# 6- Mavzu: Hisoblash algoritmlarida xatoliklarni inobatga olish usullari

## 1. Satrlarni taqqoslash

Satrlarni taqqoslash algoritmining mohiyati matndan berilgan satrga mos tushuvchi qism satrni topishdan iborat. Standart algoritm ishni berilgan satr birinchi simvolini matnning birinchi simvoli bilan taqqoslashdan boshlaydi. Agar ular mos tushsa, matnning va namuna satrning ikkinchi simvoliga o'tiladi. Ushbu jarayon qism satrning namuna satr bilan butunligicha mos tushgunga qadar yoki mos tushmaydigan simvollar uchragunga qadar davom etadi. Birinchi holda masala hal bўladi, ikkinchi holatda esa matndagi joriy holat ko'rsatlichini bir imvolga surib, ishni boshdan boshlaymiz. Ushbu algoritmning matni quyida keltirilgan:

```
subLoc=1// Namuna satrdagi taqqoslanuvchi joriy simvol ko'rsatkichi
textLoc=1//Matndagi taqqoslanuvchi joriy simvol ko'rsatkichi
 textStart=1//Matndagi taqqoslanish boshi ko'rsatkichi
while textLoc<=length(text) and subLoc<=length(substring) do
if text [textLoc]=substring[subLoc] then
textLoc=textLoc+1
subLoc= subLoc+1
else
//keyingi simvoldan boshlab boshdan taqqoslash
textStart= textStart+1
textLoc=textLoc+1
subLoc= subLoc+1
end if
end while
if (subLoc>length(substring)) then
return textStart // Mos tushish yuz berdi
else
return 0 // Mos tushish yuz bermadi
enf if
```

Algoritmdan ko'rinadiki, bunda asosiy amallar taqqoslashlar bo'lib, ularning umumiy sonini hisoblab borish talab etiladi. Algoritm ishining eng yomon holatida har bir o'tishda oxirgisidan boshqa barcha simvollar mos tushishi kuzatiladi. Bu holat har bir simvol uchun bir martadan ro'y beradi. Agar namuna satr uzunligi S ga, matn uzunligi T ga teng bo'lsa, eng yomon holatdataqqoslashlar umumiy soni S ga teng bo'ladi.

Standart algoritmning asosiy muammosi uning ko'p miqdorda befoyda mehnat sarf qilishida namoyon bo'ladi.

## 2. Chekli avtomatlar

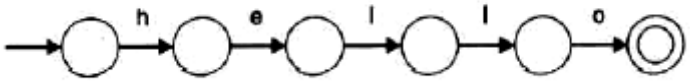
Chekli avtomatlardan berilgan so'zning berilgan alfavitga(tilga) tegishli yoki tegishli emas ekanligini aniqlashda foydalaniladi. Chekli avtomat joriy holati va o'tish funktsiyalarini ifoda etuvchi sodda tuzilmadan iboratdir. Bunda o'tish funktsiyasi navbatdagi kirish simvolining joriy holati va qiymati bo'yicha

avtomatning yangi holatini shakllantiradi. Agar kiritish oxirida avtomat qabul qilish holatida bo'lsa , avtomat holatlari qabul qiluvchi deb hisoblanadi.

Namuna so'zga moslashtirilgan chekli avtomatdan namuna bilan qiyoslash algoritida foydalanish mumkin. Agar avtomat qabul qiluvchi holatga o'tsa, namuna satr matnda topilgan deb hisoblash mumkin. Chekli avtomatlarda har bir simvol faqat bir marta qayta ishlanganligi uchun, ulardan effektiv foydalanish mumkin. Chekli avtomatlar yordamida namuna bilan qiyoslashda T dan katta bo'lmagan sondagi taqqoslash amallari bajariladi.

## 3.Knut-Morris-Pratt algoritmi.

Matnda namuna satrni izlovchi chekli avtomat qurishda boshlang'ich holatdan tugallovchi holatga o'tishlar namuna satrga kiruvchi simvollar bilan belgilab olinadi. Asosiy muammo namuna satrga tugallovchi holatga olib kelmaydigan simvollarni qo'shish jarayonida vujudga keladi.

