O’ZBEKISTON RESPUBLIKASI OLIY VA O’RTA MAXSUS TA’LIM VAZIRLIGI

ALISHER NAVOIY NOMIDAGI SAMARQAND DAVLAT UNIVERSITETI

«Algoritmlar nazaryasi» fanidan

Kurs ishi

Mavzu:ALGORITMLAR SIFATINI BAHOLASHNING ASOSIY MEZONLARI

Talabasi:Asadov Farrux
Mutaxassislik:Amaliy matema-tika va informatika
Ilmiyrahbar: Qarshiyev H.

SAMARQAND-2013

ALGORITMLAR SIFATINI BAHOLASHNING ASOSIY MEZONLARI

Reja

Kirish

1- bob.Algoritm to’g’risida umumiy tushuncha

1.1. Algoritmning ta’rifi.

1.2. Masalaning qo’yilishi

1.3. Modelni yaratish.

1.5. Algoritmni ishlab chiqish.

2-bob. Algoritm sifatini baholashning mezonlari

2.1. Algoritm to’g’riligini tekshirish.

2.2. Algoritmni amalga oshirish.

2.3. Algoritmni va ularning murakkabligini tahlil qilish.

2.4. Dasturni tekshirish.

2.5. Hujjatlashtirish.

Xulosa

Foydalanilgan adabiyotlar

KIRISH

Hozirgi kunda biror bir sohada ishni boshlash va uni boshqarishni kompyutersiz tasavvur qilish qiyin. XXI asr savodxon kishisi bo'lishi uchun kompyuter savodxon bo'lish, axborot texnologiyalarini puxta egallamoq lozim. Har bir mutaxassis, u qaysi sohada ishlashdan qat'iy nazar, o'z vazifasini zamon talabi darajasida bajarishi uchun axborotni ishlab chiqaruvchi vositalar va ularni ishlatish uslubiyotini bilish va ishlash ko'nikmalarga ega bo'lishi zarur. Talabalarni ijtimoiy-iqtisodiy va ma'naviy muammolarni hal etishga safarbar qilmoq uchun tegishli axborotlarni o'z vaqtida to'plab, qayta ishlab, muayyan bir tartibga solish va zudlik bilan kishilarga etkazish kerak bo'ladi. Buning uchun jamiyatni axborotlashtirish dasturini amalga oshirish va ilg'or axborot texnologiyasini joriy etish zarurdir.

Dasturlarni mustaqil tuzishdan maqsad kompyuterga mutloq xokimlik qilish, ya'ni ish davomida yuzaga keladigan muammolarni tezroq hal etish imkonini yaratishdir. Kompyuter dasturlari sermehnat ishlarni avtomatlashtiradi, xatolarni kamaytiradi va mehnat unumdorligini oshiradi. Bundan tashqari, dasturlar tuzish juda ham mashg'ulotdir.

Dasturlarni yaratish jarayonida qo'yilgan masalaning yechish algoritmi dastlab to'g'ri ishlab chiqilishi muhim axamiyatga ega. Shuning uchun algoritmlarni tuzish va dasturlarni ishlab chiqish bir-biri bilan chambarchas bog'liq jarayonlardir. Oliy o'quv yurtlarining informatika, axborot texnologiyalari, amaliy matematika kabi yo'nalishlarida ta'lim olayotgan talabalar algoritmni ishlab chiqish, dasturlar yaratish, ularni sinash, sozlash, tahlil qilish uchun bilimlarni puxta o'zlashtirishlari zarur. Bunda, ta'lim oluvchi uchun dasturlarni ishlab chiqishda asosiy va eng muhim bosqich hisoblangan algoritmlarni tuzish va shular asosida dasturlar yaratish haqida ma'lumotlarni beruvchi adabiyotlar kerak.

Ma'ruzalar matni Oliy o'quv yurtlari talabalari uchun mo'ljallab yozilgan va zamonaviy kompyuter texnologiyalarini mustaqil ravishda o'rganayotgan barcha qiziquvchilar uchun ham foydalidir.

