требало да добије исти цртеж који је био послат.

III КОНГРУЕНЦИЈЕ И МОДУЛАРНА АРИТМЕТИКА

У овом поглављe биће обрађени математички појмови који ће омогућити увођење нових, сложенијих метода шифровања, али и детаљније образлагање већ виђеног у првом поглављу. Поред тога као још једна примена модуларне аритметике биће објашњена структура и функционисање бар-кода.

1. Дељивост, прости бројеви

Дефиниција 1. Нека $\boldsymbol{Z}$ означава скуп целих бројева, $\boldsymbol{Z} = \{0, \pm 1, \pm 2, \dots\}$. За целе бројеве $a, b \in \boldsymbol{Z}$ кажемо да a дели b (ознака $a|b$) ако је $b = na$ за неко $n \in \boldsymbol{Z}$**.** Број a дели b ако и само ако је b умножак a.

Особине оператора |:

1. Ако $a, b, c \in \boldsymbol{Z}$ и $a|b$, онда $a|bc$. Тако, из $3|12$ следи $3|60$
2. Ако $a|b$ и $a|c$, онда $a|(b \pm c)$
3. Ако $a|b$ и не важи да $a|c$, онда не важи ни да $a|b \pm c$

Дефиниција 2. Позитиван број је прост број ако су његови једини делиоци број 1 и сам тај број. Број један није ни прост ни сложен број.

Теорема 1. (Основна теорема аритметике) Сваки број $n \in \boldsymbol{Z}$, $n > 1$, може се на јединствен начин представити у облику производа простих бројева $n = {p_1}^{\alpha_1} {p_2}^{\alpha_2} \dots {p_r}^{\alpha_r}$, где су α_i позитивни цели бројеви. Сви позитивни делиоци броја n су облика ${p_1}^{\beta_1} {p_2}^{\beta_2} \dots {p_r}^{\beta_r}$, где је $0 \le \beta_i \le \alpha_i$ за свако $i = 1, 2, \dots, r$, па број n има $(\alpha_1 + 1)(\alpha_2 + 1) \dots (\alpha_r + 1)$ различитих делилаца.

Доказ

Егзистенција разлагања. Тврђење да постоји разлагање броја $a > 1$ на просте факторе доказује се индукцијом. Ово је очигледно за $a = 2$, јер је 2 прост број. Претпоставимо да да сви бројеви из $\{2, \dots, a - 1\}$ имају бар по једно разлагање у производ простих бројева.

Ако је сам број a прост, доказ је завршен, у супротном нека је $p > 1$ најмањи нетривијални делилац броја a. Очигледно, p мора да буде прост број јер би у супротном постојао број q који је делилац броја a и мањи је од p што је у супротности са избором броја p. Према томе, важи да је $a = pa'$, где је $1 < a' < a$; даље је индуктивна претпоставка применљива на a', односно a' је производ простих бројева: $a' = p_1 p_2 \dots p_m$. Тада је $a = p p_1 p_2 \dots p_m$

што је и требало доказати.

Јединственост разлагања. Претпоставимо да је $a = p_1 p_2 \dots p_r = q_1 q_2 \dots q_s$; без умањења општости, нека је $r \leq s$. Пошто $p_1 | q_1 q_2 \dots q_s$ и p_1 је прост број, може се закључити да је $p_1 | q_{j_1}$ за неко j_1; међутим, и q_{j_1} је као и p_1 прост број, па је $p_1 = q_{j_1}$. Исти закључак се може поновити и за $p_2, \dots, p_r$ за свако $1 \leq i \leq r$ постоји индекс j_i тако да је $p_i = q_{j_i}$. При томе су сви индекси $j_1, \dots, j_r$ међусобно различити. Уколико би било $r < s$, тада би се по означавању $\{k_1, \dots, k_{s-r}\} = \{1, \dots, s\} \backslash \{j_1, \dots, j_r\}$, добило да је

$$1 = q_{k_1 \dots} q_{k_{s-r}}$$

што је очито немогуће. Дакле, мора бити да је $r = s$. ■

Пример 1. $90 = 2^1 \cdot 3^2 \cdot 5^1$. Да бисмо пронашли све позитивне делиоце броја 90, посматрајмо следећу шему.

1 1 1

2 3 5

9

Први ред представља нулте степене бројева $2, 3$ и 5, наредни ред представља прве степене истих бројева, а последњи ред је други степен броја 3 (други степени бројева 2 и 5 нису приказани у шеми јер се они не појављују у факторизацији броја 90). У општем случају прво се дати број растави на просте чиниоце. Потом се шема формира тако што се прости чиниоци подижу редом на степене од нултог до степена са којим учествују у факторизацији датог броја. Исти степени простих бројева су поређани у исти ред. На крају треба проћи све могуће путеве тако што ће се бројеви који чине шему међусобно монжити. Добијени бројеви представљају све могуће делиоце датог броја.

Даље, треба проћи све могуће путеве слева удесно: $1 \cdot 1 \cdot 1 = 1$, $1 \cdot 1 \cdot 5 = 5$, $1 \cdot 3 \cdot 1 = 3$, $1 \cdot 3 \cdot 5 = 15$, $1 \cdot 9 \cdot 1 = 9$, $1 \cdot 9 \cdot 5 = 45$, $2 \cdot 1 \cdot 1 = 2$, $2 \cdot 1 \cdot 5 = 10$, $2 \cdot 3 \cdot 1 = 6$, $2 \cdot 3 \cdot 5 = 30$, $2 \cdot 9 \cdot 1 = 18$, $2 \cdot 9 \cdot 5 = 90$.

Последица 1. Ако је p прост број и $p | ab$, онда $p | a$ или $p | b$ (Није увек тачно за сложене бројеве, на пример $4 | 6 \cdot 2$, а не важи ни једно од тврђења да $4 | 6$ или $4 | 2$).

Пример 2. Пошто $3 | 180 = 4 \cdot 45$, онда $3 | 4$ или $3 | 45$. У овом случају је тачно друго тврђење.

Дефиниција 3. Нека су a, b из $\boldsymbol{Z}^+$ позитивни цели бројеви, од којих један може бити 0. **Највећи заједнички делилац** (NZD) a и b (ознака nzd(a, b)) је највећи цели број d који дели и a и b. Приметимо да ако d дели a и d дели b, онда d дели nzd(a, b). За бројеве којима је NZD једнак 1, кажемо да су **узајамно прости**.

Ако знамо факторизације бројева $a = {p_1}^{\alpha_1}{p_2}^{\alpha_2} \dots {p_r}^{\alpha_r}$ и $b = {p_1}^{\beta_1}{p_2}^{\beta_2} \dots {p_r}^{\beta_r}$ (неки од експонената могу бити 0), онда је $\text{nzd}(a,b) = {p_1}^{\gamma_1}{p_2}^{\gamma_2} \dots {p_r}^{\gamma_r}$ где је $\gamma_i = \min\{\alpha_i, \beta_i\}$.

Пример 3. $2520 = 2^3 \cdot 3^2 \cdot 5^1 \cdot 7^1$ и $2700 = 2^2 \cdot 3^3 \cdot 5^2 \cdot 7^0$, па је $\text{nzd}(2520,2700) = 2^2 \cdot 3^2 \cdot 5^1 \cdot 7^0 = 180$

Еуклидов алгоритам Нека $a \ mod \ b$ означава остатак при дељењу a са b; ако је $\boldsymbol{a = qb + r, 0 \leq r \leq b}$, онда је $\boldsymbol{a \ mod \ b = r}$.

На пример $12 \ mod \ 5 = 2$, $7 \ mod \ 5 = 2$. Растављање великих бројева на просте чиниоце је тежак проблем. Уместо помоћу растављања на просте чиниоце, једноставан и ефикасан начин за израчунавање *NZD* је примена Еуклидовог алгоритма. Користи се чињеница да је $nzd(a,b) = nzd(a \ mod \ b, b)$.

Доказ Еуклидовог алгоритма

Нека је d произвољан заједнички делилац бројева a и b. Тада из релације $a = bq + r$ следи да је он и делилац броја r, односно заједнички делилац бројева b и r. Слично, ако је d произвољан заједнички делилац бројева b и r, из исте релације следи да је он и заједнички делилац бројева a и b. Дакле, скупови заједничких делилаца бројева a и b, односно b и r се поклапају. Зато су и међусобно једнаки и њихови највећи елементи, дакле бројеви $nzd(a,b) = nzd(b,r)$. ■

Пример 4. Одредимо $\text{nzd}(329,119)$.

Најпре делимо 329 са 119; добијамо количник 2 и остатак 91. У сваком наредном кораку делилац и остатак постају наредни дељеник и делилац:

$$329 = 2 \cdot 119 + 91$$

$$119 = 1 \cdot 91 + 28$$

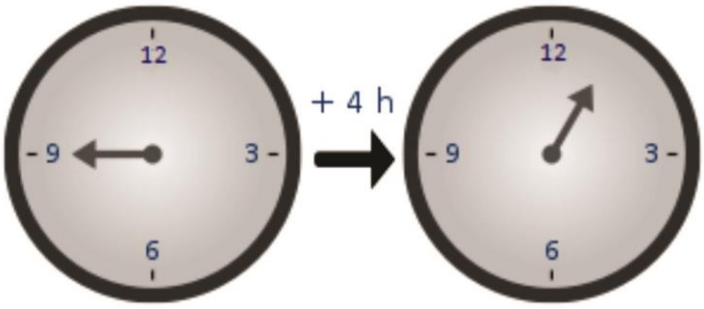
$$91 = 3 \cdot 28 + 7$$

$$28 = 4 \cdot 7 + 0$$

Последњи остатак различит од 0 је тражени *NZD*. Овде је $nzd(329,119) = 7$.

2.Модуларна аритметика

Најједноставнију илустрацију модуларне аритметике даје модел часовника. Постоје два начина за мерење времена – систем од дванаест часова у току дана и систем од двадесет четири часа у току дана, и оба су циклична. Сате у току дана не бројимо до бесконачности, већ у круг, сваки пут када се дође до одређене вредности (дванаест или двадесет четири) бројање креће испочетка. Овакво мерење времена представља примену модуларне аритметике. На пример, четири сата после 21: 00 часа (девет сати увече), не износи 25: 00 часова (као што би било код уобичајеног сабирања: 21 + 4 = 25), већ је тада 01:00 сат наредног дана. Како часови поново почињу од 00 након што прође 12 сати, односно 24 сата, овде се ради о аритметици по модулу 12, односно о аритметици по модулу 24.



Слика 3. Рачунање времена у дванесточасовном систему

Дванаесточасовно бројање времена има основни циклус од дванаест - односно броји се испочетка када се достигне дванаест сати.