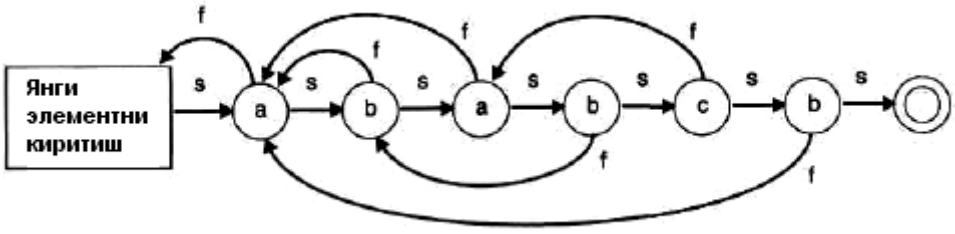



Knut-Morris-Pratt algoritmi chekli avtomat printsipiga asoslanadi, ammo unda mos tushmaydigan simvollarni qayta ishlashning boshqaa usulidan foydalaniladi. Ushbu algoritmda chekli avtomat holatlari ayni paytda mos tushishi kerak bo'lgan simvollar orqali belgilab olinadi. Har bir holatda ikki yo'nalishda o'tish imkoniyati mavjud:

**1- mos tushish ro'y bergan holat**

**2- mos tushish ro'y bermagan holat**

Mos tushish ro'y berganda avtomatning keyingi tugunga o'tishi yuzberadi, aks holda joriy tugundan oldingi (orqaga) tugunga o'tish yuz beradi. Quyidagi tasvirda ababcb namuna satri uchun tuzilgan Knut-Morris-Pratt avtomatining sxematik tuzilishi ifoda etilgan:




Har bir muvaffaqiyatli o'tish bajarilganda Knut-Morris-Prattchekli avtomatida matndan yangi simvol tanlanadi. Muvaffaqiyatsiz o'tishlarda yangi simvol tanlanmasdan, buning o'rniga oxirgi marta tanlangan simvol takroran qayta ishlanadi. Agar avtomat tugallovchi holatga o'tsa, matndan namuna satr topildi deb, hisoblanadi. Quyida ushbu algoritm matnini keltiramiz:

```
subLoc=1// Namuna satrdagi taqqoslanuvchi joriy simvol ko'rsatkichi
textLoc=1//Matndagi taqqoslanuvchi joriy simvol ko'rsatkichi
 while textLoc<=length(text) and subLoc<=length(substring) do
if subLoc=0 or text [textLoc]=substring[subLoc] then
textLoc=textLoc+1
subLoc= subLoc+1
else // mos tushmaslik yuz berdi; mos tushmaslik bo'yicha o'tish
subLoc=fail[subLoc]
end while
if (subLoc>length(substring) then
return textLoc-length(substring)+1// topilgan mos tushish
else return 0// izlangan namuna topilmadi
end if
```

## 4. Boyer-Mur algoritmi

Yuqorida tavsifiga to'xtalib o'tilgan algoritmlarda farqli ravishda Boyer-Mur algoritmi o'ngdan-chapga qarab teskari yo'nalishda amalga oshiradi. Bunda namuna satrni izlash jarayonida yanada unumdor "sakrab o'tish" usullaridan foydalaniladi. Quyidagi misolda y r bilan solishtirilganda mos tushmaslik yuz beradi. r xarfi namuna satriga kirmaganligi uchun, matnda to'rt simvolga o'ngga surish bajariladi. So'ngra y h bilan taqqoslanadi va yana mos tushmaslik yuz beradi. Ammo h namuna satrga kirganligi uchun matnda ikki simvolga o'ngga surishni bajarish mumkin. Bunda h simvollar mos tushfdi. So'ngra o'ngdan taqqoslashlar bajarilsa, matnning namuna satr bilan mos tushishi yuz beradi. Boyer-Mur algoritmida standart algoritmdagi 13 ta taqqoslash amali o'rniga 6 ta taqqoslash amali bajariladi.

Quyida algoritm matnini keltiramiz:

```
textLoc=length(pattern)
patternLoc= length(pattern)
while (textLoc<=length(text)) and (patternLoc >0) do
if text [textLoc]= pattern [patternLoc] then
textLoc=textLoc-1
patternLoc= patternLoc-1 else
textLoc=textLoc+MAX(slide[textLoc]], jump[patternLoc])
patternLoc= length(pattern)
end if
end while
if patternLoc=0 then
return textLoc+1// мос тушиш юз берди
else return 0
end if
```

Boyer-Mur algoritmi namuna satrni ikki usulda qayta ishlashi mumkin. Birinchidan, simvollarning navbatdagi mos kelmasligi yuz berganda mumkin bo'lgan surilish

uzunligi hisoblash bajariladi. Ikkinchidan, siljish uzunligini namuna oxiridagi ayni paytgacha uchragan simvollar ketma-ketligini ajratib hisoblash bajariladi.