1- bob.Algoritm to’g’risida umumiy tushuncha

1.1. Algoritmning ta’rifi.

Algoritmlarning turli ta’riflari mavjud. Rasmiy ta’riflardan biri bo’yicha algoritm bu qo’yilgan masalani bir xil yechilishiga olib keluvchi aniq harakatlarning ketma-ketligi. Bu tushunchadan algoritmning quyidagi xossalari kelib chiqadi:

1. Diskretlilik – ya’ni aniqlanayotgan jarayonni qadamba-qadam ko’rinishi.
2. Ommaviylik – algoritm o’xshash masalalar turkumini yechishi kerak.
3. Tushunarlilik – algoritmda beriladigan ko’rsatmalar foydalanuvchiga tushunarli bo’lib, uning talablariga javob berishi kerak.
4. Aniqlilik – algoritmda ma’lum tartibda amallarni bajarish nazarda tutilishi kerak va bajaruvchiga joriy qadam tugatilishi bilan qaysi qadam keyingi bo’lib bajarilishi aniq ko’rsatilishi kerak.

Algoritmlar rasmiy ravishda bajariladi, bu degani bajaruvchi bajarilayotgan amallarni mazmunini anglash shart emas.Algoritm tuzish jarayoniga algoritmlashtirish deyiladi.

Algoritm tuzish jarayonida nazariy va amaliy nuqtai nazardan algoritmlash, dasturlash va EHM larni qo’llash bilan bog’liq bo’lgan bilimlar kerak. Asosiy maqsad bu masalani qo’yish, masalaning yechish algoritmini tuzish, algoritmi mashina dasturi ko’rinishida amalga oshirish va algoritmni samaradorligini ko’rsatish muammolarini o’rganish. Bu jarayonlar algoritmni to’liq yaratish tushunchasiga olib keladi va quyidagi bosqichlarni belgilaydi:

1. Masalaning qo’yilishi.
2. Modelni yaratish.
3. Algoritmni ishlab chiqish.
4. Algoritm to’g’riligini tekshirish.
5. Algoritmni amalga oshirish.
6. Algoritmni va ularning murakkabligini tahlil qilish.
7. Dasturni tekshirish.
8. Hujjatlashtirish.

Masala qo’yilishi

Masalani yechishdan oldin, uni berilishini aniq shakllantirib olish zarur. Bu jarayon to’g’ri savollarni aniqlash bo’lib, savollar quyidagicha bo’lishi mumkin:

1.1. Dastlabki berilgan masala shartlarida hamma iboralar tushunarlimi?

1.2. Nima berilgan?

1.3. Nimani topish kerak?

1.4. Yechimni qanday ta’riflash kerak?

1.5. Qaysi berilganlar yetarli emas va hammasi kerakmi?

1.6. Qanaqa mumkinliklar qabul qilingan?

Albatta, bulardan tashqari boshqa savollarni ham ishlatish mumkin, yoki ayrim savollarni bir necha bor takror ishlatishga to’g’ri keladi.

Modelni yaratish

Akademik A. N. Tixonov fikri bo’yicha matematik modellashtirish dunyoni bilish va o’rganishda kuchli qurollardan (vositalardan) biridir.

Uning ta’rifi bo’yicha matematik model tashqi dunyoning xodisalar turkumini matematik belgilar yordamida taxminiy tavsifi.