Табела 3. Сабирање времена у дванаесточасовном систему

+	1	2	3	4	5	6	7	8	9	10	11	12
1	2	3	4	5	6	7	8	9	10	11	12	1
2	3	4	5	6	7	8	9	10	11	12	1	2
3	4	5	6	7	8	9	10	11	12	1	2	3
4	5	6	7	8	9	10	11	12	1	2	3	4
5	6	7	8	9	10	11	12	1	2	3	4	5
6	7	8	9	10	11	12	1	2	3	4	5	6
7	8	9	10	11	12	1	2	3	4	5	6	7
8	9	10	11	12	1	2	3	4	5	6	7	8
9	10	11	12	1	2	3	4	5	6	7	8	9
10	11	12	1	2	3	4	5	6	7	8	9	10
11	12	1	2	3	4	5	6	7	8	9	10	11
12	1	2	3	4	5	6	7	8	9	10	11	12

Треба приметити да број 12 има улогу коју иначе има нула. Који год број да додамо на број 12, резултат је тај број који смо додали на 12.

$$12 +_{12} 2 = 2$$

$$12 +_{12} 7 = 7$$

$$12 +_{12} 12 = 12$$

Ово значи, редом, да је два сата након 12 ч, два сата после подне, седам сати након 12 ч седам сати после подне и дванаест сати након 12 ч поноћ ($+_{12}$ је означено као сабирање сати у дванаесточасовном систему). Да би се израчунао збир у дванаесточасовном систему потребно је да се бројеви саберу као у декадном и уколико је збир већи од дванаест, од добијеног збира се одузима дванаест (напомена: за веће бројеве потребно је више пута одузети број дванаест). Други начин јесте да се израчуна збир и потом се подели бројем који представља модуо. Добијени остатак је управо резултат сабирања два броја по датом модулу.

Први начин:

$7 + 2 = 9$

$7 + 6 = 13$ и $13 - 12 = 1$

$5 + 11 = 16$ и $16 - 12 = 4$

Други начин:

$7 + 2 = 9, 9 : 12 = 0$ и остатак $= 9$

$7 + 6 = 13, 13 : 12 = 1$ и остатак $= 1$

$5 + 11 = 16, 16 : 12 = 1$ и остатак $= 4$

У првом примеру решење је 9, другом 1 и трећем 4.

На оба начина су добијени исти резултати. Добијени су исти резултати јер се операција дељења може представити узастопном применом операције одузимања. Да је било потребно израчунати колико је 50 по модулу 12, на први начин би се израчунало узастопнинм умањивањем за 12 док се не добије број који је мањи од 12, односно $50 - 12 = 38, 38 - 12 = 26, 26 - 12 = 14$ и на крају $14 - 12 = 2$, тј. 50 по модулу 12 је 2. На други начин рачун би био следећи: $50 : 12 = 4$ и остатак$= 2$, дакле резултат је исти, број 2. Остало је да се прокометарише шта значи остатак при дељењу уколико је дељеник негативан број. Колико је -3 по модулу 12? Пошто се број 12 не садржи у броју -3, резултат је 0 и остатак -3. Како се остатак, односно резултат, не може представити као негатив број, потребно позајмити 12 (када се рачуна по модулу 10 позајмљује се по 10) и тада је $-3 + 12 = 9$. Сада је -3 по модулу 12 једнако броју 9.

Пошто број дванаест, по модулу дванаест, има вредност нула, користиће се нула кад год се појави дванаест (табела 2). Овим је приказано сабирање коришћењем само цифара од 0 до 11, што представља сабирање у аритметици по модулу 12.

Табела 4. Сабирање у аритметици по модулу 12

+	1	2	3	4	5	6	7	8	9	10	11	0
1	2	3	4	5	6	7	8	9	10	11	0	1
2	3	4	5	6	7	8	9	10	11	0	1	2
3	4	5	6	7	8	9	10	11	0	1	2	3
4	5	6	7	8	9	10	11	0	1	2	3	4
5	6	7	8	9	10	11	0	1	2	3	4	5
6	7	8	9	10	11	0	1	2	3	4	5	6
7	8	9	10	11	0	1	2	3	4	5	6	7
8	9	10	11	0	1	2	3	4	5	6	7	8
9	10	11	0	1	2	3	4	5	6	7	8	9
10	11	0	1	2	3	4	5	6	7	8	9	10
11	0	1	2	3	4	5	6	7	8	9	10	11
12	1	2	3	4	5	6	7	8	9	10	11	0

За решавање наредних примера биће коришћена модуларна аритметика. У оваквим системима најбитнији је остатак. Ова аритметика опонаша модел сата. Ако је тренутно 2ч, које ће бити време за 50 сати? Једноставно се уочава да је на сваких дванаест сати казаљка на истом месту (без обзира да ли означава сате пре или после поноћи). Дељењем броја 50 са 12 добија се 4 и остатак 2, што значи да ће казаљка обићи четири пуна круга и померити се за два сата у односу на време од кога смо почели да меримо сате. Одговор је да ће за 50 сати од сада бити 4ч.

Модуларна аритметика може бити примењена на решавање проблема из свакодневног живота, али се уједно користи у сигурносним системима (приликом куповине кредитном картицом, куповином путем интернета), о чему ће касније бити речи.

Пример 5. Одредити колону у којој ће се налазити број 283.

А	Б	В	Г	Д	Ђ	Е	Ж
1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
…							

Решење:

У сваком реду се налази осам узастопних бројева. Да би се пронашло у којој се колони налази 283, потребно је поделити 283 са 8 и на основу остатка ће бити могуће одредити у којој колони је тражени број. Уколико је остатак 1, колона је А, ако је остатак 2, колона је Б итд. Дељењем 283 са 8 добија се остатак 3, што значи да је 283 у колони В.

Претходни пример би било немогуће (уколико би био већи број у питању), или изузетно напорно решити без модуларне аритметике.

Пример 6. Немања је 4. септембра купио полису осигурања. Тада је био понедељак. Полиса ће бити активирана тек када прође 45 дана. Ког дана и датума ће полиса бити активирана?

Решење:

Потребно је једино схватити да је сваки седми дан понедељак. Ако се 45 подели са 7, добија се остатак 3. Када се изброји од понедељка три дана, то ће бити четвртак 19. октобар.

3.Бар-код

Гледајући у производе који се могу наћи у продавницама, скоро да не постоји неки који на себи нема образац сачињен од линија и празнина различите дебљине. Захваљујући тим обрасцима сваки производ има јединствен код за који је у српском језику прихваћен енглески термин бар-код. Бар-кодови убрзавају процес провере артикала на каси и омогућавају постојање јединственог списка производа који се налазе у радњи.

Означавање артикала бар-кодом поникло је у Америци средином 70-тих година 20. века, а у Европи неколико година касније. GS1 систем је јединствени међународни систем шифрирања, симболизације и идентификације. Настао је као европски систем (*European Article Numbering* - Европско нумерисање артикала), а прихватањем од стране земаља широм света постао је глобални систем. GS1 систем обезбеђује неколико врста кодова за употребу, у зависности од примене. Бар-кодови које употребљава GS1 укључују EAN/UPC, RSS, GS1-128, ITF-14, Data Matrix и Композитну компоненту. На слици 4. су приказане две врсте бар-кода. [7] У раду ће бити разматран EAN-13 бар-код.

Слика 4. Примери бар-кода UPC-A (најзаступљенија врста бар-кода у Америци) и EAN-13 (најзаступљенија врста бар-кода у Европи)

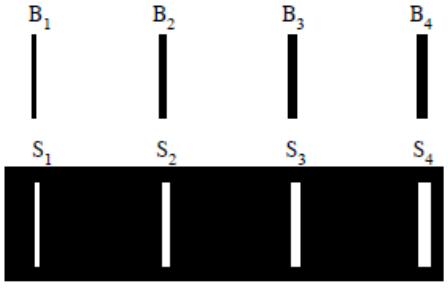
Слика 5. Примери бар-кода Data Matrix и ITF-14

Захваљујући црним линијама и размацима између њих бар-кодови су читљиви за рачунаре, а захваљујући цифрама испод читљиви за људе. Бар-код не укључује цену неког артикла, већ се помоћу читача бар-кода слика претвара у бинарни запис који се шаље до базе у којој су сачувани подаци о ценама појединачних производа. Рачунар тражи цену и шаље је назад. Читав овај процес траје око 0,3 секунде. [5]

Како линије и размаци чине број бар кода?

Дебљина линија и размака је одређена бројевима $1, 2, 3$ или 4. Линије означавају са B (енглески *bar*), а размаци са S (енглески *space*). Сваки бар-код (EAN-13) почиње и завршава се са $B_1 S_1 B_1$, док се у средини налази $S_1 B_1 S_1 B_1 S_1$. У EAN-13 систему прве две цифре (некада и три) говоре у којој земљи је израђен баркод (за Србију је 860), али то не мора да значи да је и сам производ израђен у тој земљи. Следећих пет (или четири ако је за земљу узето три) припадају компанији, а наредних пет цифара одређују конкретан производ. На крају је остала једна цифра – контролна цифра која служи за проверу исправности очитаног кода.

Слика 6. Значење цифара



Слика 7. Дебљина линија и размака [5]

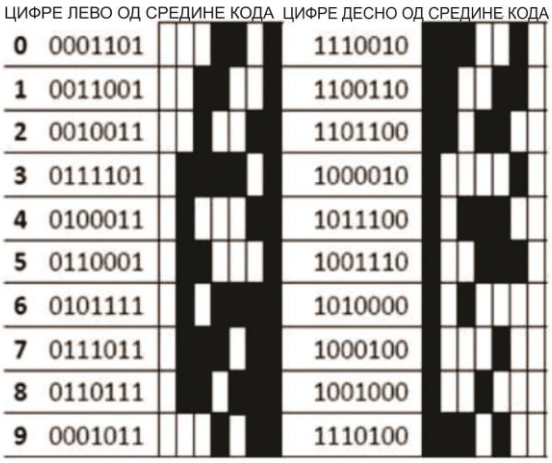
Цифре од 0 до 9 су шифроване комбинацијом линија и празнина. Цифре се представљају одговарајућом комбинацијом у зависности од тога да ли се налазе лево или десно од средине кода. У табели 5. цифре су представљене одговарајућим шифрама помоћу ознака за линије и размаке одређених дебљина.

Табела 5. *Шифровање цифара у бар-коду [5]*

Шифра	**Дебљина линија**	**Шифре десно од средине кода**	**Шифре лево од средине кода**
0	3-2-1-1	B_3 S_2 B_1 S_1	S_3 B_2 S_1 B_1
1	2-2-2-1	B_2 S_2 B_2 S_1	S_2 B_2 S_2 B_1
2	2-1-2-2	B_2 S_1 B_2 S_2	S_2 B_1 S_2 B_2
3	1-4-1-1	B_1 S_4 B_1 S_1	S_1 B_4 S_1 B_1
4	1-1-3-2	B_1 S_1 B_3 S_2	S_1 B_1 S_3 B_2
5	1-2-3-1	B_1 S_2 B_3 S_1	S_1 B_2 S_3 B_1
6	1-1-1-4	B_1 S_1 B_1 S_4	S_1 B_1 S_1 B_4
7	1-3-1-2	B_1 S_3 B_1 S_2	S_1 B_3 S_1 B_2
8	1-2-1-3	B_1 S_2 B_1 S_3	S_1 B_2 S_1 B_3
9	3-1-1-2	B_3 S_1 B_1 S_2	S_3 B_1 S_1 B_2

Табела 6. даје приказ сваке цифре. Уместо ознака за размак или линију налази се низ нула и јединица, и то за размак ширине три биће три нуле, за линију ширине два биће две једнинице. Графички приказ се прави тако што свака нула или јединица заузима правоугаоник исте ширине, с тим што ако је за нулу правоугаоник необојен, док је за јединицу правоугаоник обојен у црно.