## 5.Graflar nazariyasining asosiy tushunchalari

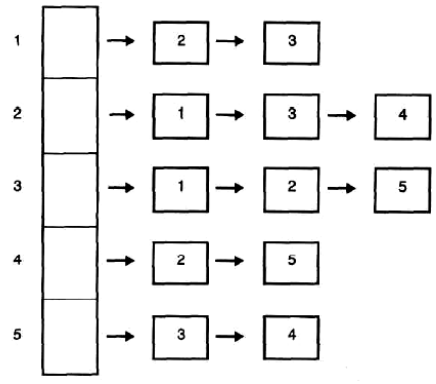
Formal jihatdan graf G=(V,E) tartiblangan to'plamlar juftligidan tashkil topib, bulardan birinchisi () tugunlar yoki uchlar, ikkinchisi (E) tomonlar yoki yo'nalishlar to'plamlaridan iboratdir. Tomon grafning ikki tugunini bir-biriga bog'laydi. Graf orientirlangan(yo'naltirilgan) yoki aksinchi bo'lishi mumkin. Orientirlanmagan(yo'naltirilmagan) grafda bir-biriga bog'langan tugundan ikkinchisiga har ikkala yo'nalishlarda xarakat qilish ruxsat etiladi. Quyidagi tasvirda yo'naltirilgan va yo'naltirilmagan graflar ularning formal ifodasi bilan bbirga berilgan.

Graflar to'g'risidagi ma'lumotlar ikki usulda saqlanishi mumkin:

**Birlashmalar matritsalari**

**Birlashmalar ro'yxatlari**

Birlashmalar matritsasi graf tomonlari(yo'nalishlari) to'g'risidagi ma'lumotga tez murojaat qilish imkoniyatini beradi. Ammo grafda tomonlar soni kichik bo'lsa, ushbu matritsa to'ldirilgan elementlardan ko'ra, ko'proq bo'sh elementlarga ega bo'ladi. Birlashmalar ro'yxatining uzunligi graf tomonlari soniga teng bo'lgani holda, tomon tog'risidagi ma'lumotga murojaat vaqti uzayadi. Agar grafda tugunlar soni katta bo'lib, ularni bog'lovchi tomonlar soni kichik bo'lsa, ushbu garf to'?risidagi ma'lumotni birlashmalar ro'yxati ko'rinishida saqlao' qulaydir. Aksincha, grafning tugunlari soni kichik bo'lib, ularni birlashtiruvchi tomonlar soni katta bo'lganda, garfni birlashmalar matritsasi ko'rinishida saqlash maqsadga muvofiq bo'ladi. Quyidagi tasvirlarda yo'naltirilgan va yo'naltirilmagan G graflarning birlashmalar matritsasi (a rasm) va birlashmalar ro'yxatlari( b rasm) ko'rinishidagi ifodalari keltirilgan:




## 6.Deykstra-Prim algoritmi

**Deykstra-Prim algoritmi**. 1950-yillarning oxirlarida Deykstra va Prim bir-birlaridan mustaqil ravishda grafning minimal qoldiqni izlash algoritmini taklif

etdilar. Bog’liqli vaznli garfning VMY (vazni minimal yig’indisi) i deganda uning barcha tugunlari va ularni bog’lab turuvchi ba'zi tomonlari dan iborat bo’lgan qism grafni tushunish mumkin. Algoritm ishida “ochko’z” algoritm printsipidan foydalaniladi. “Ochko’z” algoritm vaqtning har bir momentida berilgan ma'lumotlarning bir qismidan foydalanib, ular asosida eng yaxshi echimni topishga xarakat qiladi. Graf bilan ishlaganda har bir qadamda VMY (vazni minimal yig’indisi) ning qurilgan qismiga birlashgan tomonlar(yoylari) to’plami ko’rib chiqiladi va ular ichidan minimal og’irlikka ega bo’lgani tanlab olinadi.

