ՀԱՅԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅԱՆ
ԿՐԹՈՒԹՅԱՆ ԵՎ ԳԻՏՈՒԹՅԱՆ ՆԱԽԱՐԱՐՈՒԹՅՈՒՆ

ՀԱՅԱՍՏԱՆԻ ՊԵՏԱԿԱՆ ՃԱՐՏԱՐԱԳԻՏԱԿԱՆ
ՀԱՄԱԼՍԱՐԱՆ

Ռ.Վ.Աղգաշյան
Վ.Պ.Ղուկասյան

ԾՐԱԳՐԱՎՈՐՈՒՄ ԲՈԼՈՐԻ ՀԱՄԱՐ

ՄԱՍ I

ԱԼԳՈՐԻԹՄՆԵՐԻ ԿԱՌՈՒՑՈՒՄ

ՈՒՍՈՒՄՆԱԿԱՆ ՁԵՌՆԱՐԿ

Երևան 2000

УДК 681.3.06
Ա-48

Ձեռնարկը նախատեսված է քոմփյութերային ծրագրավորման ուսուցման համար: Առարկան մատուցված է անսովոր, սակայն բոլորին մատչելի՝ պատկերավոր ձևով, գծանկարների միջոցով: Նման մոտեցումը հասանելի է դարձնում շարադրվող նյութը մարդկանց լայն շրջանակներին՝ սկսած դպրոցական հասակից մինչև պատկառելի հասակը:

Գրախոսներ՝ պրոֆ. Յու.Այվազյան
դոց. Ս.Ավետիսյան

Խմբագիր՝ Ն.Խաչատրյան

ԲՈՎԱՆԴԱԿՈՒԹՅՈՒՆ

Ն Ե Ր Ա Ծ Ու Թ Յ Ու Ն

Ծրագրավորումը նույնքան հին է, որքան մարդկության պատմությունը: Յուրաքանչյուր մարդ իր գիտակցական կյանքում ղեկավարվում է բազմաթիվ ծրագրերով, երբեմն չգիտակցելով այդ հանգամանքը: Դա բացատրվում է նրանով, որ գործողությունների մեծ մասը կատարվում է ենթագիտակցորեն, այսինքն առանց հիմնավորելու, թե ին-չու՞ այսպես, և ոչ թե այնպես: Սակայն դա չի նշանակում, որ ենթագիտակցությունը գործում է տարերայնորեն: Ամենևին: Չէ՞, որ այն ձևավորվում է մանուկ հասակից, բազմաթիվ ծրագրեր փորձարկելով և լավագույնը ընտրելով: Օր օրի, համաձայն ընտրված ծրագրի, միևնույն գործողությունները կատարելով և համոզվելով նրանց հարմարավետության մեջ, մարդու գիտակցությունը ընդունում է այդ ծրագիրը և ՚գրանցում՚ ուղեղի համապատասխան բջիջներում, որ հարմար պահին գործածի այն:

Ասածը հիմնավորելու համար կարելի է բերել բազմա–թիվ օրինակներ: Հիշեք, թե ինչպե՞ս էիք առաջին անգամ մենակ անցնում փողոցը՝ դպրոց գնալու կամ փողոցի հանդիպակաց կողմում գտնվող խանութում գնումներ կա–տարելու համար: Հավանաբար, դուք, ինչպես և մյուսնե–րը, բազմաթիվ անգամ կրկնել եք այն ծրագիրը, որը ձեզ թելադրել է ձեր ծնողը, և հետո ոտքը դրել փողոցի վրա. այն է՝ սկզբում նայել ձախ և, եթե մեքենա չի երևում կամ հեռու է, ապա անցնել փողոցի առաջին կեսը մինչև միջ–նագիծ: Կանգնել և նայել աջ. եթե նույնպես մեքենա չի երևում կամ հեռու է, ապա անցնել ճանապարհի երկրորդ հատվածը: Եթե որևէ հատված անցնելիս նկատել եք մո–տեցող մեքենա, ապա, ըստ երևույթին, սպասել եք մինչև մեքենան անցնի, որից հետո կատարել ձեր հաջորդ քայլը:

Եվ այսպես, ամեն օր իրականացնելով նկարագրված ծրագիրը, դուք վստահություն եք ձեռք բերել փողոց անց–նելու ասպարեզում և մի օր էլ նկատել, որ փողոցը անց–նում եք ՚բնազդաբար՚, առանց վերը նշված ծրագիրը հի–շատակելու: Դա արդյունք է այն բանի, որ ծրագիրը հա–

ջողությամբ փործարկումներ անցնելուց հետո, վերջա–պես, գտավ իր տեղը ենթագիտակցության մեջ և այլևս նման ծրագիր կազմելու կամ հիշատակելու մեջ անհրա–ժեշտություն չկա:

Յուրաքանչյուր անձ ամեն օր ստիպված է լուծել տար- բեր բնույթի բազմաթիվ խնդիրներ: Եթե որևէ խնդրի նմանը նախկինում արդեն հանդիպել և հաջողությամբ լուծվել է, ապա իրականացված մեթոդը կամ նրա հիմքում ընկած գաղափարը կարելի է կիրառել նոր պայմաններում, միգուցե չնչին փոփոխություններով: Հակառակ դեպքում տվյալ անձը ստիպված է կազմելու նոր ծրագիր առաջացած խնդիրը լուծելու համար: Իսկ այդ ծրագրի արդյունավե- տությունը, բնական է, կախված է իր՝ հեղինակի տրամա- բանորեն դատելու, բազմաթիվ պայմաններ հաշվի առնելու և ճիշտ համակցելու ունակությունից: Եթե բոլոր պայման- ները հաշվի են առնվել և ճիշտ հերթականությամբ են իրագործվում, ապա կազմած ծրագիրը կաշխատի անթերի, հակառակ դեպքում, երբեմն այն կբերի սխալ արդյունքի:

Քիչ է պատկերացնել խնդրի լուծումը, լուծման բազ– մաթիվ տարբերակները. անհրաժեշտ է հստակ շարադրել տվյալ խնդրի լուծման **ալգորիթմը**, այսինքն՝ գործողու– թյունների այն հաջորդականությունը, որի իրականացումը կբերի խնդրի ճշգրիտ լուծմանը: Յուրաքանչյուր անձ գոնե իր համար պետք է հստակեցնի լուծման գործընթացը, հատկապես երբ այն պետք է հաղորդվի մեկ այլ անձի, առավել ևս, համակարգչին: Բազմիցս հանդիպում ենք մարդկանց, որոնք դժվարանում են մտքերը հստակ ձևա- կերպելու հարցում. այն դեպքում, երբ նրանք հիանալի տիրապետում են այն լեզվին, որով մտածում են: Ուրեմն պատճառը ո՞չ թե ստվար բառապաշարն է, այլ հստակ մտածելակերպը, կարելի է ասել՝ ալգորիթմի բացակայու- թյունը. զուր չէ ասված. “Ով պարզ մտածում է, նա պարզ շարադրում է”: Փորձեք, օրինակ, որևէ մեկին հարցնել, թե **a, b, c** երեք թվերից ո՞րն է ամենամեծը և ինչու՞: Եթե առաջին հարցի պատասխանը կստանաք գրեթե անմիջա- պես, ապա երկրորդինը համոզիչ չի լինի, քանի որ բոլորը ունակ չեն քայլ առ քայլ շարադրելու մեծագույն արժեքի որոշման ալգորիթմը: Այստեղ բավական չէ ասել թե մեծագույն արժեքը որոշվում է թվերի զույգ–զույգ համե- մատություններով, էականը՝ այդպիսի համեմատությունների հաջորդականությունն ու փոխկապակցությունն է, այսինքն, ալգորիթմն է: Այդ հանգամանքը որոշիչ է դառնում, երբ երեք թվերի փոխարեն վերցնում ենք ավելի մեծ բազմություն:

Համակարգիչների կիրառումը զգալի ազդեցություն է գործում մարդու մտածելակերպի, նրա տրամաբանության զարգացման վրա: Տարիներ շարունակ նույնատիպ խնդիրներ միևնույն ձևով լուծելուց հետո, որքա՜ն անսպասելի և հաճելի է հայտնաբերել, որ գոյություն ունեն նույն խնդրի այլ, անսովոր, միգուցե, ավելի գեղեցիկ լուծումներ, որոնք կապված են միմիայն համակարգիչների յուրահատուկ տրամաբանության հետ: Իսկ այդ յուրահատկությունը պայմանավորված է գործընթացի մեջ ենթագիտակցականի ներգրավումով, այն հարուստ փորձով և գիտելիքներով, որոնք ձեռք են բերվել անցած տարիների ստեղծագործ աշխատանքի շնորհիվ: Միմիայն գիտակցորեն կիրառելով ենթագիտակցականը կարելի է լավագույնս պատկերացնել երևույթը, տալ նրա ճիշտ լուծումը և ճշգրիտ ձևով նկարագրել այն:

Ծրագրավորումը սկսվում է վերը նշված պարզ խնդիրների ալգորիթմների կառուցումով: Համապատասխան ունակություններ ձեռք բերելուց հետո կարելի է անցնել ավելի բարդ, բազմաթիվ պայմաններ ընդգրկող, խնդիրների ալգորիթմների կառուցմանը կամ այնպիսի խնդիրների, որոնց լուծումները ներկայացվում են կրկնվող գործողությունների հաջորդականությամբ, այսպես կոչված, **ցիկլերով**:

Գոյություն ունեցող ձեռնարկները, հիմնականում, հետևողական չեն առարկան մատչելի ձևով ներկայացնելու, ծրագրավորման տրամաբանությունը լիովին բացահայտելու հարցում: Ծրագրավորմանը նվիրված գրեթե բոլոր դասագրքերի ուսումնասիրության հիմնական առարկան է` այս կամ այն ծրագրավորման լեզուն, մոռանալով այն հանգամանքը, որ լեզուն միմիայն **միջոց է** ալգորիթմները նկարագրելու համար: Անհրաժեշտ գործողությունները ընտրելուց և կարգավորելուց հետո միայն կարելի է խոսել ծրագրավորման մասին:

Սույն ձեռնարկը նախաձեռնվել է նշված բացը լրացնելու նպատակով: Այն հատկապես օգտակար կլինի ծրագրավորումը ինքնուրույն ուսումնասիրողների համար` առանց մասնագիտացված դասընթացներ հաճախելու: Այստեղ հիմնական ուշադրությունը հատկացված է ալգորիթմների դասակարգմանը, նրանց կառուցմանը, միևնույն խնդրի լուծման տարբեր ալգորիթմների համեմատությանը` լավագույնը ընտրելու նպատակով: Ալգորիթմները ներկայացնելու համար ընտրված է վաղուց ընդունված և տարածում գտած գրաֆիկական մի միջոց, որը իրականացվում է բլոկ–սխեմաների տեսքով: Տվյալ ընտրությունը բացատրրվում է այն հանգամանքով, որ մարդկանց զգալի մասը ավելի լավ է ընկալում պատկերավոր բնույթի ինֆորմացիան, քան թե տեքստի միջոցով շարադրված երևույթը: Այսինքն, ալգորիթմը ավելի մատչելի է, եթե այն գծագրված է, այլ ոչ թե նկարագրված ծրագրավորման որևէ լեզվով:

Ձեռնարկի երկրորդ մասը նվիրված է ծրագրավորման՝ Տուրբո–Պասկալ համապիտանի լեզվի ուսուցմանը, որը թույլ կտա արդեն կառուցված ալգորիթմները փորձարկել համակարգչի վրա: Տվյալ լեզվի ընտրությունը արդարացված է իր պարզությամբ և բնական էությամբ, այսինքն խոսակցական լեզվին առավելագույնս նմանությամբ: Չէ՞ որ այդ լեզուն Վիրտի կողմից ստեղծվել է ծրագրավո–րումը հեշտ ու արագ ուսուցանելու նպատակով: Այդ իսկ պատճառով ծրագրավորման սկզբունքները նպատակա–հարմար է դասավանդել նշված լեզվով: Ծրագրավորման այլ լեզվի երկրպագուները թող չվշտանան, քանի որ ծրագրավորման հիմունքները, այն է՝ ալգորիթմների կա–ռուցման սկզբունքները յուրացնելով և լավ տիրապետելով Տուրբո–Պասկալ տիպի բարձր մակարդակի լեզվին, ծրագրավորման այլ լեզվի ինքնուրույն յուրացումը ոչ մի դժվարություն չի ներկայացնի:

Բնական է, յուրաքանչյուր լեզու պահանջում է իրեն բնորոշ մտածելակերպ, որը և թելադրում է խնդիրների լուծման այս կամ այն ալգորիթմը: Ուրեմն, որպեսզի որևէ լեզվի հնարավորությունները լիարժեք օգտագործվեն ձեր պրակտիկայում, դուք պետք է լիովին ըմբռնեք տվյալ լեզ–վի տրամաբանությունը, նրա ընձեռած միջոցները: Իսկ դրան դուք կարող եք հասնել հետևողական ու համբերատար աշխատանքով՝ սկսած պարզագույն վարժությունների կատարումից մինչև վերջնամասում առաջարկվող ավելի բարդ կամ դժվարագույն խնդիրները:

Առարկան լավագույնս յուրացնելու համար առաջար–կվում է յուրաքանչյուր բաժնի նյութը ուսումնասիրելուց հետո ինքնուրույն կատարել տվյալ բաժնին կից բոլոր առաջադրանքները առանց վերապահումների: Սկզբնական շրջանում դժվարությունները անխուսափելի են, ինչը բխում է առարկայի յուրահատուկ տրամաբանությունից, պահանջվող անսովոր մտածելակերպից: Սակայն, համոզ–

ված ենք, որ ամենօրյա համառ աշխատանքով դուք ընդունակ եք հաղթահարելու այն անվստահությունը և կասկածամտությունը սեփական ուժերի նկատմամբ, որոնք առաջին քայլերում կարող են ձեզ պատել:

Եթե ամեն ինչ հաջող ընթանա, առաջին բաժինը յուրացնելիս դուք կսովորեք մտածել և խոսել այն ոճով, որը բնորոշ է ծրագրավորողին: Երկրորդ բաժինը յուրացնելով, դուք կսովորեք գրել համակարգչին մատչելի մի լեզվով: Ալգորիթմների կառուցման և նկարագրման զգալի փորձ ձեռք բերելուց հետո ձեր հետագա գործունեության ընթացքում, դուք, միգուցե, չզգաք բլոկ–սխեմաների կարիքը յուրաքանչյուր նոր խնդիր լուծելիս` դա կլինի լավագույն վկայությունը այն բանի, որ մենք հասանք մեր նպատակին: Սակայն բլոկ–սխեմաների կառուցման փորձը ձեզ կարող է պետք գալ բարդ, տրամաբանական հանգույցներ հստակ պատկերացնելու և դրանց ճիշտ լուծումներ տալու համար: Մյուս կողմից, ինչու՞ չէ, կգա մի օր, որ դուք ցանկություն կունենաք ձեր փորձը փոխանցել մյուսներին, իսկ եթե ոչ` ընդունենք այն պարզ թեզը, որ **ոչ մի գիտելիք ավելորդ չի լինում**:

Այսպիսով, մոտիկ ապագան բավականին գայթակղիչ է, ուրեմն առա՜ջ:

1. ԱԼԳՈՐԻԹՄԻ ՀԱՍԿԱՑՈՒԹՅՈՒՆ ԵՎ ՆԵՐԿԱՅԱՑՄԱՆ ՁԵՎԵՐ

Հետագա նյութը կարդալիս մի փորձեք այն վերագրել համակարգչին, այլ պատկերացրեք, որ ձեր առաջ նստած է մի երեխա, որին դուք պետք է բացատրեք զանազան խնդիրների լուծման եղանակները: Բնական է, դուք կընտրեք նախ և առաջ պարզ միջոցներ, որոնք կբացառեն երկ-մտությունը, հետևաբար և ավելորդ հարցերը: Նման դեպ-քերում մեծահասակներս, սովորաբար, դիմում ենք պատկե-րավոր ձևերին՝ օգտագործելով նկարներ, համեմատություն-ներ և այլն: Սակայն պարզվում է, որ նկարներ սիրում են ոչ միայն երեխաները, այլ նաև մեծահասակները: Հետևաբար, մենք կփորձենք շարադրել հետագա նյութը յուրահատուկ նկարների՝ գծանկարների միջոցով:

Ցանկացած խնդիր լուծելիս մեզանից յուրաքանչյուրը մտաբերում է այն մեթոդները, որոնք կարելի է կիրառել տվյալ դեպքում: Ընտրելով լավագույնը՝ մենք որոշում ենք այն անհրաժեշտ գործողությունները որոնք պետք է կա-տարվեն քայլ առ քայլ մեթոդը իրականացնելու համար և դասավորում դրանք որոշակի հերթականությամբ՝ ինչպես վերը ասվեց, կառուցում ենք **ալգորիթմը:** Ըստ խնդրի բնույթի գործողությունները կարող են արտահայտվել տարբեր ձևերով, օրինակ, մաթեմատիկական բանաձևերի տեսքով, խոսակցական լեզվի առանձին նախադասությունների շարադրանքով կամ նշված երկու ձևերի կապակցումով: Ընդհանուր դեպքում նշված միջոցներից ոչ մեկը նախ-ընտրելի չէ մյուսների համեմատությամբ, քանի որ նրանցից յուրաքանչյուրը կիրառելի է խնդիրների որոշակի ոլորտում և չի կարող հավակնել ալգորիթմների նկարագրման ունի-վերսալ միջոցի:

Դիտարկենք մի օրինակ:

Դիցուք, առաջարկվում է հաշվել **a, b, c** կողմեր ունեցող եռանկյան մակերեսը: Այս խնդիրը լուծելու հա–մար ընտրենք ամենահարմարը՝ Հերոնի բանաձևը. $S=\sqrt{p(p-a)(p-b)(p-c)}$, որտեղ p–ն եռանկյան կիսապա–րագիծն է: Մակերեսի հաշվարկը քայլ առ քայլ ներկայաց–նելով՝ կստանանք հետևյալ ալգորիթմը **(Ա0.1)**.