Табела 6. *Шифровање цифара у бар-коду [5]*



Може се закључити да једној цифри одговара више линија/празнина. За сваку цифру је одвојено седам правоугаоника исте ширине који су различито обојени на основу табеле за шифровање. На следећој слици може се уочити које то линије/празнине одговарају цифри 4 на EAH-13 бар-коду. Ако се тај део кода увелича видеће се да одговара кодној шеми за цифру 4 - линија дебљине један, за којом следи празнина дебљине један, линија дебљине три и на крају размак дебљине два.

Слика 8. Бар-код са увеличаним делом који одговара једној цифри

Задатак 1. Дешифовати последње три цифре бар-кода.

Решење:

Прво је потребно да се погледа у последња три означена дела бар кода и упоредити их са вредностима које су дате у табели 5 или 6. У првој групи постоји једна линија дебљине 1 (B1), иза које следи размак дебљине 4 (S4), па линија дебљине 1 (B1) и на крају размак дебљине 1 (S1). Ако се погледа у табелу, низу B1S4B1S1, односно низу цифара 1-4-1-1 одговара цифра 3. На исти начин се долази до закључка које су преостале две цифре.

Задатак 2. Ако су прве две цифре које се налазе испод бар кода 5 и 7, како би изгледао одговарајући бар код?

Решење:
Из табеле се може пронаћи да је цифра 5 шифрована као $B1S2B3S1$, а цифра 7 као $B1S3B1S2$. Дакле, бар код који одговара датим цифрама је:

Ово важи за случај када је цифра 5 десно од средине кода. Слично за случај ако би се цифра 5 налазила лево од средине кода.

Чему служи контролна цифра?

Када се бар-код скенира могуће су грешке које могу бити узроковане на разне начине. На пример, дешава се да скенер не учита лепо бар-код, па уместо кода кутијице шибица прочита код који одговара скупом вину... Како би овакве ствари могле да се открију, сваки бар-код поседује своју контролну цифру, за чије се израчунавање користи модуларна артиметика, на основу које се препознаје да ли је дошло до грешке или не. У случају да је дошло до грешке приликом читања бар-кода, оно се може поновити или се могу унети цифре које се налазе испод бар-кода. Такође, приликом ручног уноса кода може доћи до грешке и најчешће грешке су пермутовање редоследа цифара, на пример 75 уместо 57 или унос погрешне цифре. Систем ће пронаћи грешку ако је само једна цифра неисправна или ако је редослед две цифре промењен.

Читач бар-кода сваки пут обавља израчунавање на основу кога би требало да добије контролну цифру, и уколико се добијена цифра разликује од прочитане контролне цифре, то указује на грешку.

На основу чега систем зна да ли је прочитан бар-код исправан бар-код?

Израчунавањем збира цифара које се налазе на непарним местима и троструке вредости цифара које се налазе на парним местима и једноструке вредности контролне цифре добија се број који уколико је резултат дељив са 10 показује да је прочитан исправан бар-код, у супротном да је дошло до грешке.

Да би се проверило да ли је дошло до грешке потребно је извршити следеће рачунање:

$$((a_1 + 3a_2 + a_3 + 3a_4 + a_5 + 3a_6 + a_7 + 3a_8 + a_9 + 3a_{10} + a_{11} + 3a_{12}) + a_{13}) \; mod \; 10$$

где су $a_1, a_2, \dots, a_{13}$ цифре које се налазе на бар-коду. Уколико добијени резултат није 0 то значи да је дошло до грешке приликом читања бар-кода или унете цифре не представљају цифре бар-кода.

Пример 7. Проверити исправност бар-кода са слике 8.

Ознака земље	Број произвођача	Број артикла	Контролни број
860	61033	5800	4

Слика 9. Пример бар-кода

Ако се погледа пример са слике 8 на основу претходно дате формуле добија се

$8 + 3 \cdot 6 + 0 + 3 \cdot 6 + 1 + 3 \cdot 0 + 3 + 3 \cdot 3 + 5 + 3 \cdot 8 + 0 + 3 \cdot 0 + 4 = 90$, а $90\ mod\ 10$ је 0, што значи да су ово заиста цифре неког бар-кода и није дошло до грешке наведеног типа приликом читања.

Задатак 3. Да ли је број 0 94187010890 1 исправан бар код?

Решење:

Првих једанаест цифара су 0 9418701089, док је контролна цифра 1. Проверавање збира цифара које се налазе на непарним местима и троструке вредости цифара које се налазе на парним местима добија се број:

$$(0 + 3 \cdot 9 + 4 + 3 \cdot 1 + 8 + 3 \cdot 7 + 0 + 3 \cdot 1 + 0 + 3 \cdot 8 + 9 + 3 \cdot 0) = 99$$

Додавањем контролне цифре на добијени збир $99 + 1 = 10$ закључује се да је бар код био валидан јер је 100 умножак броја 10, односно даје резултат 0 при модулу 10.

Како систем открива једноструке грешке?

Када читач бар-кода анализира бар-код, треба му делић секунде да израчуна контролну цифу и да упореди са прочитаном, оном која постоји на бар-коду. Уколико је оштећена само једна цифра, читач бар-кода чак може да израчуна оштећену цифру реверзибилним поступком израчунавања контролне цифре.

Пример 8. Нека је скенер прочитао број 4#02030145620 где # представља нечитљиву цифру. Процес израчунавања непознате цифре би био следећи:

$$(4 + 3 \cdot x + 0 + 3 \cdot 2 + 0 + 3 \cdot 3 + 0 + 3 \cdot 1 + 4 + 3 \cdot 5 + 6 + 3 \cdot 2) + 0 =$$
$$(4 + 0 + 0 + 0 + 4 + 6) + 3 \cdot (x + 2 + 3 + 1 + 5 + 2) + 0 = 14 + 3 \cdot x + 39 = 53 + 3x$$

Добијени збир треба да даје остатак нула при дељењу са 10. Додавањем броја 7, 17, 27, 37,... на 53 добија се број дељив са 10, али од предложених бројева треба одабрати онај који представља умножак броја 3 и неке цифре. Закључује се да је тражена цифра 9, јер она задовољава једначину $53 + 3x \equiv 0\ mod\ 10$.

Пример 9. Ако је приликом читања бар-кода

дошло до грешке у читању четврте цифре и уместо цифре 1 прочитана је цифра 2. Захваљујући рачуници коју сваки пут обавља скенер утврдиће се да је дошло до грешке.

$$(0 + 3 \cdot 9 + 4 + 3 \cdot 2 + 8 + 3 \cdot 7 + 0 + 3 \cdot 1 + 0 + 3 \cdot 8 + 9) + 1 = 103$$

Пошто добијени резултат није дељив са 10, то значи да је дошло до неке грешке приликом читања бар кода.

3.1.Контролна цифра

На пример нека су цифре бар кода 0360002914522. Израчунавањем контролне суме се добија

$$(0 + 3 \cdot 3 + 6 + 3 \cdot 0 + 0 + 3 \cdot 0 + 2 + 3 \cdot 9 + 1 + 3 \cdot 4 + 5 + 3 \cdot 2 + 2)\ mod\ 10$$
$$= 70\ mod\ 10 = 0$$

Што би даље значило да није дошло до грешке.

Предност коришћења контролне цифре јесте што се откривају све грешке које се тичу једне цифре и скоро све грешке које се тичу премештања две суседне цифре. Уколико цифре са друге и треће позиције бар кода 0360002914522 замене места рачуницом ће се добити $76\ mod\ 10 = 6$, што значи да је дошло до грешке.

Међутим, ако су цифре бар кода 016000291454 контролна цифра ће бити $60\ mod\ 10 = 0$, што значи да није дошло до грешке. Шта би се десило уколико би дошло до пермутовања цифара које се налазе на другом и трећем месту? Бар код би изгледао 061000291454 и његова контролна цифра би била $50\ mod\ 10 = 0$. Рачуница показује да није дошло до грешке, али очигледно да грешка постоји. Зашто ова грешка није откривена? Грешке типа замене суседних цифара које се разликују за 5 никада не могу бити откривене овим методом.

Зашто може доћи до лажног прихватања бар кода као исправно прочитаног?

Ако цифре a_i и a_{i+1} замене места укупан збир би се променио за $3 \cdot a_i + a_{i+1} - 3 \cdot a_{i+1} - a_i = 2 \cdot (a_i - a_{i+1})$, што би значило да ако је разлика две узастопне цифре једнака 5, односно $|a_i - a_{i+1}| = 5$, збир би се променио за ± 10. Пошто је $10\ mod\ 10 = 0$, укупан збир по модулу 10 се не би променио.

3.2. Грешке при преносу података

При преносу података често долази до промене појединих битова података због сметњи на преносном путу или различитих типова шумова на локацијама одашиљања и пријема. Сметње се дешавају без обзира на удаљеност уређаја, тј. како при преносу података између два рачунара тако и између компоненти истог рачунара.

Постоје два основна приступа решавању овог проблема, а то су: **контрола грешака унатраг** (контрола са повратном спрегом) и **контрола грешака унапред**. Код контроле грешака унатраг поред података шаљу се и додатне информације које служе да се установи да ли постоје грешке, али не и да се оне отклоне. Неисправно пренети подаци се поново шаљу. Контрола грешака унапред такође захтева слање додатних информација које служе да се установи да ли грешке постоје и да се одреди њихова локација. Неисправно пренети подаци се аутоматски коригују (негирањем примљених информација).

Како расте количина пренесених битова тако се повећава и број контролних битова. Контрола грешака унапред захтева преношење много веће количине додатних информација па се зато у оквиру једног система (нпр. меморија) често употребљава метод контроле унапред, док се код преноса информација међу системима (нпр. телекомуникације) обично употребљава метод контроле унатраг.

Најћешће коришћене **методе за откривање грешака** су: **контрола парности**, **провера збира блока** и **циклична провера редунданци**.

Контрола парности је један од најједноставнијих метода за откривање грешака. Уз сваку $n-$ битну реч додаје се по један бит тако да укупан број бинарних јединица у тако проширеној речи буде паран. Слабост ове методе је што се грешка не примећује уколико је број измењених битова паран. Постоји и **контрола парности у $2D$** при чему се описани метод проширује на следећи начин: свака основна реч се проширује на већ описани начин ради хоризонталне провере парности, али се и читавом блоку дадаје још једна реч тако да за сваку вредност бита постоји додатна (веертикална) провера парности. Овим методом се значајно умањује вероватноћа неоткривених грешака.

Пример 10. Пример кода који служи за откривање грешака, али не може да служи за исправљање грешака.

Рецимо да су слова енглеске абецеде представљена ASCII кодом, где слово A има вредност 65, слово B 66, односно у бинарном запису 10000001, 10000010 итд. Да би се открише грешке, додаје се осми бит сваком од карактера енглеске абецеде. Тај бит се назива бит парности или цифра парности. Бит парности се налази на крајњој левој позицији и рачуна се тако да вредност свих осам битова буде паран број. На пример, с обзиром да је збир битова слова A $1+0+0+0+0+0+0+1=2$, што је већ паран број, бит парности ће због тога бити 0, па ће слово A имати бинарну вредност 010000001.