## 7.Graf tugunlari

VMY (vazni minimal yig’indisi) ning qurilgan qismiga kirgan tugunlar, qurilgan qismning chekka (eng yaqin)tu tunlari va qolgan tugunlar. Grafning ixtiyoriy tugunini tanlab, uni VMY (vazni minimal yig’indisi) ga kiritaylik. Ushbu tugun bilan bog’langan barcha tugunlarni chekka tugunlar guruxiga kiritamiz. So’ngra VMY (vazni minimal yig’indisi) ni chekka tugunlar bilan birlashtiruvchi tomonlar ichidan eng kam vaznga ega bo’lgani qidiriladi. Bu tomon yangi tugun bilan birgalikda VMY (vazni minimal yig’indisi) kiritiladi. VMY (vazni minimal yig’indisi) ga ko’rib chiqilayotgan grafning barcha tugunlari kiritilgandan so’ng algoritm ishi tugallanadi.

## 8.Kruskal algoritmi

**Kruskal algoritmi**. Deykstra-Prim algoritmi VMY (vazni minimal yig’indisi) ni qurishni boshlang’ich grafning ixtiyoriy tugunidan boshlaydi va daraxtning qurilgan qismini tobora kengaytirib boradi. Ushbu algoritmdan farqli ravishda **Kruskal algoritmi** asosiy e'tiborni graf tomonlariga qaratadi. Bunda ishni bo’sh grafdan boshlab, unga tomonlarini ular vaznining o’sib borish tartibida ketma-ket qo’shib boradi. Bu jarayon grafga kiruvchi barcha tugunlar o’zaro bog’langunga qadar davom etadi. Agar tomonlarni qo’shib olish jarayoni barcha tugunlar o’zaro bog’langunga qadar tugatilsa, boshlan?ich grafning to’liq bog’lanmagan ekanligi kelib chiqadi.

## 9.Deykstra algoritmi

Grafning VMY (vazni minimal yig’indisi) ini anqlashda ishlatiluvchi “ochko’z” algoritm tugunlar orasidagi eng qisqa yo’lni aniqlashga yaramaydi, chunki har bir vadvmdv u faqat bitta tomon uzunligini hisobga oladi. Agar ushbu algoritmni har qadamda boshlang’ich tugundan chegara tugungacha bo’lgan eng qisqa yo’lning qismini tashkil qiluvchi tomonni tanlaydigan qilib o’zgartirsak, kerakli natijaga erishish mumkin bo’ladi.

## 10.Seriyalarni ketma-ket kodlash algoritmi

Axborotlarni siqish muammosi xozirgi kunda juda dolzarb bo'lib hisoblanadi.Buning boisi shundaki. Axborotlar oqimining tezlik bilan ortib borishi , ularni qayta ishlash, saqlash va uzatish tezligini ni oshirish muammosini keltirib chiqaradi. Bundan tashqari axborot texnologiyalarining turli hayotiy jabhalarda foydalanilish visitalarining rivojlanib borishi ushbu vositalar ishini ta'minlaydigan dasturiy mahsulotlar xajmining ortib borish tendensiyasini belgilaydi. Bunday sharoitda axborotlarni saqlash, xotira xajmini tejash muammolarining eng optimal echimi axborotlarni siqish usullari va ularga mos dasturiy vositalardan foydalanishdir.

EHM xotirasidagi axborotlarni siqilgan holda saqlash yuqoridagi muammolarni hal etishning birusulidir. Axborotlarni siqish(arxivlash) maxsus dasturlar yordamida amalga oshiriladi.. Ushbu dasturlar esa konkret siqish algoritmlari bo'yicha ishlaydi. Bugo'ngi mavzu ana shunday bir necha siqish algoritmlarining mohiyati bilan tanishishga qaratilgan.

## 11.Siqish jarayonlari asosiy texnik harakteristikalari

1. **Siqilish darajasi yoki boshlang'ich va natijaviy axborot oqimlarining nisbati;**
2. **Siqilish tezligi;**
3. **Siqilish sifati;**

Siqish yoki arxivlvsh jarayonining maqsadi – boshlashg'ich(kiruvchi) axborot oqimini ma'lum usullar bilan kompakt chiquvchi(natijaviy) axborotlar bilan almashtirishdir.