1. $p=(a+b+c)/2$;
2. $y=p(p-a)(p-b)(p-c)$;

3. $S=\sqrt{y}$ ։

Այստեղ մենք ենթադրեցինք, որ **a, b, c** մեծությունները հայտնի են և նման երկարություններ ունեցող կողմերով հնարավոր է կառուցել եռանկյունի: Սակայն, հաշվի առ– նելով հակառակի հավանականությունը, գալիս ենք այն եզրակացության, որ կազմված ալգորիթմը թերի է: Ուրեմն, վերը նշված գործողությունները պետք է կատարել միայն համոզվելով, որ եռանկյունը հնարավոր է կառուցել: Յաշվի առնելով այս հանգամանքը, լիարժեք ալգորիթմը կարելի է նկարագրել ոչ միայն բանաձևերով, այլև խոսակցական լեզվի միջոցները կիրառելով, և կստանանք բարդ ստորադասական նախադասություն **(Ա0.2)**.

եթե (a<b+c) և (b<a+c) և (c<a+b), ապա կատարել.

1. $p=(a+b+c)/2$;
2. $y=p(p-a)(p-b)(p-c)$;
3. $S=\sqrt{y}$ *;*
4. *գրել պատասխանը,*

հակառակ դեպքում գրել 'լուծում չկա':

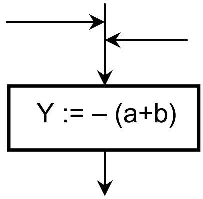
Ինչպես տեսնում ենք, մայրենի լեզուն էլ հստակու–թյուն չավելացրեց ալգորիթմը նկարագրելու հարցում, քանի որ պարզ չէ, թե նշված պայմանի ճիշտ լինելու դեպքում քանի գործողություններ պետք է կատարել. մե՞կ, երկու՞, թե՝ բոլոր չորսը: Այդ պատճառով մենք դիմեցինք կառուցողական որոշ հնարքների և նախադասությունում ընդգրծեցինք կատարվելիք գործողությունները՝ գրելով յուրաքանչյուրը նոր տողից և ձախ եզրից որոշ հեռավորության վրա:

Խոսակցական լեզվին բնորոշ երկիմաստությունից ազատվելու համար նպատակահարմար է այսուհետև ալ– գորիթմները նկարագրել ավելի պատկերավոր միջոցներով, ինչպես վերը նշվել է, գծանկարների լեզվով, որտեղ յուրա- քանչյուր գործողություն ներկայացվում է համապատասխան բլոկի միջոցով: Բայց մինչ այդ եղանակին անցնելը ևս մեկ դիտողություն, կապված '=' (հավասար) նշանի գործածման հետ: Կրկին դիմենք մեր օրինակին: Առաջին երեք գործո- ղություններում մենք կիրառել ենք '=' նշանը, սակայն մենք իրավասու չենք ասելու, օրինակ, '*p–ն* **հավասար է**', քանի դեռ չենք հաշվել աջ մասում բերված արտահայտության արժեքը: Տվյալ պարագայում ճիշտ կլիներ ասել. '*p–ին*

վերագրել’ համապատասխան արտահայտության արժեքը, այսինքն՝ հաշվել ինչ–որ արժեք և վերագրել այն նշված փոփոխականին: Այսուհետև մենք ’=’ նշանը կօգտագործենք առնչություններում՝ պայմաններ ստուգելիս, իսկ ’:=’ նշանը՝ հենց վերագրման գործողությունը նշելու համար: Օրինակ. *եթե x=0, ապա y:=–(a+b)* նախա–դասությունը կարդացվում է հետևյալ ձևով. *եթե x–ը հավասար է զրոի, ապա y–ին վերագրել –(a+b) արտահայտության արժեքը:* Եվ դա կլինի արդարացի:

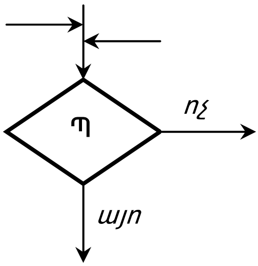
Այժմ ընտրենք այն բլոկները, որոնք պետք է օգտա–գործել բլոկ–սխեմաներ կառուցելիս: Առաջին հայացքից թվում է, թե գործողությունների քանակը այնքան մեծ է, որ յուրաքանչյուրին յուրահատուկ բլոկ համապատասխա–նեցնելը անհնար է: Սակայն դա այդպես չէ: Եթե ուշադիր վերլուծենք վերը բերված օրինակները, ապա կտեսնենք, որ մենք կիրառել ենք ընդամենը երկու տեսակի գործողու-թյուններ. **ա) վերագրման, բ) պայմանի ստուգման ու որոշման կայացման**: Հետագայում դուք կհամոզվեք, որ ընդհանրապես, գրեթե բոլոր խնդիրների լուծման ալգո-րիթմները կարելի է նկարագրել նշված երկու տեսակի գոր-ծողություններով՝ ավելացնելով ևս երկուսը. **գ) տվյալների ներմուծման, դ) արժեքների արտածման**: Չէ՞ որ խնդիրը լուծելու համար անհրաժեշտ է նախապես սահմանել որոշ պարամետրերի սկզբնական արժեքները (մեր օրինակում՝ եռանկյան կողմերի **a**, **b**, **c** երկարություններն են), իսկ խնդիրը լուծելուց հետո (կամ ընթացքում) պետք է ստացված (կամ ընթացիկ) արդյունքները ինչ–որ ձևով գրանցել: Հետագայում հիմնական գործողությունների ցանկը մենք կընդլայնենք՝ ավելացնելով նոր տիպի գործողություններ և իրենց համապատասխան բլոկներով, սակայն այդ ամենը կկատարվի հետզհետե, ըստ անհրաժեշտության:

1.Այսուհետ վերագրման գործողությունը կփոխարի-նենք **վերագրման** բլոկով, որն ուղղանկյուն է և ունի մեկա-կան մուտքային և ելքային սլաքներ: Մուտքային (վերևի) սլաքը նշանակում է, որ տվյալ գործողությունը կարելի է կատարել սլաքին նախորդող գործողությունից հետո: Եթե մուտքային սլաքները շատ են, ապա բոլորը (նկ.1.1) պետք է միանան բլոկին մեկ սլաքով: Յուրաքանչյուր վերագրման գործողությունից հետո կարող է կատարվել մի որոշակի գործողություն, այդ պատճառով բլոկից դուրս է գալիս մի սլաք:



Նկ.1.1. Վերա-գրման բլոկ:

2. Պայմանի ստուգման և որոշման կայացման նախադասությունը (հրահանգը) կարելի է ներկայացնել շեղանկյունով, ինչպես Նկ.1.2–ում: Շեղանկյան մեջ գրվում է այն պայմանը (Պ–ն), որը հարկավոր է ստուգել: Որպես մուտքային անհրաժեշտ է օգտագործել միմիայն շեղանկյան վերին գագաթը, իսկ որպես ելքային՝ մնացած գագաթներից ցանկացած երկուսը, ընդ որում ելքային սլաքները պետք է ուղեկցվեն *'այո'* և *'ոչ'* բառերով: Տվյալ բլոկը փոխարինում է հետևյալ հրահանգին. ***եթե Պ պայմանը առկա է, ապա անցնել 'այո', հակառակ դեպքում՝ 'ոչ' սլաքների ուղղությամբ տեղադրված առաջին բլոկներին՝ համապատասխանաբար:***



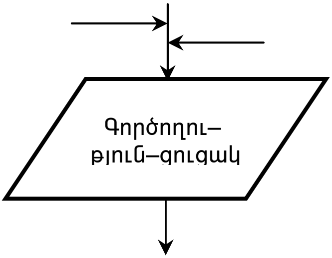
Նկ.1.2. Պայմանական բլոկ:

Այստեղ Պ պայմանի ստուգման արդյունքում որոշում է կայացվում շարունակության ուղղության մասին, ինչը թելադրվում է կոնկրետ խնդրի լուծման ալգորիթմով: Հետագայում այդ երկու ճյուղերը կարող են կրկին հատվել, սակայն կարևորը՝ տվյալ պահին ճիշտ որոշում կայացնելն է, ընտրելով երկու հնարավոր շարունակություններից անհրաժեշտը:

Տեղի սղության պատճառով պայմանավորվենք՝ մեկ շեղանկյան մեջ գրել մեկ հարց, կամ մեկ առնչություն, այն է՝ երկու արտահայտությունների համեմատություն: Հակառակ դեպքում դժվար է պատկերացնել, թե ինչպես կարելի է այսպիսի փոքր բլոկում տեղավորել այնպիսի երկար արտահայտություն, ինչպիսին է վերը բերված Հերոնի բանաձևով եռանկյան մակերեսի հաշվարկման **Ա0.2** ալգորիթմում կիրառվածը:

3. Տվյալների ներմուծման և արտածման համար նախատեսված է կիրառել զուգահեռագիծ (նկ.1.3), որում նշվում է գործողությունը (**ներմուծել** կամ **արտածել**) և թվարկվում են բոլոր այն պարամետրերը, որոնց նկատմամբ պետք է կատարվի նշված հրահանգը:

Առաջին դեպքում կատարվում է բլոկում թվարկված փոփոխականների սկզբնական արժեքների ներմուծում՝ վերը բերված **Ա0.2** ալգորիթմում դա եռանկյան կողմերի՝ **a**, **b** և **c**

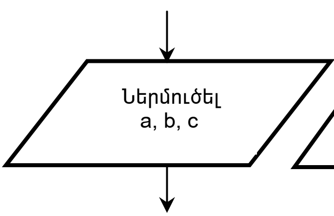


Նկ.1.3. Ներմուծման և արտածման բլոկ:

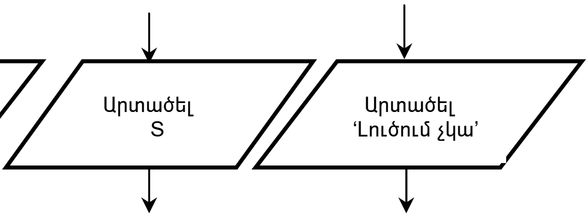
երկարությունները են (նկ.1.4): Երկրորդ դեպքում ցուցակը պարունակում է՝ կամ այն փոփոխականների անունները, որոնց արժեքները պետք է արտածվեն (եռանկյան S–մակերեսը), կամ տեքստեր (նկ.1.5):

Սակայն, եթե հիշու՞մ եք, մենք պայմանավորվեցինք ալգորիթմները առայժմ նկարագրել գծանկարների օգնությամբ, լրիվ բացառելով բառերի, առավել ևս, նախադասությունների գործածումը գործողություններ մեկնաբանելիս: Հետևվելով մեր իսկ կողմից ընդունած կանոններին, մենք, բնական է, պետք է հետագայում տարբերակենք ներմուծման բլոկը արտածման բլոկից, նշված երկու տեսակի բլոկներից լրիվ հեռացնելով պարզաբանող բառերը:

Արտածումը կարող է իրականացվել տարբեր եղա–

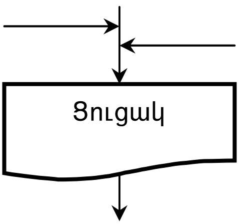


Նկ.1.4



Նկ.1.5

նակներով՝ ձեռքով խնդիրը լուծելիս, մենք, սովորաբար, արդյունքը գրանցում ենք թղթի վրա գրչով կամ մատիտով, իսկ համակարգիչով լուծելիս, արդյունքը մենք կարող ենք ստանալ տարբեր տիպի սարքերի, այսպես կոչված, **կրիչների** վրա. էկրանի (տեսատիպի), թղթի, մագնիսական՝ սկավառակի կամ ժապավենի և այլն: Սակայն տվյալ պարագայում այս հանգամանքը էական չէ: Մենք կարող ենք ընտրել նշված յուրաքանչյուր կրիչը ներկայացնող բլոկը, որպես հավաքական արտածման

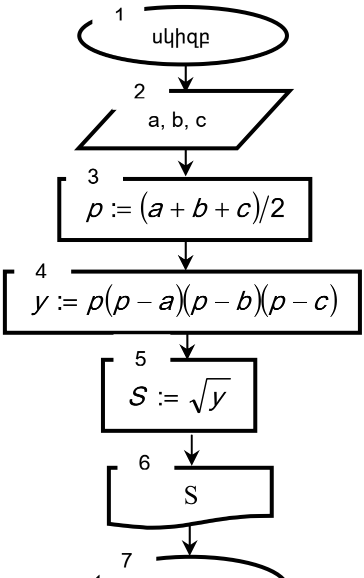


Նկ.1.6. Արտածման բլոկ:

բլոկ, թողնելով վերը ընտրած զուգահեռագիծը ներմուծման համար:

Այսպիսով, արտածման գործողությունը ներկայացնելու համար ընտրենք նկ. 1.6–ում բերված՝ ՛**փաստաթուղթ**՛ կոչված բլոկը, որում բացակայում է ՛**արտածում**՛ բառը: Նման ձևով ներմուծման բլոկից հանենք ՛**ներմուծում**՛ բառը, որին փոխարինելու է զուգահեռագիծը:

Ահա այս չորս բլոկներով մենք պետք է կարողանանք կառուցել զանազան տիպի ալգորիթմներ: Դժվարությունն այն է, որ արտահայտչական միջոցները սուղ են՝ ընդամենը չորս բլոկ, իսկ խնդիրների տեսակները՝ շատ: Սակայն մենք չենք պատրաստվում լուծել բոլոր խնդիրները կամ քննարկել այդ մեծ բազմության որևէ ենթաբազմություն, որն ընդգրկում է որոշակի բնագավառի, ասենք, մաթեմատիկայի, խնդիրները: Ո՞չ և կրկին ո՞չ: Ինչպես նշվեց նախաբանում, մեր նպատակն է՝ ձեզ ծանոթացնել խնդիրների լուծման ալգորիթմների կառուցման սկզբունքներն: Այնպես, որ անկախ խնդրի բնույթից դուք կարողանաք կատարել ալգորիթմի ճիշտ ընտրություն:



Նկ. 1.7. Ա1 ալգորիթմի բլոկ–սխեմա:

Այժմ փորձենք վերը բերված Ա0.1 և Ա0.2 ալգորիթմները ներկայացնել բլոկ–սխեմաների տեսքով, կիրառելով ընտրված բլոկները: Բլոկ–սխեմաներին ամբողջական տեսք տալու համար ընդունված է սխեմաները սկսել և ավարտել օվալաձև բլոկներով, ընդ որում առաջին բլոկում գրում են ՛**սկիզբ**՛, իսկ վերջին բլոկում՝ ՛**վերջ**՛ բառերը:

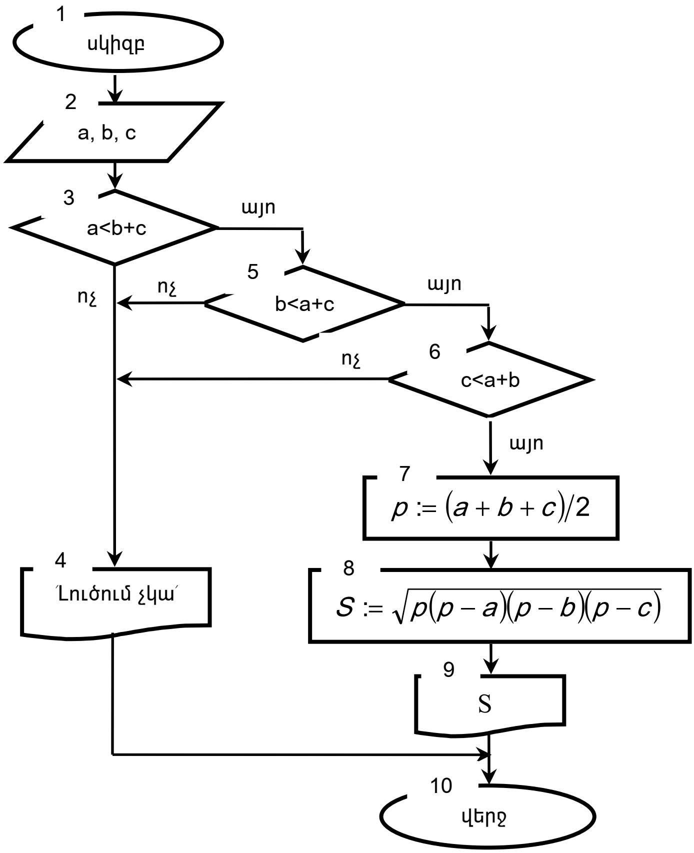
Նկ.1.7–ում բերված է Ա0.1 ալգորիթմի բլոկ–սխեման, որը, ինչպես երևում է, **գծային** բնույթի է, այսինքն բլոկներում նշված գործողությունները կատարվում են հաջորդաբար՝ սլաքի ուղղությամբ, յուրաքանչյուրը մեկական անգամ: Ըստ բլոկ–սխեմայի խնդրի լուծումը ավարտվում է S-արդյունքը արտածելուց հետո:

Նկ.1.8–ում բերված Ա0.2 ալգորիթմի բլոկ–սխեման արդեն ճյուղավորված է, քանի որ սխեման պարունակում է պայմանի ստուգման և որոշման կայացման (պայմանական) բլոկ:

Ի տարբերություն առաջին ալգորիթմի՝ երկրորդում եռանկյան S մակերեսը հաշվել ենք ոչ թե երեք, այլ երկու քայլով, հրաժարվելով միջանկյալ՝ y փոփոխականից: Դա արվել է միմիայն տեխնիկական նկատառումներից ելնելով՝ բլոկների քանակը կրճատելու նպատակով: Հետագայում, երբ որևէ բանաձև ամբողջությամբ կտեղավորվի մեկ բլոկում, նման կրճատումների հաճախակի կդիմենք: Մնացած դեպքերում ստիպված կլինենք երկար բանաձևերը մասնատել և հաշվարկները կատարել մաս–մաս:

Մյուս կողմից, ճյուղավորված ալգորիթմների բլոկ–սխեմաները, որպես կանոն, փռված տեսքի են և, ինչպես երևում է նկ.1.8–ից, դեպի աջ կամ ձախ ուղղված սլաքները հարկ է լինում թեքել, որպեսզի հաջորդ բլոկի մուտքը անպայման կատարվի վերևից: Այսպիսի դեպքերում ընդունված է սլաքները ներկայացնել բեկյալի տեսքով, որի հատվածները փոխադարձ ուղղահայաց են, ընդ որում առաջին հատվածն ունի հորիզոնական ուղղվածություն:

Բլոկ–սխեմաները մեկնաբանելիս հարկ է լինում մատ–նացույց անել այս կամ այն բլոկը: Ամենահարմար միջոցը՝ բլոկների համարակալումն է, որը և իրականացված է ներկայացված երկու գծանկարներում. յուրաքանչյուր բլոկի համարը գրվում է այդ բլոկի վերևի ձախ անկյունում, հատելով սահմանագիծը: Քանի որ ճյուղավորված ալգորիթմներում անհնար է գործողությունները դասակարգել ժամանակի առումով, ապա ընդունված է համարակալումը կատարել սյուն առ սյուն և վերևից ներքև:



Նկ.1.8. Ա0.2 ալգորիթմի բլոկ–սխեմա:

2. ՃՅՈՒՂԱՎՈՐՈՒՄՆԵՐ

Ինչպես հայտնի է, ցանկացած օբյեկտների բազմու–թյուն կարելի է տրոհել որոշակի ենթաբազմությունների՝ հիմնվելով տարբեր չափանիշների վրա: Այսպես, մարդ–կությունը կարելի է բաժանել երկու մասի, ելնելով մարդ–կանց սեռական պատկանելիությունից, կամ վեց մասերի (ենթաբազմությունների), չափանիշ ընտրելով աշխարհա–մասը, որտեղ նրանք բնակվում են: Այս տեսանկյունից մենք կտարբերենք եվրոպացուն ամերիկացուց կամ ավստրա-լացուց և այլն: Իսկ եթե տրոհման չափանիշ ընտրենք պե-տությունը, ապա ենթաբազմությունների քանակը կորոշվի 200–ին մոտ թվով: Այսպես կարելի է երկար շարունակել, սակայն մի կողմ թողնենք մարդկության պրոբլեմները և վերադառնանք մեր հիմնական խնդրին:

Անդրադառնալով խնդիրների բազմությանը, մենք այն կարող ենք տրոհել նույնպես տարբեր չափանիշներով: Օրի-նակ, ըստ գիտության բնագավառի պատկանելության տրո-հումը խնդիրները բաժանում է մաթեմատիկական, ֆիզիկա-կան, փիլիսոփայական և այլ տարատեսակների: Այլ հարց է, թե ի՞նչն է մեզ հետաքրքրում: Իսկ մեզ հետաքրքրում է այն ընդհանուրը, որ միացնում է բոլոր խնդիրները անկախ նրանց բնույթից. այն է՝ խնդիրների լուծման ալգորիթմները: Ահա այս տեսանկյունից դիտարկելով, բոլոր խնդիրները կարելի է բաժանել երկու խմբերի, յուրաքանչյուրում ընդգրկելով.

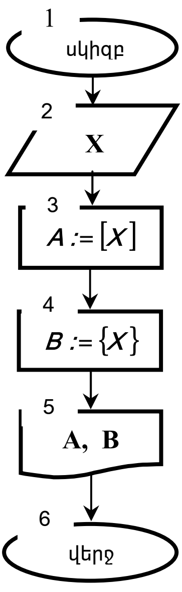
1) այն խնդիրները, որոնց լուծման ալգորիթմը **գծային** է, այսինքն չի պարունակում ոչ մի պայմանի ստուգման և որոշման կայացման գործողություն;

2) այն խնդիրները, որոնց ալգորիթմները **ճյուղավոր–ված** են, այսինքն պարունակում են գոնե մեկ պայմանի ստուգման և որոշման կայացման գործողություն:

Ակնկալում ենք ծրագրավորմանը քիչ թե շատ ծանոթ մարդկանց տարակուսանքը ալգորիթմների միմիայն նշված երկու տիպերի բաժանման առաջարկը ստանալուն պես, քանզի բազմաթիվ հեղինակների մոտ տրոհումը չի սահ-մանափակվում երկու տիպով: Խոսքը գնում է ցիկլիկ բնույթի ալգորիթմների մասին: Հետագայում բազմաթիվ օրինակներով դուք կհամոզվեք, որ ցիկլերը ընդամենը ճյուղավորված ալգորիթմների տարատեսակն են և ոչ թե ինքնուրույն տեսակ: Այլ բան է, որ որոշ տեսակի ցիկլերի նկարագրման համար կիրառվում է հատուկ տեսակի բլոկ: Սակայն դրա մասին ավելի ուշ:

Այս երկու տիպի ալգորիթմների մեկական օրինակներին մենք արդեն ծանոթացանք և նկատեցինք, որ **Ա2** ալգորիթմը իր մեջ ընդգրկում է որպես գծային մաս **Ա1** ալգորիթմը: Ընդհանրապես գծային ալգորիթմները ավելի հաճախ հանդիպում են ճյուղավորվածների կազմում, որպես գծային հատվածներ: Հետևաբար, մեր հետագա ուսումնասիրությունները կապված կլինեն ճյուղավորված ալգորիթմների կառուցման եղանակների հետ: Այդ ընթացքում մենք, կամա թե ակամա, կառնչվենք գծային հատվածներին, իսկ առայժմ դիտարկենք գծային ալգորիթմի ևս մեկ օրինակ, որը ներկայացնում ենք ձեր ուշադրությանը՝ հետագայում օգտագործվող նշանակումները պարզաբանելու նպատակով:

Խնդիր 1: Տրված է X իրական թիվը: Պահանջվում է A փոփոխականին վերագրել X–ի արժեքի ամբողջ մասը, իսկ B փոփոխականին՝ X–ի արժեքի կոտորակային մասը:



Նկ.1.9. Խնդիր 1 ալգորիթմի բլոկ-սխեմա:

Նկ.1.9–ում ներկայացված է տվյալ խնդրի լուծման ալգորիթմի բլոկ–սխեման, որը, ինչպես տեսնում եք, գծային է: Բլոկ 3–ում իրականացվում է A փոփոխականին X–ի արժեքի ամբողջ մասի վերագրում, իսկ բլոկ 4–ում՝ B փոփոխականին նույն արժեքի կոտորակային մասի վերագրում, որից հետո նախատեսված է A և B փոփոխականների ստացած արժեքների արտածում (բլ.5):