Табела 1. Цифра парности

СЛОВО	ДЕКАДНИ ЗАПИС ASCII КОДА	БИНАРНИ ЗАПИС ASCII КОДА	+ ЦИФРА ПАРНОСТИ
A	65	10000001	010000001
B	66	10000010	010000010
C	67	1000011	11000011
…			

Овај метод је сличан као контролна цифра која се користи код бар-кода. Такође се наилази на исти проблем лажног прихватања поруке са грешком као исправне.

Пример 11. Пример $2D$ контроле парности.

	Битови парности	ASCII код	Карактер
	0	1000010	B
	1	1000101	E
	1	1001111	O
	0	1000111	G
	1	1010010	R
	0	1000001	A
	0	1000100	D
Реч парности	1	1011000	

Ако је блок дужи од једне речи често се примењује метод **контроле збира** чији је алгоритам: формира се збир свих речи у блоку и пренесе се заједно са поруком. Прималац поново израчуна збир и пореди са примљеном поруком. Слабости метода су што не може да препозна грешке неисправног редоследа речи, као ни додавање или губљење јединица блока чији су сви битови бинарне нуле.

Циклична провера редунданци (енгл. Cyclic Redundancy Checking- CRC) открива све грешке на непарном броју битова, све двобитне грешке, као и грешке чија је дужина мања од броја редундантних битова. Овај метод се често имплементира хардверски, а почива на аритметици по модулу 2 ($0+0=1+1=0; 0+1=1+0=1; 1-1=0-0=0; 0-1=1-0=1$) и дељењу полинома по модулу 2. Низ битова блокова који се преноси посматра се као низ коефицијената полинома, нпр. $a_n a_{n-1} \dots a_1 a_0$ одговара полиному $M(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x^1 + a_0$.

Поступак кодирања се састоји од:

1. Одабира полинома генератора $G(x)$ степена k
2. Израчунавања $x^k M(x)/G(x)$; добијени остатак се означава са $R(x)$
3. Коефицијенти остатака (њих k) се додају на крај поруке

Поступак декодирања:

1. Примљена полиномијална кодна реч се дели са $G(x)$
2. Ако је остатак дељења 0, нема грешака при преносу
3. Ако остатак није 0, постоје грешке при преносу

Добрим одабиром полинома генератора смањује се број грешака које се овако не могу открити. Добри полиноми генератори су:

$CRC-16 = x^{16} + x^{15} + x^2 + 1$

$CRC-CCITT = x^{16} + x^{12} + x^5 + 1$

$CRC-CCiTT = x^{32} + x^{26} + x^{23} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + 1$

Пример 11. Нека је ниска битова коју треба послати: 11100110 и нека је полином генератор $G(x) = x^4 + x^3 + 1$. Која ниска се шаље примаоцу? Како изгледа провера исправности пријема?

На бит се додају четири нуле, а након тога се дели:

$$
\begin{array}{r}
111001100000 \\
\underline{-11001} \\
\quad 1011100000 \\
\quad \underline{-11001} \\
\quad\ 111000000 \\
\quad \underline{-11001} \\
\quad\quad 1010000 \\
\quad\quad \underline{-11001} \\
\quad\quad\quad 110100 \\
\quad\quad\quad \underline{-11001} \\
\quad\quad\quad\quad 0110
\end{array}
$$

На оригиналну ниску се дописује остатак 0110 и добија се 111001100110.

$$
\begin{array}{r}
111001100110 \\
\underline{-11001} \\
\quad 1011100110 \\
\quad \underline{-11001} \\
\quad\ 111000110 \\
\quad \underline{-11001} \\
\quad\quad 1010110 \\
\quad\quad \underline{-11001} \\
\quad\quad\quad 110010 \\
\quad\quad\quad \underline{-11001} \\
\quad\quad\quad\quad\quad 0
\end{array}
$$

Пошто је остатак нула, пренос података је био исправан.

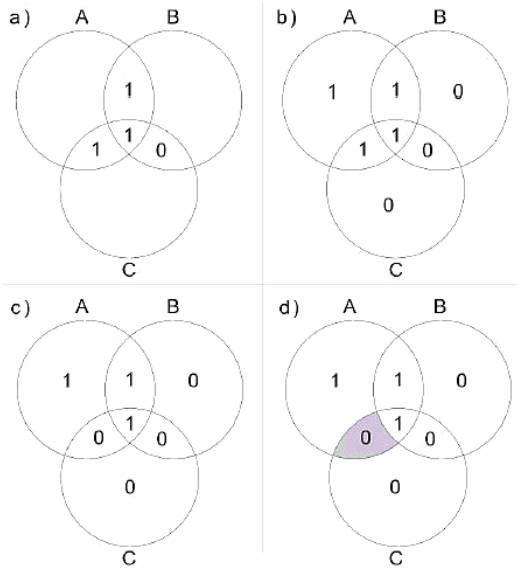
Методи за откривање и исправљање грешака

Користи се се често у раду са меморијом јер ту осим при преносу постоји и могућност настајања грешке при записивању и читању података, као и током њиховог чувања. Наиме, неки оод битова података може да буде промењен после записивања података. Међутим, пренос података од првобитног извоеа не може да се понови, јер у највећем броју случајева извор више не постоји. Стога се подаци тако записују да је при њиховом читању омогућено не само откривање него и корекција пронађене грешке.

Врсте кодова су: **SED** (Single error detection) – код који омогућава детекцију грешака на једном биту (слично DED, TED, …), **SEC** (Single error correction) који омогућава корекцију грешака на једном биту (слично DEC, TEC, …).

Најпростији кодови за откривање и корекцију грешака је Хамингоов код. Наредни пример представља пример за речи дужине 4 (3 додатна бита).

Пример 12. Хамингов код за речи дужине 4. У поље које представља пресек бар два круга упише се по једна вредност бита из речи. У преостала поља се уписује 0 или 1 тако да се у сваком кругу очува парност.

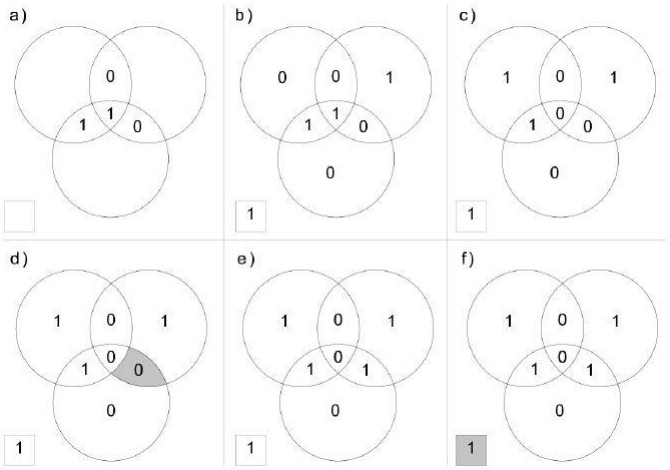


a) Битови података
b) Са битовима парности
c) Промена бита услед грешке
d) Локализована грешка

SEC-DED кодови

Често се SEC кодови проширују тако да осим SEC обављају и посао DED. Обично је довољно додавање свега једног бита. Меморија најчешће поседује уграђени SEC-DED.

Пример 13. Хамингов SEC-DED код.



a) Битови података
b) Битоови парности
c) Промењена два бита
d) Локализована грешка
e) Исправљена грешка
f) Препознато да постоји грешка

3.4.Математичка шифровања

Прва врста шифровања, која је била изложена у поглављу I Кратка историја криптографије, јесте шифровање померањем. Сам назив говори о томе на који начин се шифрује нека порука. Наиме, једноставно се слово А помери за неколико места удесно, тако да алфабет почиње одатле, настављајући тако у круг до последњег слова. Начин на који се померени алфабет повезује са алфабетом јесте шифра.

Ако би се померај вршио улево уместо удесно, добила би се друга шифра, али с обзиром да су помераји циклични, сваки померај улево се може представити померајем удесно само за други број.

Иста врста шифровања може се применити ако се свако слово представи бројем: $\text{А} - 1, \text{Б} - 2, \text{В} - 3$ и тако до $\text{Ш} - 30$. На овај начин се шифровање са померајем n удесно може добити простим увећавање бројева за n.

На овај начин може се уочити да се добијају и бројеви већи од 30, а којима не одговара ни једно слово у азбуци. Међутим, треба увек имати на уму да се врши померање слова око слова Ш, тако да ни њихове бројчане репрезентације не могу бити веће од нумеричке репрезентације слова Ш, односно више од 30. Овај проблем се решава тако што се од добијеног збира који је већи од 30 одузме 30 и тако добијени број представља нумеричку репрезентацију слова које је добијено померањем датог слова за n места. Иза овога се крије модуларна аритметика.

На жалост, овакав начин шифровања је изузетно рањив, па неке од метода којима се повећава сигурност шифре су: прегруписавање бројева у блокове одређене дужине, уклањање цртица како се не би видело где почиње један карактер и које цифре представљају једно слово.

3.5.Уопштено о математичким шифрама

До сада спомињана шифровања су веома слаба и несигурна. Уколико се она поткрепе математиком могу прерасти у много јаче шифре.

Шифровање азбуке са померајем 14, математички се може представити као

$$C \equiv (P + 14)\ mod\ 30$$

где је P основни текст који се шифрује, C шифровани текст. Да би се текст дешифровао једноставно се израчуна:

$$P \equiv (C - 14)\ mod\ 30$$

У суштини могуће је изабрати било који модуо и било који померај. Такође, нема потребе да се сваки карактер понаособ шифрује већ је могуће шифровати по групама.

Пример 14. Нека је потребно шифровати следећу поруку:

ТЕШКО ЗЕМЉИ КУДА ПРОЂЕ ВОЈСКА.

Порука има 25 карактера, па ће бити подељена у групе од по 5 карактера. Након превођења слова у бројеве, сваки блок ће имати до десет карактера. Порука кодирана у нумеричку вредност слова:

Т	Е	Ш	К	О	З	Е	М	Љ	И	К	У	Д	А	П	Р	О	Ђ	Е	В	О	Ј	С	К	А
22	7	30	12	18	9	7	15	14	10	12	24	5	1	19	20	18	6	7	3	18	11	21	12	1

Највећи број је 227301218, тако да ће шифровање за било који модуо већи од овог броја радити. Нека то буде, на пример, број 300000000. Још је потребно одабрати померај и нека је то рецимо 100000000. Сада је потребно да се шифрује по формули

$$C \equiv (P + 100000000)\ mod\ 300000000$$

где је P било који блок од шест цифара.

Шифровањем се добија:

$$227301218\ +\ 100000000 = 327302218 \equiv 27302218\ mod\ 300000000$$

$$97151410\ +\ 100000000 = 197151410 \equiv 197151410\ mod\ 300000000$$

$$12245119\ +\ 100000000 = 112245119 \equiv 112245119\ mod\ 300000000$$

$$2018673\ +\ 102018673 = 102018673 \equiv 102018673\ mod\ 300000000$$

$$181121121\ +\ 281121121 = 281121121 \equiv 281121121\ mod\ 300000000$$

Тиме је почетни текст шифрован са

$$27302218197151410112245119102018673281121121.$$

3.6.Шифровање множењем

За разлику од претходног шифровања за које је коришћено модуларно сабирање, за наредна шифровања биће коришћене шифре за које се користи модуларно множење, које је мало сложеније. Редослед приказивања различитих врста шифровања води ка RCA шифровању, које се данас највише користи.