Xodisani tavsiflash uchun uning muhim xususiyatlarini, qonuniyliklarini, ichki aloqalarini, ayrim xossalarning ahamiyatini aniqlash zarur.Eng muhim faktorlari aniqlanganda, ahamiyatlari kamroq bo’lganlarini hisobdan chiqarish mumkin.Umuman, modelni tanlash fandan ko’ra, ko’proq san’at ishi deb hisoblanadi, yahshi tuzilgan modellarni o’rganish esa – modellashtirishda tajriba orttirishning eng yahshi usuli. Modelni yaratishda quyidagi savollarni aniqlash maqsadga muvofiq:

2.1. Masalani yechish uchun qaysi matematik struktura ko’proq mos keladi?

2.2. O’xshash masalaning yechimi bormi?

2.3. Masalaning barcha muhim ma’lumotlari matematik ob’yektlar orqali tavsiflanadimi?

2.4. Izlanayotgan natija biron bir matematik o’lchamga mos keladimi?

2.5. Modelning ob’yektlari orasidagi bog’lanishlar aniqlanganmi?

2.6. Tuzilgan model bilan ishlash qulaymi?

Algoritmni ishlab chiqish

Algoritmlarni yaratish ijobiy ish, shuning uchun ixtiyoriy zarur algoritmlarni tuzish imkonini beradigan bir umumiy usul mavjud emas. Lekin algoritmlarni ishlab chiqishni asoslangan oddiy sxemalarini beradigan ko’pgina algoritmlashtirish nazariyalari bor. Bunday sxemalar va yangi algoritmlarni paydo qilishning o’rtasida qattai bog’liqlik kuzatiladi. Tez uchraydigan va ko’p foydalaniladigan usullarni quyidagicha ajratib olish mumkin:

1. **Algoritmlarni konstruksiyalash.**Bu usulda yangi algoritm mavjud algoritmlardan tarkibiy qismlar sifatida foydalanib, bir-biriga moslab bir butunlik hosil qilish yo’li bilan ishlab chiqiladi.
2. **Algoritmlarni ekvivalent qayta ishlash.** Ikki algoritm ekvivalent hisoblanishi uchun quyidagi shartlar bajarilish kerak:
    - Bittasi uchun mumkin bo’lgan dastlabki berilganlar varianti, ikkinchisi uchun ham mumkin bo’lishi kerak.
    - Bir algoritmni qandaydir dastlabki ma’lumotga qo’llanilishi, ikkinchi algoritmni ham shu berilganga qo’llanilishiga kafolat beradi.
    - Bir xil dastlabki berilgan ma’lumot uchun ikkala algoritm ham bir xil natija berishi. Lekin bu algoritmni ikki xil shakllarini ekvivalent deb nomlash noto’g’ridir.

Shunday qilib, algoritmni ekvivalent qayta ishlash deb, natijada dastlabki algoritmga ekvivalent algoritmni paydo qiladigan o’zgartirilishlarga aytiladi.

Misol tariqasida, algoritmni bir tildan boshqa tilga o’tkazishni keltirish mumkin.Shu bilan birgalikda algoritmni ekvivalent qayta ishlash usuli bilan keskin o’zgartirish mumkin, lekin bu holda asosiy e’tiborni dastlabki algoritmga nisbatan yahshi algoritmni yaratishga berish kerak.

3, <u>Toraytiruvchi o’zgartirishlar.</u>Bunday o’zgartirishlar natijasida dastlabki algoritmlar yechish kerak bo’lgan masalalarning xususiy holati yechimi algoritmlari ishlab chiqiladi.Odatda, bu usulda ekvivalent qayta ishlash jarayonida algoritmni ixchamlashtirish maqsaddida foydalaniladi.

4. <u>Formal usulni matematikaga bog’liq bo’lmagan muammoga qo’llash.</u>Buyerda matematik muammo matematik ko’rinishga o’tkazilib, uning algoritmini ishlab chiqishga uriniladi.Agar o’xshash matematik masala yechimining algoritmi ma’lum bo’lsa, undan foydalaniladi.

Algoritmlashtirish jarayoni uslublari bo’yicha matematik modellarni tuzish jarayoniga juda yaqin. Har bir algoritmni ishlab chiqish bevosita o’ziga xos yondashishni talab qilishiga qaramasdan, bu faoliyatni umumiy uslub va bosqichlari ham mavjud. Ba’zan dasturlarni tezroq yozib boshlashga hohish paydo bo’ladi. Lekin bu xatoli, chunki aynan algoritmni ishlab chiqish bosqichiga va uning to’g’riligiga masalaning to’liq yechimi bog’liqdir.Algoritmlarni tuzish turli xil uslublari mavjud.