## 12.Siqilish usullari kategoriyalari

## Tiklanmas siqilish
## Tiklanuvchi siqilish

**Tiklanmas siqilishda** berilganlarning boshlang'ich oqimi qayta ishlanishi natijasida tashqi harakteristikalar bo'yicha boshlang'ich oqimga judu o'xshash, ammo informasion strukturasi bo'yicha yo'qotishlarga ega bo'lgan natijaviy oqim chiqariladi. Axborotlarni arxivlvshning bunday usuli rastrli grafik fayllar, video va foto axborotlar, nutq va boshqa analogli signallrni raqamli ko'rinishda ifodalashda qo'llaniladi.

**Tiklanuvchi siqilish** deganda axborotlar xajmini informasion strukturani yo'qotishlarsiz qisqartirish tushuniladi.Ushbu siqilgan axborotlarni faqat dekompressiya(tiklash) qilingandan keyingina qayta ishlash mumkin. Dekompressiya natijasida axborotlar oldingi xajmlarni egallaydi.

## 13.Tiklanuvchi algoritmlarning asosiy nazariy prinsiplariga

Endi tiklanuvchi algoritmlarning asosiy nazariy prinsiplariga to’xtalib o’tamiz. Eng sodda va mashxur arxivlash usuli – **seriyalarni ketma-ket kodlash (RLE) algoritmidir.** Algoritm mohiyati takrorlanuvchi baytlarketma-ketliklari yoki zanjirlarini bitta kodlovchi bayt va ularning takrorlashlar soni hisobchisiga almashtirishdan iborat. Masalan,

44 44 44 11 11 11 11 11 01 33 AA 22 22 berilgan ketma-ketlik bo’lsin. Uni RLE algoritmi yordamida siqish natijasida quyidagi ketmaqketlikka ega bo’lamiz:

03 44 05 11 00 03 01 33 AA 02 22

Birinchi bayt takrorlanuvchi baytlar sonini, ikkinchi bayt takrorlanuvchi baytning o’zini bildiradi.00 – baytidan keyin takrorlanmaydigan baytlar soni va takrorlanmaydigan baytlarning o’zi keladi.

Ushbu metod juda ko’p takrorlanuvchi baytlarga ega rastrli grafik tasvir fayllari uchun juda effektiv bo’lib hisoblanadi. RLE usulining kamchiligi qisish darajasining pastligidadir.

## 14.Xaffman algoritmi

**Xaffman algoritmi** – yana bir arxiflash usuli bo’li hisoblanadi.Axborotlarni Xaffman bo’yicha siqishda fayl butunligicha o’qiladi va undagi uchraydigan har bitta simvol uchun umumiy yig’indi miqdorlar hisoblanadi.Bunda 256ta simvolning hammasi hisobga olingan bo’lib, algoritm uchun bajariluvchi fayl bilan matnli fayl orasida hech qanday farq bo’lmaydi.hisoblash jarayoni natijalaridan foydalanib, binar daraxt tuziladi.

## 15.Lempel-Ziv algoritmi

**Lempel-Ziv algoritmi.** Ushbu algoritm ilk bor Abraham Lempel va YAkob Ziv ishlarida tasvirlab berilgan. Bugo’ngi kunda bu algoritm LZ - algoritmi deb yuritiladi.Ushbu algoritm modifikasiyalari boshqa algoritmlarga nisbatan ancha keng tarqalgan. LZ algoritmi asosida fayllarda ko’p uchraydigan ketma-ketliklarni maxsus yaratiluvchi lug’atda saqlanuvchi namunalarga murojaatlar bilan almashtirish yotadi.

Masalan, agar arxivlash dasturi lug’atda “ABV” ketma-ketlikka ega bo’lsa va “ABVA” ketma-ketlikka duch kelsa, chiqish fayliga “ABV” uchun lug’atdan kod yoziladi, keyin A uchun kod yoziladi, so’ngra “ABVA” ketma-ketlik lug’atga kiritiladi.Agar keyinroq “ABVAB” ketma-ketlik uchrasa, uning o’rniga “ABVA” uchun kod yoziladi, keyin “B” simvol uchun kod yoziladi hamda “ABVAB” yana lug’atga kiritiladi. Dastur lug’atda mavjud ketma-ketliklarni uchratsa, bu ketma-ketlik uchun kodni beradi va lug’atga bir bayt uzunroq yangi yozuvni kiritadi. Ushbu lug’atning xajmi turli dasturlarda turlichabo’lishi mumkin. Masalan, Lharc dasturi 4 Kbaytli buferdan, LHA va PKZIP 8 Kilobaytli, ARJ dasturi esa 16 Kbaytli buferdan foydalanadi.