Претпоставимо да је P текст који је потребно шифровати, m модуо и c је неки цео број. Тада се рачунањем

$$C \equiv c \cdot P \ mod \ n$$

може генерисати шифровани текст C. Поставља се питање како се текст шифрован на овај начин дешифрује, односно како се поништава модуларно множење. Одговор је, наравно, модуларно дељење.

Псеудокод за шифровање множењем би био сличан оном за шифровање померајем, само што би се уместо сабирања користила операција множења.

Пример 15. Нека Милица жели да пошаље Лазару поруку БОЉА ЈЕ СРЕЋА НО ПАМЕТ користећи ширфовање модуларним множењем где је $n = 2500$ и $c = 3$:

Као и обично потребно је поруку превести у бројеве, па потом разбити у блокове.

Б	О	Љ	А	Ј	Е	С	Р	Е	Ћ	А	Н	О	П	А	М	Е	Т
2	18	14	1	11	7	21	20	7	23	1	16	18	19	1	15	7	22

Односно, порука разбијена на блокове од два броја ће изгледати:

$$218\ 141\ 117\ 2120\ 723\ 116\ 1819\ 115\ 722$$

Затим се сваки од блокова множи са $c = 3$, што износи:

$$654\ 423\ 351\ 6360\ 2169\ 457\ 345\ 2160$$

и на крају се рачуна колико је сваки блок по модулу $n = 2500$:

$$654\ 423\ 351\ 1360\ 2169\ 457\ 345\ 2160$$

Сада, Лазар, мора да дешифрује поруку. Наравно, она зна да је $n = 2500$ и $c = 3$, али се поставља питање како се дели по модулу. Требало би размотрити шта је инверз броја 3 по модулу 2500. Приметимо да је

$$3 \cdot 1667 \ = 5001 \equiv 1 \ mod \ 2500$$

што значи да ако је $C \equiv 3 \cdot P \ mod \ 2500$, онда је

$$1667 \ \cdot C = 1667 \cdot 3 \cdot P \equiv 1 \cdot P \ mod \ 2500$$

Односно, да би се поништило множење са 3 по модулу 2500, потребно је да се израз помножи са $c^{-1} = 1667$. Ознака c^{-1} означава да су 3 и 1667 инверзни по модулу 2500 (као што је на пример инверз броја 7 у скупу рационалних бројева $7^{-1} = \frac{1}{7}$).

Дакле, Лазар примајући поруку облика 654 проналази инверз броја c $(= 3)$, што је у овом случају $c^{-1} = 1667$ и рачуна $c^{-1} \cdot C$, што је:

$$1667 \cdot 654 \ = 1090218$$

и након тога проналази $c^{-1} \cdot C \ mod \ n$, тј.

$$1090218 \equiv 218 \ mod \ 2500$$

Настављајући исти рачун за други примљени блок добија се:

$$1667 \cdot 423 = 705141 \equiv 141 \ mod \ 2500$$

Следећи овај поступак и за преостале бројеве долази се до

$$218\ 141\ 117\ 2120\ 723\ 116\ 1819\ 115\ 722$$

што је управо послата порука.

Из овог примера се чини да једино што је потребно да би се дешифровала порука јесте да се пронађе инверз c^{-1}. Међутим, шта би се десило да је уместо $c = 3$ био изабран број $c = 2$? У том случају не би могла да се дешифрује порука из разлога што не постоји инверз броја 2 по модулу 2500, јер је $2 \cdot 1250 = 2500 \equiv 0 \ mod \ 2500$. Уколико би постојао инверз d, важило би да је $2 \cdot d = 1 \ mod \ 2500$. Комбиновањем две конгруенције се добија

$$0 \equiv d \cdot 0 \equiv d \cdot 2 \cdot 1250 \equiv 1 \cdot 1250 \equiv 1250 \ mod \ 2500$$

Пошто не важи да је $0 \equiv 1250 \ mod \ 2500$, значи да је број 2 нема инверз по модулу 2500.

Алгоритам за рачунање модуларног инверза

Неки елемент $x \in \boldsymbol{Z_m}$ има мултипликативни инверз $(1/x) = x^{-1}$ у $\boldsymbol{Z_m}$ ако је $nzd(x, m) = 1$: пошто се $nzd(x, m) = 1$ изрази у облику целобројне линеарне комбинације $1 = ax + bm$, види се да је $ax = 1 \; mod \; m$.

Пример 16. $(1/2) = 2^{-1} = 3 \; mod \; 5$, јер је $2 \cdot 3 = 1 \; mod \; 5$.

Пример 17. Пронаћи инверз броја 7 по модулу 9.

Коришћењем проширеног Еуклидовог алгоритма добија се:

$$9 = 1 \cdot 7 + 2$$

$$7 = 3 \cdot 2 + 1$$

$$2 = 2 \cdot 1 + 0$$

па је

$$1 = 7 - 3 \cdot 2$$

$$1 = 7 - 3(9 - 7)$$

$$1 = 4 \cdot 7 - 3 \cdot 9$$

Ако се ова једнакост посматра по модулу 9 (то може јер је $a \equiv a \; mod \; m$). Добија се да је

$$1 = 4 \cdot 7 - 3 \cdot 9; 1 \equiv 4 \cdot 7 - 3 \cdot 0 \; mod \; 9 \equiv \; 4 \cdot 7 \; mod \; 9$$

На основу тога је $1 \equiv 4 \cdot 7 \; mod \; 9$, односно $7^{-1} \equiv 4 \; mod \; 9$.

Пример 18. Чему је једнако $\frac{2}{7}$ по модулу 9?

$$\frac{2}{7} = 2 \cdot \frac{1}{7} \equiv 2 \cdot 4 \; mod \; 9 = 8 \; mod \; 9$$

Следи псеудокод на основу кога проблем проналажења модуларног инверза може бити решен помоћу рачунара:

Нека су a и m природни бројеви тако да важи $1 \leq a < m$ и $nzd(a,m) = 1$. Алгоритам израчунава y као инверз броја a по модулу m:

begin

$y \coloneqq 0; v \coloneqq 1; g \coloneqq m; r \coloneqq a;$

$pr \coloneqq r; q \coloneqq \left[\frac{g}{pr}\right]; r \coloneqq g - pr\ q;$

(целобројно дељење g са pr, да би се добило $g = prq + r$)

if $r = 0$ ***then***

$y \coloneqq 1; g \coloneqq pr$

else

$r = pr;$

while $r \neq$ ***do***

$pr \coloneqq r; pv \coloneqq v;$

$q \coloneqq \left[\frac{g}{pr}\right]; r \coloneqq g - prq;$

$v \coloneqq y - pvq;$

if $v < 0$ ***then***

$v \coloneqq m - ((-v) mod\ m)$

else

$v \coloneqq v\ mod\ m$

endif

$g \coloneqq pr; y \coloneqq pv$

endwhile;

endif;

$ivnerz(a) \coloneqq y$

end

IV ОЈЛЕРОВА ФУНКЦИЈА

У циљу увођења још неких метода шифровања у оквиру овог поглавља биће обрађена Ојлерова функција. Ова функција је значајна за различите врсте шифровања, међу којима се може издвојити РСА метод шифровања као најдоминантнији и најсигурнији у садашњем тренутку.

1. Z_n и Ојлерова φ функција

У поглављу о модуларној артметици било је показано како изгледају рачунске операције у Z_n. Ако се одабере један елемент из Z_n да ли је могуће да се узастоним додавањем тог истог елемента добију сви елементи Z_n? Покушајмо са бројем 9 из Z_{12}.

Дакле, почиње се са 9

Онда $9 + 9 \equiv 6 \ mod\ 12$,

па $9 + 9 + 9 \equiv 3 \ mod\ 12$,

даље је $9 + 9 + 9 + 9 \equiv 0 \ mod\ 12$

и видимо уколико бисмо наставили са додавањем 9 добили бисмо поново бројеве $0, 3, 6, 9$, али ово нису једини елементи Z_{12}.

Сада покушајмо исто само са бројем 5

$5 + 5 \equiv 10 \ mod\ 12$,

$5 + 5 + 5 \equiv 3 \ mod\ 12$,

$5 + 5 + 5 + 5 \equiv 8 \ mod\ 12$,

$5 + 5 + 5 + 5 + 5 \equiv 1 \ mod\ 12$,

$5 + 5 + 5 + 5 + 5 + 5 \equiv 6 \ mod\ 12$,

$5 + 5 + 5 + 5 + 5 + 5 + 5 \equiv 11 \ mod\ 12$,

$5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 \equiv 4 \ mod\ 12$,

$5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 \equiv 9 \ mod\ 12$,

$5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 \equiv 2 \ mod\ 12$,

$5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 \equiv 0 \ mod\ 12$.

Видимо да су овога пута генарисани сви елементи Z_{12}, редом $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$ и 11.

Дакле, кренувши од броја 9 добијају се бројеви $\{0, 3, 6, 9\}$, али ако се крене од 5 добијају се сви елементи $\{0,1, 2, 3,4,5, 6,7,8, 9,10,11\}$. Зашто постоји разлика у броју елемената? Одговор је зато што бројеви 9 и 12 нису узајамно прости, док су бројеви 5 и 12 узајамно ппрости. Тако да уколико желимо да нађемо елемент који генерише све елементе Z_n

потребно је да пронаћи број k такав да је $nzd(k,n) = 1$, односно број k који је узајамно прост са бројем n.

Сада се поставља питање колико елемената $\boldsymbol{Z_{12}}$ може да генерише остале елементе $\boldsymbol{Z_{12}}$? Односно, колико је бројева из $\boldsymbol{Z_{12}}$ узајамно просто са бројем 12? За мале бројеве лако је проверити све могућности и видети за који број ово важи. Међутим овакав начин постаје немогућ ако се ради о великим бројевима. Постоји много лакши начин да се ово уради и укључује Ојлерову φ функцију.

Дефиниција 1. Број природних бројева који нису већи од датог природног броја m и узајамно су прости са са њим, означава се са $\varphi(m)$. Функција φ назива се **Ојлерова функција** [5].

Неколико првих вредности функције φ:

m	1	2	3	4	5	6	7	8	9	10	11	12
$\varphi(m)$	1	1	2	2	4	2	6	4	6	4	10	4

Теорема 1. Нека је p прост број, k позитван цео број и $nzd(a,b) = 1$, тада важе следеће особине Ојлерове функције:

1. $\varphi(p) = p - 1$
2. $\varphi(p^k) = p^{k-1}(p-1)$
3. $\varphi(ab) = \varphi(a)\varphi(b)$

Позивајући се на Основну теорему аритметике, која говори да се сваки позитиван цео број може на јединствен начин представити као производ степена простих бројева, за позитиван цео број n важи:

$$n = {p_1}^{k_1}{p_2}^{k_2}{p_3}^{k_3} \dots {p_s}^{k_s}$$

Пошто су било која два проста природна броја уједно и узајамно проста, користећи друго својство Ојлерове функције добија се да је

$$n = {p_1}^{k_1-1}(p_1-1){p_2}^{k_2-1}(p_2-1)\dots{p_s}^{k_s-1}(p_s-1)$$

Сада, коришћењем Ојлерове функције може се лако одговорити на питање колико има елемената који генеришу целу групу? Одговор је:

$$\varphi(12) = \varphi(2^2 \cdot 3) = 2(2-1)3^0(3-1) = 4$$

Може се и врло лако одговорити на питање које изгледа сувише тешко чак и коришћењем рачунара. Колико има бројева који су узајамно прости са 100000000000?