2-bob. Algoritm sifatini baholashning mezonlari

<u>2.1.Algoritmni to’g’riligini tekshirish</u>

Dastur to’g’riligini isbotlashning eng keng tarqalgan turi – bu uni testlardan o’tkazishdir.

Algoritmni tekshirishda nazoratchi boshlang’ich ma’lumotlarni majmui algoritmik test deb nomlanadi.

To’g’ri deb shunday algoritmga aytiladiki, u masalaning qo’yilishida talab qilinadigan natijani har qanday ruxsat etilgan boshlang’ich ma’lumotlar bilan ham shakllantirib biladi. Odatda, dastur bergan natijalar ma’lum bo’lgan yoki qo’lda hisoblangan ma’lumotlar bilan taqqoslanadi, va ular to’g’riligi aniqlansa dastur to’g’ri ishlaydi degan hulosaga kelish mumkin. Ammo bu usul bilan foydalanuvchini hamma shubhalardan xalos qilib bo’lmaydi, ya’ni dastur ishlamaydigan hamma holatlarni hisobga olib bo’lmaydi.

Gudman va Xidetniyemi [2] lar tomonidan algoritm to’g’riligini isbotlash uchun quyidagi uslubiyat taklif qilingan.

Algoritm 0 dan *m* gacha bo’lgan qadamlar ketma-ketligi ko’rinishida tavsiflangan deb tahmin qilaylik.Har bir qadam uchun qandaydir asoslanishni taklif etamiz. Xususan, qadamdan oldin va keyin ishlaydigan shartlar haqida lemma kerak bo’lishi mumkin. Shu bilan birgalikda, algoritm chekliligining isbotini ham taklif etamiz, va hamma ruxsat etilgan kiritish ma’lumotlarini tekshirib, hamma mumkin bo’lgan chiqarish ma’lumotlarni olamiz. Algoritmni to’g’riligi bilan samaradorligi o’rtasida hech qanday aloqa yo’qligini ta’kidlab o’tamiz.Aslida hamma talablarga bir xil yahshi javob beradigan algoritm kamdan-kam ishlab chiqiladi.

Algoritmni amalga oshirish

Algoritmni amalga oshirish deganda, EHM uchun dasturni yozish deb tushuniladi. Buning uchun quyidagi savollarga javob berish kerak:

5.1. Asosiy o’zgaruvchilarni aniqlash.
5.2. O’zgaruvchilarning turlarini aniqlash.
5.3. Nechta massiv yoki fayllar va qanday kattalikda ular kerak bo’ladi?
5.4. Bog’lanilgan ro’yhatlardan foydalanish ma’nolimi?
5.5. Qanday dasturiy qismlar kerak bo’lishi mumkin (tayyor bo’lsa ham)?
5.6. Qaysi dasturlash tilini tanlash?

Dastur yozish yoki tuzishning hilma-hil usillari va uslublari mavjud.

Algoritmni va uning murakkabligini tahlil qilish

Algoritmni tahlil qilishdan maqsad – algoritmga ma’lumotlarni aniq muvaffaqiyatli qayta ishlash uchun kerak bo’ladigan xotira hajmi va ishlash vaqtining baholari va chegaralarini olish. Bir masalani yechadigan ikki algoritmni taqqoslash uchun qandaydirsonli mezon topish kerak.

Faraz qilaylik, A – qandaydir bir turkumdagi masalalarni yechadigan algoritm, n – esa shu turkumdagi alohida bir masalaning kattaligi.Umumiy holda, n – oddiy skalyar yoki massiv yoki kiritiladigan ketma – ketlikning uzunligi bo’lishi

mumkin. $f_A(n)$ - n kattalikdagi ixtiyoriy masalani yechadigan algoritm A bajarish kerak bo'lgan asosiy amallarni (qo'shish, ayirish, taqqoslash,…) yuqori chegarasini beradigan ishchi funksiya. Algoritmningsifatini baholash uchun quyidagi mezonni ishlatamiz.