Berilganlarni lug’atdagi qism-satrlar bilan almashtirish jarayoni quyidagicha amalga oshiriladi: berilgan qism-satr bilan mos tushuvchi lug’atdagi qism-satrlardan eng uzuni topiladi va chiquvchi oqimga 2 ta satr uzatadi (lenght, distanse): lenght – lug’atda topilgan qism-satr uzunligi, distanse- lug’atdagi qism-satrdan kirish qism-satrigacha bo’lgan masofa. Agar bunday qism satr topilmasa. CHiqish oqimiga kirish oqimining navbatdagi simvoli qo’shiladi.

Shunday qilib, Lempel-Ziv algoritmi boshlang’ich berilganlarning simvollar ketma-ketligini 2 ta parallel uzunliklar va masofalar jadvaliga aylantirib beradi.Ushbu jadvalga yuqorida to’xtalib o’tilgan RLE yoki Xaffman algoritmlaridan birini qo’llash mumkin bo’ladi. SHu tarzda 2 bosqichli kodlash amalga oshiriladi . Ushbu metodni realizasiyasida ikkala oqimning bitta faylga chiqarilishiga erishish kerak.Bu muammo ikkala oqim simvollarini oralatib yozish yo’li bilan hal etiladi.

## 16.Xotira bo’yicha murakkablik

Biz asosan algoritmlarning vaqt bo'yicha murakkabligini muhokama qilamiz, ammo ish bajarish uchun u yoki bu algoritmga qancha xotira kerakligi haqida ham aytish mumkin. Kompyuter xotirasi (ham ichki, ham tashqi) hajmi chegaralangan. Kompyuterlar rivojlanishining dastlabki bosqichlarida bu tahlil uslubiy xarakterga ega edi. Barcha algoritmlar chegaralangan xotira yetarli yoki qo’shimcha maydonni talab qiluvchi algorilmlarga bo’linadi. Ko’pincha dasturlovchilar xotirasiga ega va tashqi qurilmalar talab qilmaydigan sekin ishlovchi algoritmni tanlashar edi. Kompyuter xotirasiga bo’lgan talab juda katta edi, shuning uchun qaysi ma’lumotlar saqlanib qoladi, bunday saqlashning samarali usullari qanday kabi savollar o’rganilar edi. Faraz qilaylik, masalan, biz -10 dan +10 gacha intervaldagi verguldan keyin bitta o’nli belgiga ega bo’lgan moddiy son yozyapmiz. Moddiy sonni yozishda ko'pchilik kompyuterlar 4 dan 8 baytgacha xotira sarflaydi, lekin agar bu sonni awaldan 10 ga ko’paytirsak, -100 dan + 100 gacha intervaldagi butun son hosil qilamiz va uni saqlash uchun bor yo'g’i bir bayt sarflanadi.

## 17.Kiruvchi ma’lumotlarning sinflari

Algoritmni tahlil qilishda kiruvchi maTumotlarni tanlash uning bajarilishiga ta’sir qilishi mumkin. Aytaylik, ba’zi saralash algoritmlari, agar kirish ro'yxati saralangan bo’lsa, juda tez ishlashi mumkin, boshqa algoritmlar shunday ro'yxatda uncha kata bo'lmagan natijani ko’rsatadi. Tasodifiy ro’yxatda esa natija buning teskarisi bo'lishi mumkin. Shuning uchun biz ma’lumotlaming bir kirish ro’yxatidagi algoritmlar harakatini tahlil qilish bilan chegaralanmaymiz. Biz algoritmni ham eng tez, ham eng sekin ishlashini ta ’minlovchi ma’lumotlarni qidiramiz. Bundan tashqari, biz barcha mavjud ma’lumotlar to'plamidagi algoritmlarning o’rtacha samarasini ham baholaymiz.

## Foydalanilgan adabiyotlar


1. Informatika va informatsion texnologiyalar, M.Aripov va boshqalar. Oliy o‘quv yurti talabalari uchun darslik. Toshkent-2019 y.
2. Axborot texnologiyalari, M.Aripov va boshqalar. Oliy o‘quv yurti talabalari uchun o‘quv qo‘llanma. Toshkent-2019 y.
3. Delphi tilida dasturlash asoslari, Sh.Nazirov. Toshkent-2018 y.