Пошто је $100000000000 = 10^{11} = 2^{11} \cdot 5^{11}$, Ојлерова функција износи $\varphi(1000000000) = 2^{10} \cdot (2-1)5^{10} \cdot (5-1) = 4 \cdot 10^{10}$.

Задатак 1. Израчунај: $\varphi(43), \varphi(100), \varphi(455)$.

Решење:

1. $\varphi(43) = 42$, јер је 43 прост број (Прва особина Ојлерове функције)
2. $\varphi(100) = \varphi(25 \cdot 4) =_{\text{трећа особина О.ф.}} \varphi(25) \cdot \varphi(4)$
   $= \varphi(5^2) \cdot \varphi(2^2) =_{\text{друга особина О.ф.}} 5 \cdot 4 \cdot 2 \cdot 1 = 40$
3. $\varphi(455) = \varphi(19 \cdot 25) = \varphi(19) \cdot \varphi(25) = 18 \cdot \varphi(5^2) = 18 \cdot 5 \cdot 4 = 360$

2.Експоненцијална шифра

Као што и само име каже, потребно је да се основни текст P шифрује тако што ће се наћи P^s након чега се израчуна колики је тај број по модулу n.

Пример 1. Милица жели да пошаље Лазару поруку користећи експоненцијално шифровање за $c = 3$ и $n = 41$

НЕКА БУДЕ БОРБА НЕПРЕСТАНА

С обзиром да експоненцијална функција расте веома брзо, слова након претварања у бројеве ће бити груписана, тако да ће порука преведена у нумеричку вредност слова изгледати:

16 7 12 1 2 24 5 7 2 18 20 2 1 16 7 19 20 7 21 22 1 16 1

Следеће што је потребно јесте да се сваки од бројева подигне на трећи степен, што даје

4096 343 1728 1 8 13824 125 343 8 5832 8000 8 1 4096 343 6859 8000 343 9261 10648 1 4096 1

И да се израчуна колико сваки од добијених бројева износи по модулу 41

37 15 6 1 8 7 2 15 8 10 5 8 1 37 15 12 5 15 36 29 1 37 1

Да би се примљена порука $C \equiv P^e \ mod \ n$ дешифровала потребно је још мало теорије.

Теорема 2. (Мала Фермаова) Нека је n позитиван цео број. Тада за било који цео број a који је узајамно прост са n важи:

$$a^{n-1} \equiv 1 \ (mod\ n).$$

Доказ

Нека су $a, 2a, 3a, \dots, (n-1)a$ цели бројеви и $r_1, r_2, r_3, \dots, r_{n-1}$ низ остатака који је добијен дељењем бројева из првог низа са n. Пошто важи да је $nzd(a,n) = 1$, ниједан од бројева из првог низа није дељив са n и важи да је $1 \leq r_i \leq n-1$ за $i = 1, \dots, p-1$. Потребно је доказати да су ови остаци међусобно различити. Уколико не би били различити и рецимо $r_i = r_j$ за $1 \leq i < j \leq n-1$. Пошто је $ai = r_i \ mod\ n$ и $aj = r_j \ mod\ n$ може се закључити да је $aj - ai = r_j - r_i \ mod\ n$. На основу претпоставке да је $r_i = r_j$ добија се $aj - ai = 0 \ mod\ n$, што би занчило да n дели $a(j-i)$, те на основу претпоставке теореме да је $nzd(a,n) = 1$ важило би да n дели $j-i$. Пошто је $1 \leq i < j \leq n-1$, а самим тим и $1 \leq j-i \leq n-1$, долази се до контрадикције, односно претпоставка да су остаци једнаки није тачна. Дакле, постоји $n-1$ различитих остатака који су различити од нуле и отуда важи да су они баш $\{r_1, r_2, \dots, r_{n-1}\} = \{1,2, \dots, n-1\}$. На основу својстава конгруенције добија се $a \cdot 2a \cdot 3a \cdots (n-1)a = 1 \cdot 2 \cdot 3 \cdots (n-1) \ mod\ n$, што је даље $(a^{n-1} - 1) \cdot (n-1)! = 0 \ mod\ n$. Поново n дели $(a^{n-1} - 1) \cdot (n-1)!$, али пошто су n и $(n-1)!$ узакамно прости бројеви, следи да n дели $(a^{n-1} - 1)$, што је и требало доказати.■

На основу теореме 2. да би се примљена порука дешифровала потребно је да се пронађе

$f = e^{-1} \ mod\ \varphi(n)$, јер је $C^f \equiv (P^e)^f$

$$\equiv (P)^{k \cdot \varphi(n)+1}, \text{за неко } k$$

$$\equiv (P^{\varphi(n)})^k \cdot P$$

$$\equiv 1 \cdot P$$

$$\equiv P \ mod\ n$$

и израчуна C^f.

Што значи да би Лазар дешифровао поруку потребно је да израчуна $3^{-1} mod\ \varphi(41)$. На основу својстава Ојлерове функције $\varphi(41) = 40$, па је потребно наћи инверз броја 3 по модулу 40.

На основу Еуклидовог алгоритма: $40 = 13 \cdot 3 + 1$, односно $1 = 40 - 13 \cdot 3$, тј. $1 \equiv -13 \cdot 3 \ (mod\ 40)$, односно $1 \equiv 27 \cdot 3 \ (mod\ 40)$. Дакле, пронађен је инверз броја 3 по модулу 40, а то је $27 \equiv 3^{-1} \ (mod\ 40)$.

Биће показано како дешифровање изгледа на једном броју, а принцип је исти за све бројеве. Нека је, на пример, то број 5. Потребно је да се израчуна колико је $5^{27} mod\ 41$. Пошто је $5^{27} = 5^{3^{3^3}}$, кренуће се од рачунања $5^3 mod\ 41, 5^3 = 125 \equiv 2\ mod\ 41$, даље је $2^3 = 8 \equiv 8\ mod\ 41$ и на крају $8^3 = 512 \equiv 20\ mod\ 41$, чиме је дешифрован почетни број 5.

За крај се поставља питање колико пажње треба да би се одабрали n и e за неко експоненцијално шифровање. За почетак треба инсистирати да n буде прост број, или да буде производ два проста броја, на пример $n = p \cdot q$. На основу теореме о особинама Ојлерове функције φ, види се да e које се бира мора да буде узајамно прост број са $\varphi(n) = (p-1) \cdot (q-1)$.

3.RCA шифровање

Веома важна примена модуларне аритметике је у RCA енкрипцији. Ако је потребно послати информацију на безбедан начин тако да јој други људи немају приступ, то је могуће обезбедити помоћу RCA енкрипције (на пример куповином преко интернета потребно је заштитити пин своје кредитне картице). Овај метод је изузетно сигуран, а назив је добио по почетним словима презимена људи који су га открили *Ron Rivest, Adi Shamir* и *Leon Adelman*. Метод је откривен 1977.

Код RCA енкрипције свака цифра кредитне картице се мења или енкриптује, тако да број који се шаље не личи на оригинални број. Када је енкриптована порука послата, прималац поруке треба да зна кључ да би декриптовао поруку коју је примио тако да може да реконструише оригиналну поруку. Метод је веома једноставан за коришћење и у исто време готово немогућ да се обори. Откривање кључа за декрипцију захтева факторисање огромних бројева, што ни најбољи рачунари не би могли даураде у реалном времену.

У наставку ће описано како ради ова метода, не улазећи у детаље, али довољно да опише како функционише и како се помоћу RCA енкрипције постиже циљ – безбедно слање и примање информација. Ако се нешто наручује преко интернета, на који начин је то осигурано?

RCA укључује три корака: генерисање кључа, шифровање и дешифровање. RCA је заиста невероватан алгоритам који користи ништа више од модуларне аритметике, коју је могуће савладати у средњој школи.

Генерисање кључа

РСА користи јавни и тајни кључ – јавни је онај који је доступан за све и њиме се порука шифрује, док је за дешифровање поруке потребан тајни кључ. Оба кључа се генеришу на следећи начин:

1. Изаберу се два различита (случајно одабрана) велика проста броја p и q. Требало би да имају више од 100 цифара и чувају се у тајности
2. Израчуна се $n = p \cdot q$. Овај број се користи као модуо за оба кључа и доступан је свима.
3. Одабере се један цео број e такав да је $1 < e < \varphi(n)$ и $nzd(e, \varphi(n)) = 1$, где је $\varphi(n)$ Ојлерова функција о којој је раније било речи. Број e ће бити коришћен као експонент јавног кључа.
4. Израчунати f тако да важи $fe \equiv 1\ mod\varphi(n)$. Ово је могуће увек израчунати јер је $nzd(e, \varphi(n)) = 1$ и важи да је f јединствени инверз од e. Број f се чува као експонент приватног кључа и отуда не сме бити свима доступан.

Јавни кључ се састоји од модула броја n и јавног експонента e, док се тајни кључ састоји од модула броја n и приватног експонента f.

Шифровање

Милица шаље свој јавни кључ (n, e) Лазару, док тајни кључ чува. Ако би Лазар желео Милици да пошаље поруку **П**, потребно је да прво претвори поруку у број $m < n$. Затим шифрује текст у c, тако да важи $c \equiv m^e\ mod\ n$. На крају Лазар шаље Милици шифровани облик своје поруке **П** - поруку c.

Дешифровање

Милица сада може да открије m из c користећи експонент свог приватнног кључа f на следећи начин:

$$m \equiv c^f\ mod\ n$$

Сада, на основу m може да открије оригиналну поруку П.

Претходно описани процес декрипције ради јер је

$c^f \equiv (m^e)^f mod\ n = m^{ef}\ mod\ n = m$ јер су f и e инверзни елементи.

Пример 2. Нека је број кредитне картице број 9. Методом RCA енкрипције потребно га је шифроовати.

1. Генерисање кључа

Изаберимо два проста броја: $p = 5$ и $q = 7$. Отуда је $n = p \cdot q = 35$. Ојлерова функција за израчунато n је $\varphi(n) = (p-1) \cdot (q-1) = 24$. Затим се одабере број e такав да је $nzd(e, \varphi(n)) = 1$. Изаберимо број $e = 5$. Даље, потребно је одредити f потребно је одредити инверз броја 5 по модулу $\varphi(35)$. На сонову Еуклидовог алгоритма:

$$24 = 4 \cdot 5 + 4$$
$$5 = 1 \cdot 4 + 1$$
$$4 = 4 \cdot 1 + 0$$

Ако изразимо 1 из друге једнакости добија се $1 = 5 - 1 \cdot 4$, ако се даље број 4 изрази из прве једнакости добија се $1 = 5 - 1 \cdot (24 - 4 \cdot 5) = 5 - 24 + 4 \cdot 5 = 5 \cdot 5 - 24$. Пошто је потребно одредити инверз по модулу 24, то значи да је $1 = 5 \cdot 5\ (mod\ 24)$, што даље значи да је инверз броја 5 по модулу 24 број 5, тј. $f = 5$. Након израчунатог експонента шифровања следи само шифровање поруке.

2. Шифровање

Оригиналну поруку треба подићи на степен $f = 5$ и израчунати остатак при дељењу са 35. Пошто је $9^5 \equiv 4 \bmod 35$, биће послата порука 4.