Agar $f_A(n)$ o'sish tartibi *n* dan bog'liq bo'lgan polinomdan katta bo'lmasa, A algoritm polinomial deb aytiladi, aks holda algoritm A eksponensial hisoblanadi.

Shular bilan birgalikda tahlil jarayonida ko'p matematik fanlarda standart bo'lgan iboralar ishlatiladi.

$f_A(n)$ funksiya O[g(n)] deb belgilanadi, va $\lim_{n\to\infty} \frac{f(n)}{g(n)} = const \neq 0$ bo'lganda, uni tartibi katta n lar uchun *g(n)* deb qabul qilinadi. Demak *f(n)=O[g(n)]*.

$f_A(n)$ funksiyasi o[z(n)] deb katta n lar uchun belgilanadi, va unda $\lim_{n\to\infty} \frac{h(n)}{z(n)} = 0$ sharti bajariladi.

Bu begilar "katta O" va "kichik o" deb nomlanadi. Agar *f(n)=O[g(n)]* bo'lsa, ikkala funksiya ham $n \to \infty$ bo'lganda bir xil tezlikda o'sadi.

Agar *f(n)=O[g(n)]* bo'lsa,unda g(n), f(n) nisbatan ancha tez o'sadi.

Demak, $P_k(n)$- qandaydir n o'zgaruvchidan bog'liq va k darajadagi polinom uchun $f_A(n) = O[P_k(n)]$ yoki $f_A(n) = oP_k(n)$ bo'lganda algoritm polynomial hisoblanadi, aks holda algoritm eksponensial.

Eksponensial algoritm yahshi ishlamaydigan deb hisoblanadi.Agar algoritmlar eksponensial bo'lsa, ular orasida eng samaralisini topish kerak, n kattalikdagi masalani $O(2^n)$ qadamda yechadigan algoritm $O(n!)$ yoki $O(n^n)$ qadamda masalani yechadigan algoritmdan afzalroq.

Dasturni tekshirish

Biz dasturni har bir qismini tekshiradigan kirituvchi ma’lumotlar to’plamini tanlashimiz kerak.Ko’p murakkab algoritmlarni matematik tomondan tadqiq qilish yoki juda qiyin yoki mumkin emas. Bunday holatlarda algoritmni faoliyat jarayonida va qiyinligi bo’yicha tekshiradi. Bundan tashqari dasturlarni hisoblash imkoniyatlarini aniqlash uchun ham testlash maqsadga muvofiq.Ko’p dasturlar qandaydir kiritiladigan ma’lumotlar bilan yahshi ishlasa, boshqalari bilan yomon ishlaydi.“Yahshi” lardan “yomon” larga o’tish “mayin” bo’lish kerak.Testlash uchun ma’lumotlar dasturning qiyinligiga, mavjud vaqt resurslariga, kiritish-chiqarishsoniga bog’liq holda tanlanadi. Bu yerda analitik va eksperimental tahlil bir-birini to’ldiradi.

Hujjatlashtirish

O’zingiz yozmagan dastur kodini o’qish juda qiyin.Bu muammoni hujjatlashtirish yordamida yechsa bo’ladi. Hujjatlashtirish o’z ichiga hamma yordamchi ma’lumotlarni oladi va dasturda nima bajarilishini tushuntirib beradi, xususan, blok-sxemalardagi boshqarishni uzatish, berilganlarni kiritish-chiqarish shaklini batafsil tavsif qilish, siklning parametrlari, yordamchi local va global proseduralarni bajarilishi va boshqalar.

Hujjatlashtirishning eng asosiy qoidasi bu “boshqalar yozgan dasturlarni qanday ko’rishni istasangiz, o’zingiz ham dasturni shunday ko’rinishda rasmiylashtiring”.