3. Дешифровање

На крају, да би се добила оригинална порука, потребно је примљену поруку 4, подићи на експоненет дешифровања и потом наћи остатак при дељењу са 35. Тиме се добија $4^5 \equiv 9 \bmod 35$, односно добија се да је оригинална порука била 9.

Пример 3. Одаберимо просте бројеве p и q. Нека је $p = 17$ и $q = 19$, онда је $n = pq = 323$. Следећи корак јесте израчунавање Ојлерове функције $\varphi(n) = 288$. Сада је потребно одредити e, експонент шифровања, нека је $e = 11$. Остало је да се израчуна експонент дешифровања f, тако да важи да је $ef = 1\ (mod\ 288)$. На исти начин као малопре проналазимо да је експонент приватног кључа $f = 131$.

Пример 4. Уколико неко жели да пошаље поруку „АБВ“, користећи ову шему (кључеве из примера 3.), прво је потребно да поруку преведу у нумерички облик. Нека „АБВ“ постане „123“. Да би шифровао поруку потребно је да зна $e = 11$. Сходно томе шифрована порука гласи:

$$123^{11} \ (mod\ 323) \equiv 81$$

Да би прималац дешифровао поруку потребно је да израчуна
$81^{131} \ (mod\ 323) \equiv 123,$
што и јесте била послата порука. На крају, с обзиром да зна на који начин су слова преведена у бројеве, може да одреди реверзибилан поступак и од „123“ добиће оригиналну поруку „АБВ“.

Пример 5. Следи још једна илустрација РСА. [9]

Запише се порука користећи велика слова енглеског алфабета A,B,C,…,Z након чега се та порука преводи у број тако да важи $A = 11, B = 12, \dots, Z = 36$. Рецимо да је Лазар одабрао два проста броја $p = 12553$ и $q = 13007$, тако да је $pq = 163276871$ и $(p-1)\cdot(q-1) = 163251312$. Лазар, такође проналази $e = 79921$, који је узајјамно прост са $(p-1)\cdot(q-1)$ и након тога се проналази инверз броја e по модулу $(p-1)\cdot(q-1)$ - број f користећи Еуклидова алгоритам. Број f ће у овом примеру износити $f = 145604785$. Сада предпоставимо да је Алберт примио поруку, издељену на групе од по девет цифара, јер $n = 163276871$ има девет цифара, која гласи

$$145387828 \quad 47164891 \quad 152020614 \quad 27279275 \quad 353356191$$

Следећи корак јесте да Лазар дешифрује поруку коришћењем свог тајног кључа (f, m), где је $f = 145604785$ и проналази да је

$$145387828^{145604785} \equiv 30182523\ mod\ 163276871$$
$$47164891^{145604785} \equiv 26292524\ mod\ 163276871$$
$$152020614^{145604785} \equiv 19291924\ mod\ 163276871$$
$$27279275^{145604785} \equiv 30282531\ mod\ 163276871$$
$$353356191^{145604785} \equiv 122215\ mod\ 163276871$$

Што значи да је дешифрована порука облика
$30182523 \ 26292524 \ 19291924 \ 30282531 \ 122215$
Коначно, превођењем сваке две цифре у одговарајуће слово енглеског језика добија се порука

$$THOMPSONISINTROUBLE$$

Односно Thompson is in trouble (Томсон је у невољи).

Из наведених примера се види да су бројеви са којима се барата све већи. Како би се рачунање скратило могу послужити рачунари.

Алгоритми за рачунање степена по модулу

Прво ће бити изложен алгоритам за ефикасно рачунање $x^n \ mod \ m$, где је $n \geq 1$. Идеја јесте да се поступак множења своди на рекурзију. Уколико је n паран број, односно $n = 2k$, онда је

$$x^n = x^{2k} = (x^k)^2$$

што значи да се x^k рачуна рекурзивно и на крају се квадрира. Уколико је n непаран број, рецимо $n = 2k + 1$ онда је

$$x^n = x^{2k+1} = (x^k)^2 x$$

Даље се x^k рачуна рекурзивно, квадрира се и множи са x.

Ако се број n напише у бинарном облику:

$$n = b_l \cdot 2^l + \ b_{l-1} \cdot 2^{l-1} + \cdots + b_1 \cdot 2^1 + b_0,$$

где је $b_i \in \{0,1\}$. Означимо са $J = \{j | b_j = 1\}$, тада се број n може представити као

$$n = \sum_{j \in J} 2^j,$$

Онда је $x^n = x^{\sum_{j \in J} 2^j} = \prod_{j \in J} x^{2^j} \ mod \ m$. Ако се са r_j означи $x^{2^j} = r_j (mod \ m)$, онда је $x^n = r_l \dots r_0 (mod \ m)$.

Пример 6. Израчунати $999^{179} mod 1763$.

Важи да је $179 = 2^7 + 2^5 + 2^4 + 2^1 + 2^0$.

На основу претходно показаног алгоритма израчунавају се редом степени:

$$999^{2^1} \equiv 143 \ (mod \ 1763)$$

$$999^{2^2} = 143^2 \equiv 1056 \ (mod \ 1763)$$

$$999^{2^3} = 1056^2 \equiv 920 \ (mod \ 1763)$$

$$999^{2^4} = 920^2 \equiv 160 \ (mod \ 1763)$$

$$999^{2^5} = 160^2 \equiv 918 \ (mod \ 1763)$$

$$999^{2^7} = 918^2 \equiv 10 \ (mod \ 1763)$$

$$999^{2^3} = 10^2 \equiv 100 \ (mod \ 1763)$$

Односно,

$$999^{179} \equiv 999 \cdot 143 \cdot 160 \cdot 918 \cdot 100 \ (mod\ 1763)$$

$$\equiv 54 \cdot 160 \ \cdot 918 \cdot 100 \ (mod\ 1763)$$

$$\equiv 1588 \cdot 918 \cdot 100 \ (mod\ 1763)$$

$$\equiv 1546 \cdot 100 \ (mod\ 1763)$$

$$\equiv 1219 \ (mod\ 1763)$$

На крају је $999^{179} = 1219 \ (mod\ 1763)$.

Претходно описани метод може бити имплементиран без претварања броја n у основу 2. Следи псеудокод на основу кога проблем израчунавања $x^n mod\ m$ може бити решен помоћу рачунара. Ако је n паран број, представља се као $n = 2k$, онда је $n/2 = k$ и ако је n непаран број, представља се као $n = 2k + 1$, онда је $(n - 1)/2 = k$.

begin

$u := 1;\ a := x;$

while $n > 1$ ***do***

If n паран број ***then*** $e := 0$ ***else*** $e := 1;$

If $e = 1$ ***then*** $u := a \cdot u\ mod\ m;$

$a := a^2\ \ mod\ m; n := (n - e)/2$

endwhile;

$u := a \cdot u\ mod\ m$

end [9]

Следи стварни пример РСА шифровања.

Пример 7. Стварни пример РСА шифровања. [10]

Сада, за прави пример РСА шифровања потребно је шифровати поруку „attack at dawn“. Прво је потребно поруку превести у нумеричку (сваком карактеру биће придружена његова ASCII вредност). После превођења порука ће гласити

$$19766202164023008896244827187751 50$$

Генерисање кључева

Сада се бирају два велика проста броја p и q. Ова два броја треба да буду случајно изабрана и не сувише близу један другом. Ово су два проста броја генерисана *Rabin-Miller*-овим тестом:

$$p = 12131072439211271897323671531612440428472427633 70$$

$$14109256345493123019643730420856193241973653224168665$$

$$41017057361365214171711713797974299334871062829803541$$

$$q = 12027524255478748885956220793734512128733387803 6$$

$$82075433653899983955179850988797899869146900809131611$$

$$15334681705083209602216014636634639181247098710541523 3$$

Сада на основу ова два броја може се израчунати број n и Ојлерова функција $\varphi(n)$.

n =14590676800758332323018693934907063529240187237535716439958187101987343879
900535893836957140267014980212181808629246742282815702292207674690654340122488
967247240792696998710058129010319931785875366371086235765651050788371429711563
734278891146353510271203276516651841172685983798867211183720508552634661874005
3

$\varphi(n)$ =14590676800758332323018693934907063529240187237535716439958187101987343
879900535893836957140267014980212181808629246742282815702292207674690654340122
488964831381123227996631730139777785236530154784827347887129722205858745715289
160645926971811926897116355507080264399952954964411681194751651393818429668352
1280

$e = 6553,$ јавни кључ

f =894894250092744443682285459217730939196695860658842574454978544564876748396298183909349419732628796167979706089172836798754993315741611138540888132754881105882471930775825272784379065040156806234235500672400424666656542323835029222154936232894721388664458187891279461234078077257026266440910365023725451397132, тајни кључ

Шифровање поруке

$$19766202164023008896244827187751 50^{e}\ (mod\ n)$$

што износи

350521113386730266902124239370533285118807608115799816206428023466858106231098502359430490809733862411137840407947041939782153784997654130836464387847409523069325349451950801838615742252262188798272324539128205968864403775360824656817500744174591514854074458625110234722355608230534977915189288202722577877 86

Дешифровање поруке

Потребно је израчунати:

$$350521113386730266902124239370533285118807608115799816206$$

$$428023466858106231098502359430490809733862411137840407 94$$

$$704193978215378499765413083646438784740952306932534945 19$$

$$508018386157422522621887982723245391282059688644037753 60$$

$$082465681750074417459151485407445862511023472235560823$$

$$053497791518928820272257787786^{e}\ mod\ n$$

Што износи 19766202164023008896244827187751 50 (заиста јесте ASCII код поруке која је послата).

Овај пример реалне примене РСА показује величину бројева који су коришћени у алгоритму.

Да би се разбио овај систем, потребно је наћи бројеве p и q, што захтева факторисање броја pq. Ако су p и q огромни, са рецимо више од 200 цифара, факторисање производа два броја од по перко 200 цифара представљао би и за суперкомпјутере вишемесечни задатак. При том, фирме које користе РСА енкрипцију, мењају бројеве p и q, неке чак и на дневном нивоу, како би разбијање кода учинили немогућим.

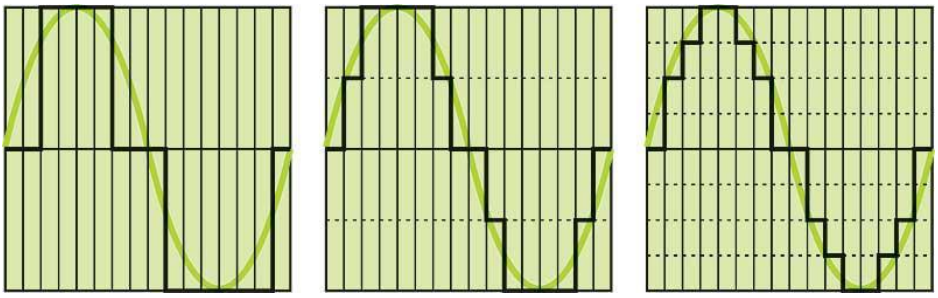
Могуће је слати све врсте порука РСА енкрипцијом. Текстуалне поруке се трансформишу у бројеве, користећи различите бројеве да представе свако слово, размак или запету. Порука се шаље као број (преведен у бинарни систем) и резултат се поново декриптује у речи.

V ЈОШ НЕКЕ ПРИМЕНЕ МАТЕМАТИКЕ У ТЕОРИЈИ КОДИРАЊА

Последње поглавље садржи интересантне приче о теорији кодирања које могу послужити као мотивација за даљу разраду и представљање истих или сличних тема. Ово поглавље илуструје ширину примене теорије кодирања.

1. Претварање звука у бројчани запис

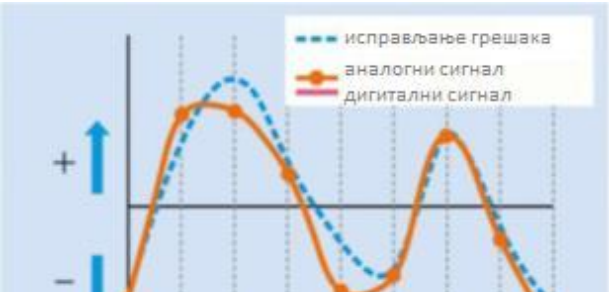
Звук је талас. Звук бубњева осцилује кроз ваздух и та осцилација допире до наших ушију. Извор звука може бити инструмент, гласне жице итд. Као и остали типови података, звук се у рачунару изражава низом нула и јединица, а производи се захваљујући уређају који се зове звучна картица. Пошто је звучни сигнал аналоган, потребно је сигнал превести у дигиталну форму. То се ради тако што се аналогни сигнал електронски узоркује (од енглеске речи *sample* - узорак) према одређеним временским интервалима. Сваки пут када се узме узорак, мери се амплитуда узорка, и та вредност се преводи у бинарну величину. Електронско коло које обавља ту функцију назива се **A-D конвертор** и налази се на већини звучних картица.



(а) (б) (в)

Слика 1. Аналогни и дигитални сигнал – узорковање

За разлику од аналогног сигнала који ако је искривљен, вредност која се исправља је непозната, за дигитални сигнал се зна да постоје само две могућности за исправљање, а то су нула и једница, што исправљање дигиталног сигнала чини лакшим.



Слика 2. Исправљање грешака код

аналогног и дигиталног сигнала

2. Кодови за исправљање грешака

У овом делу биће приказани методи за пренос података и исправљање грешака којима се бави теорија кодирања, која за разлику од криптографије нема за циљ да сакрије садржај поруке, већ да је на безбедан начин пренесе и исправи евентуалне грешке које су настале приликом преноса (на том принципу почивају бар-кодови). Неки од метода су већ поменути у поглављу II Бројевни системи.

2.1. Подаци из свемира

Како астронаут може бити сигуран да ће порука коју је послао из своје свемирске станице стићи на Земљу? На који начин се такве поруке шаљу на земљу? Како се обезбеђује поузданост информације?

Метод који је од историјског значаја за обезбеђивање сигурног преноса информација, а коришћен је када је откривен телефон, и данас је сачуван у неком облику. Наиме, саговорници преко телефона, уместо уобичајеног изговарања речи, за свако слово речи изговарали су нову реч која би за почетно слово имала слово те речи која би требало да се пошаље. Па тако, на пример, уместо да речи „ГРЕШКА" рећи ће: „Гледам Реку.. ". Овај метод је користила америчка и британска војска у Првом светском рату. Овај метод обезбеђује сигурност информација умножавањем информација. Техника умножавања информација налази се испод сваког метода за откривање и исправљања грешака.

2.2.Метод редундантних информација

Овај метод осим што може да открије грешку, може и да је исправи. Почива на

простом понављању елемената поруке. Подразумева троструко слање сваког карактера како би примљена порука могла да се исправи на адекватан начин.

Пример 2. Ако је порука која треба да се пошаље : „ПОМОЋ“ може бити кодирана тако што се сваки карактер понови по два пута, тако да би порука гласила : „ППООММООЋЋ“. Међутим, уколико би дошло до грешке у преносу последњег карактера, тако да примљена порука изгледа : „ППООММОРЋ“, коју реч одабрати као исправну? ПОМОР или ПОМОЋ?

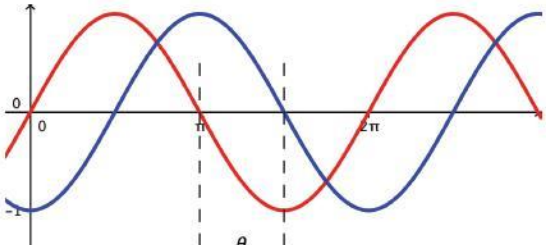
С обзиром да је једнака вероватноћа за обе речи, овакво кодирање које подразумева само дуплирање података може помоћи једино код откривања грешака, али не и код исправљања. Да би се грешка успешно исправила, потребно је три пута послати један исти бит. Тако уколико би дошло до грешке у преносу последњег карактера, била би примљена следећа реч: „ППООММООЋЋР“, која би била декодирана као ПОМОЋ.

На сличан начин се могу слати слике из свемира.

3. Пренос слике из свемира и корективни кодови

Слика 3. Функција $sinx$ *[1 1]*

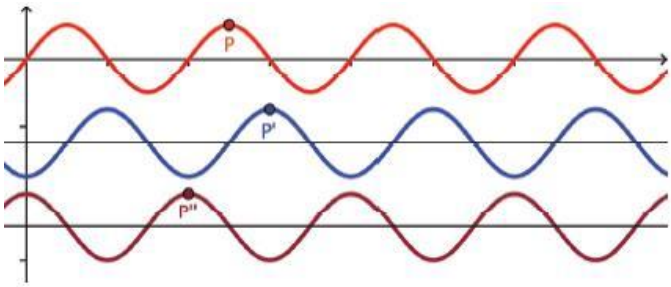
Функција $y = sinx$ и њене модификације су тајне преноса информација из свемира до наше планете. Електромагнетни таласи путују кроз свемир брзином од 300 000 km/s. Ништа не путује том брзином. Како ови радиоталаси преносе информације? Како ови таласи преносе низ нула и јединица? Да би се одговорило на ово питање потребно је изблиза погледати на график функције $sinx$. *Слика 4.* приказује два синусна таласа различитих фаза.



Слика 4. Илустрација фазе [11]

Црвена линија приказује график функције $sinx$. Плава линија има потпуно исти облик, али мало померен. То је график функције $sin(x-\theta)$, где је θ фаза таласа.

На *Слици 5.* су приказана три различита графика. Када је свемирска летелица спремна да пошаље информацију на Земљу почиње са емисијом једноставних таласа. Они су нешто касније примљени на Земљи при чему пријемник детектује фреквенцију и фазу примљеног таласа. Претпоставимо да је фаза нула. Тачка P је карактеристична тачка таласа. Када је успостављена веза између одашиљача летелице и пријемника на Земљи, одашиљач почиње да мења талас тиме што мења његову фазу. Тачка P' на таласу чија је фаза померена за $90°$ који може да представља цифру 0. Тачка P'' се налази на таласу чија је фаза померена за $-90°$ који може да представља цифру 1. Тиме се модулацијом фазе таласа шаље низ нула и јединица на Земљу. Цео овај поступак захтева веома мало енергије.



Слика 5. Померање фазе таласа [11]

3.1.Откривање и исправљање грешака

Уколико је сигнал слаб и раздаљина велика, постоји велика вероватноћа да дође до грешака приликом преноса информација. Оне могу настати приликом креирања сигнала, слања сигнала или пријема сигнала. Најједноставнија форма откривања и исправљања грешака подразумева слање сваке бинарне цифре три пута. Уколико претпоставимо да је веома мали део информације која треба да се пошаље 1011 тада ће рачунар који преноси информације трансформисати овај низ бинарних цифара у 111 000 111 111. *Слика 6.* приказује низ таласа који ће бити послати на Земљу. Свака цифра 1 у овом систему се шаље као $\sin(x + \pi/2)$, а свака цифра 0 као $\sin(x - \pi/2)$. Гледајући у фазу таласа научници закључују да су примили поруку која гласи 111 000 111 111. Знајући да постоје сувишне цифре, може се закључити да је оригинална порука гласила 1011.

Слика 6. Слање информације 111 000 111 111 [11]

Ако се претпостави да је уместо низа таласа приказаних на *Слици 6.* пристигао мало другачији низ, приказан на *Слици 7.* то би сигнализирало да је дошло до грешке.

Слика 7. Грешка 111 000 111 011 [11]

У неком тренутку, један од таласа је променио своју фазу, за шта постоји много узрока. Резултат је примљена порука која не представља триплете цифара. Уместо тога, пристигли низ цифара изгледа 111 000 111 011. Чињеница је да свака цифра није три пута записана, моментално алармира пријемник да је дошло до грешке. Пријемник чак зна где је грешка: 011. Чињеница да је цифра 1 стигла два пута, а цифра 0 једном, снажно имплицира да је цифра 0 погрешна цифра, односно да је пријемник требало да прими поруку која изгледа 111 000 111 111, што значи да је оригинални низ био 1011.

ЗАКЉУЧАК

У раду су били изложени начини којима се настава математике може обогатити и улепшати. Дате су разне чињенице за поткрепљивање реченице „Математика има огромне примене“ и начини за њихово систематично излагање. Примери из свакодневног живота служили су да посведоче истинитост тврдње и покажу природан и лак начин за увођење математичких садржаја који можда нису свима интуитивни.

Такође, овај рад би могао да послужи као идеја за реализацију интердисциплинарних часова. Уска повезаност обрађених математичких садржаја и продуката њене примене у рачунарству упућује на природну повезаност математике и рачунарства и информатике као наставних предмета.

Наведени садржаји би представљали значајно освежење за наставу математике, ученицима би се вероватно допали и кроз њих би на пријемчив начин усвојили битна математичка знања.

Литература

[1] Ана Драгановић, Мастер рад Криптоанализа Вижнерове шифре, Математички факултет Београд 2009.

[2] D. Johnson, T. Mowry, Mathematics a Practical Odyssey, Brooks/Cole, Cengage Learning, Belmont, 2012.

[3] Н. Икодиновић, Математика, уџбеник са збирком задатака за први разред гимназија и средњих стручних школа, Клет, Београд 2014.

[4] Светозар Курепа, Увод у математику, Техничка књига, Загреб, 1984.

[5] http://www2.ohlone.edu/people2/bbradshaw/math155/miscellaneous/m155_book.pdf Приступљено 28.4.2015.

[6] A. Sultan, A.F. Artzt, The Mathematics that every secondary school math teacher needs to know, Routledge, Taylor & Francis, New York, 2011.

[7] http://www.gs1yu.org/ Приступљено 10.6.2015.

[8] З. Каделбург, В. Мићић, С. Огњановић, Анализа са алгебром 2, уџбеник са збирком задатака за 2. разред Математичке гимназије

[9] Jean Gallier, Discrete Mathematics, Springer, New York, 2011.

[10] http://doctrina.org/How-RSA-Works-With-Examples.html Приступљено 28.4.2015.

[11] J. Maasz, J. O' Donoghue, Real-World Problems for Secondary School Mathematics Students, Sense Publishers, Rotterdam, 2011.

[12] Ф.Марић: Информатика, уџбеник за први разред гимназије, Klett, Београд, 2014.

[13] A. Young, Mathematical Chipers: From Caesar to RSA, Mathematical World, American Mathematical Society 2006.