Masalalar yechish.

1 - misol. *Berilgan to’rt xonali butun sonning raqamlari ko’paytmasini toping.*

Test

Test tartibi	Tekshirish	Son	Natija
1	Musbat son	2314	P = 24

2	Manfiy son	-1245	P = 40

Algoritmi:

alg Butun_son (**but** Num, P)
arg Num
natija P
boshlbutun i, j, k, l
  Num := abs(Num)
  i := Num div 1000
j := ((Num div 100) mod 10)
k := ((Num div 10) mod 10)
  l := Num mod 10
  P := i * j * k * l;
Tamom

Turbo Pascaldagi dasturi:

```
Program Farrux;
Var Number, i, j, k, l, P : Integer;
BEGIN
  ReadLn(Number);    Number:=Abs(Number);
    i := Number div 1000; Write(i:3);
  j := Number div 100 mod 10; Write(j:3);
  k := Number div 10 mod 10; Write(k:3);
  l := Number mod 10; WriteLn(l:3);
  P := i * j * k * l ;    WriteLn( P);
   ReadLn
END.
```

2 - misol. Butun qiymatli A(N, M) matritsa berilgan. Agar matritsa satrining hech bo’lmaganda biror elementi manfiy bo’lsa, u holda bu satrning barcha elementlarini
nollar bilan almashtiring

Test

Berilgan		Natija
N	A matritsa	A matritsa

3	$\begin{pmatrix} 1 & -2 & 1 \\ 1 & 2 & 1 \\ -1 & 2 & -2 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{pmatrix}$

Algoritmi

alg Modifikasiya(**but** N, **haq jad** A[1:N, 1:N])
boshl but i, j, **lit** Flag
kiritish N
sb i**uchun** 1 **dan** N **gacha**
sbj **uchun** 1 **dan** N **gacha**
kiritishA[i,j]
so
so
sbi **uchun** 1 **dan** N **gacha**
j := 1; Flag := "Yuk"
sb toki (j<=N) va (Flag = "Yo′q")
agar A[i, j]<0 u holda Flag := "Ha"
aks holda j:=j+1
h**al bo′ldi**
so
agar Flag = "Ha"u holda
sbj **uchun** 1 **dan** N **gacha** A[i, j]:=0
so
hal bo′ldi
so
sbi **uchun** 1 **dan** N **gacha**
sbj **uchun** 1 **dan** N **gacha**
chiqarishA[i,j]
so
so
tamom.

Algoritmning bajarilishi

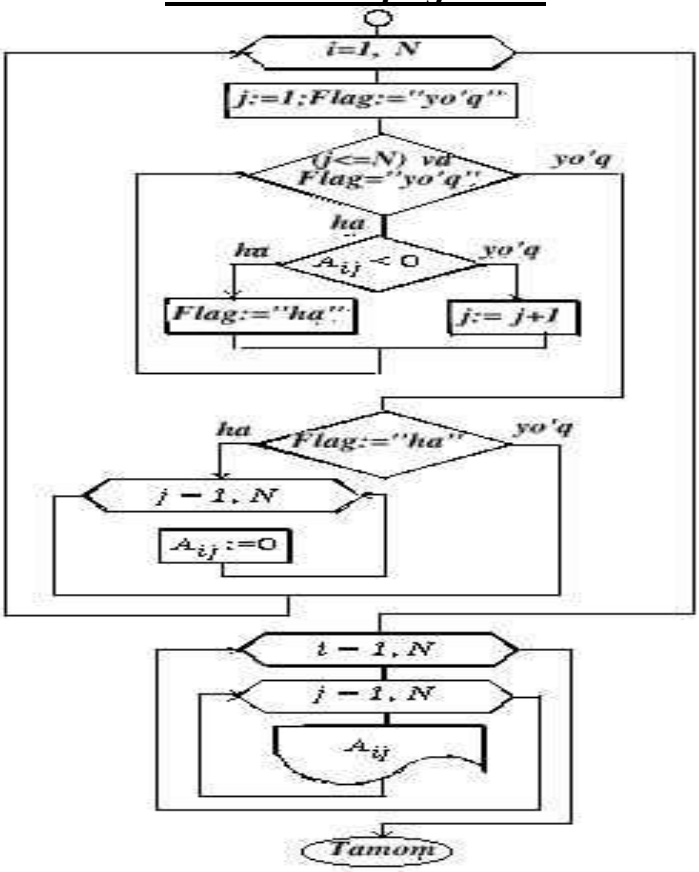
Tekshirilayotgan shartning belgilanishi:

(j<=N) va (Flag = "Yo'q")=> (1)

i	Flag	j	(1)	A[i,j]<0	Flag="Ha"	A[i,j]
1	"Yo'q"	1	+	-	+	
	"Ha"	2	+	+		A[1,1]=0
						A[1,2]=0

		1 2 3	-(so)			A[1,3]=0
2	"Yo'q"	1 2 3 4	+ + + -(so)	- - -	-	
3	"Yo'q" "Ha"	1 1 2 3	+ -(so)	+	+	A[3,1]=0 A[3,2]=0 A[3,3]=0

Blok-sxemasi fragmenti:



Turbo Pascaldagi dasturi:

```
Program Modify;
  Var A          : Array[1..10, 1..10] of Real;
      N, i, j : Integer;
```

```
Procedure InputOutput;
  Begin
    ReadLn(N);
     For i := 1 to N do
         For j := 1 to N do
             begin Write('A[' , i , ',' , j , '] = ');
                       ReadLn(A[i, j])
            end;
     For i := 1 to N do
        begin
             For j := 1 to N do Write(A[i, j] : 5 : 1);
             WriteLn
        end;
  End; { of InputOutput }
{-----------------------------------------}
Procedure Line(Var i : Integer);
  Var Flag : Boolean;
  Begin
    j := 1; Flag := FALSE;
     While (j<=N) and not Flag do
         If A[i, j]<0 then Flag:=TRUE else j:=j+1;
     If Flag then
         For j := 1 to N do A[i, j] := 0
End;
{-----------------------------------------}
Procedure OutRes;
  Begin
    WriteLn(' Natija- Matritsa:'); WriteLn;
     For i := 1 to N do
        begin
             For j := 1 to N do Write(A[i, j]:5:1);
             WriteLn
        end; ReadLn
  End; { of OutRes }
BEGIN
   InputOutput;
   For i := 1 to N do Line(i);
   OutRes;
   END.
```

Xulosa

Foydalanilgan adabiyotlar.

1. Абрамов С.А. и др. Задачи по программированию.-М.:Наука, 1988.-224 стр.
2. Gulomov S.S. va boshqalar. Axborot tizimlari va texnologiyalari. Toshkent, 2000
3. Ахо А., Хопкрофт Дж. Построение и анализ вычислительных алгоритмов. - М: Мир, 1979 г., 535 с.
4. Вирт Н.. Алгоритмы и структуры данных. – Досса, Хамарайан, 1997.
5. Кнут Д. Искусство программирования для ЭВМ. Основные алгоритмы.-М: Мир, 2000 г.
6. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. М.: МЦНМО, 2001.- 960 с.
7. Лебедев В.И. Введение в системы программирования. М: Статистика, 1975.
8. Поляков Д.Б., Круглов И.Ю. Программирование в среде Turbo Pascal: Справ.-метод. пособие.- М.: Изд-во МАИ, 1992.-576 с.
9. Попов В.В. Общение с ЭВМ на естественном языке. М:Наука, 1982.
10. Тыугу Х. Концептуальное программирование. М: Наука, 1984.
11. Успенский В.А., Семенов А.Л.. Теория алгоритмов: основные открытия и приложения. – М: Наука, 1987, 287 с.
12. Файсман А. Профессиональное программирование на Турбо-Паскале.- Info&F, 1992.-270 стр.