Այստեղ X թվի ամբողջ մասը նշել ենք մաթեմատիկայում կիրառում գտած ուղղանկյուն փակագծերով՝ [ ], իսկ կոտորակային մասը նշել ենք ձևավոր փակագծերով՝ { }: Հետագայում ձևավոր փակագծերը կօգտագործենք նաև բաժանման արդյունքում առաջացած մնացորդը նշելու համար: Օրինակ, որպեսզի պարզենք, թե N ամբողջ թիվը պատի՞կ է մեկ այլ՝ K ամբողջ թվին, բնական է բաժանել N թիվը K թվին և առաջացած մնացորդը համեմատել զրոյի հետ: Այսինքն կարճ ձևով տվյալ ալգորիթմը կարելի է ներկայացնել հետևյալ պայմանական նախադասությամբ. *եթե* $\{N/K\}=0$*, ապա ՛պատիկ է՛, հակառակ դեպքում ՛ոչ՛:*

Մասնավոր դեպքում նման ձևով կարելի է պարզել ցանկացած ամբողջ թվի զույգությունը, եթե վերցնենք K=2:

Հետագա նյութը շարադրված է կոնկրետ օրինակների քննարկման հիման վրա: Որոշ դեպքերում առաջարկված են մեկից ավելի լուծումներ, անհրաժեշտության դեպքում` համեմատությունների արդյունքով կատարված է, որոշ իմաստով, լավագույնի ընտրություն, իսկ մնացած դեպքերում ավելի հարմար ալգորիթմի ընտրությունը թողնված է ընթերցողին:

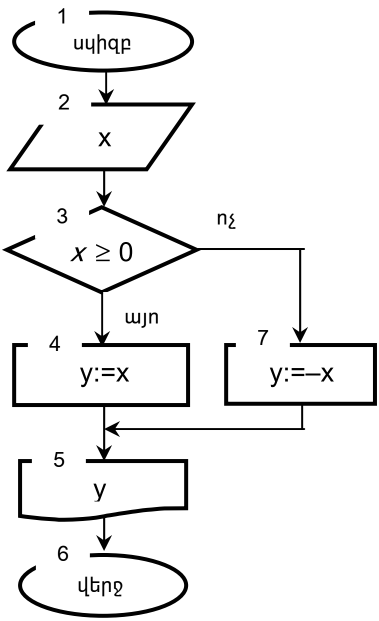
Կարծեք թե ամեն ինչ նախապատրաստված է, և չցանկանալով այսուհետև չարաշահել ձեր համբերությունը, անցնենք մեր հիմնական խնդրի լուծմանը: Սկսենք պարզագույն, առայժմ հաշվարկային տիպի, խնդիրներից:

<u>Խնդիր 2:</u> Դուք, հավանաբար, ծանոթ եք $y = |x|$ ֆունկ–ցիային: Այո, դա x փոփոխականի բացարձակ արժեքի հաշվման ֆունկցիան է, որը որոշվում է հետևյալ ձևով.

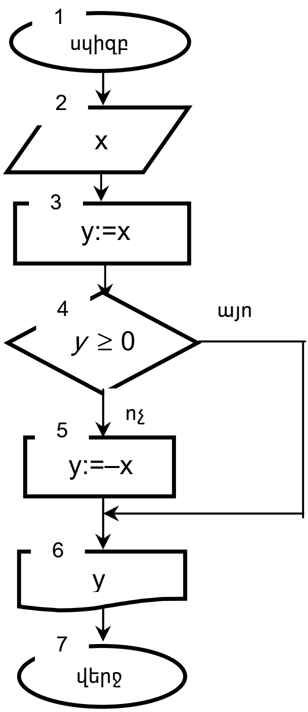
$$y = \begin{cases} x, & \text{եթե } x \geq 0, \\ -x, & \text{եթե } x < 0 : \end{cases} \qquad (1.1)$$

Ֆունկցիայի նման պարզաբանումը կոնստրուկտիվ է, քանզի պարունակում է իր մեջ տվյալ խնդրի լուծման ալ–գորիթմը, որը հնչում է այսպես. ***y–ին վերագրել x–ի արժեքը, եթե այն բացասական չէ, և –x արժեքը, եթե x–ի արժեքը բացասական է***: Ինչպես տեսնում եք, մեր ֆունկ–ցիայի (1.1) նկարագրի համաձայն վերագրման գործողությունը նախորդում է պայմանի ստուգմանը, այն դեպքում, երբ չիմանալով *x*–ի արժեքի նշանը, մենք չենք կարող կայացնել ճիշտ որոշում: Այստեղ և հետագայում մենք նման հրահանգները պետք է ձևափոխենք, տեղերով փոխանակելով վերագրման գործողությունը պայմանի ստուգման հետ: Այսինքն (1.1) տիպի արտահայտությունները պետք է ընթերցել հակառակ հերթականությամբ. ***եթե x–ի արժեքը բացասական չէ, ապա y–ին վերագրել x–ի արժեքը, հակառակ դեպքում` y–ին վերագրել (–x)–ի արժեքը:***

Նկ.1.10–ում բերված է վերը շարադրված Ա2.1 ալգո–րիթմի բլոկ–սխեման, որը, հավանաբար, մեկնաբանու–թյունների կարիք չունի: Ինչպես հիշում եք, մենք խոս–տացել էինք անհրաժեշտության դեպքում առաջարկել որոշ ալգորիթմների այլ, անսովոր լուծումներ: Դրանցից մեկն է` նկ.1.11–ում ներկայացված Ա2.2 ալգորիթմի բլոկ–սխեման: Ինչպես տեսնում եք,երկրորդ ալգորիթմում մենք սկզբից *y* ֆունկցիային վերագրում ենք *x* փոփոխականի արժեքը, որից հետո այն ճշտում ենք, ստուգելով նշանը: Նման հնարքների հաճախակի են դիմում, եթե վերագրվող արժեքը թույլատրելի տիրույթում է: Հակառակ դեպքում առանց պայմանի ստուգման վերագրելը վտանգավոր է:



Նկ.1.10. Ա2.1 ալգորիթմի բլոկ–սխեմա:



Նկ.1.11. Ա2.2 ալգորիթմի բլոկ–սխեմա:

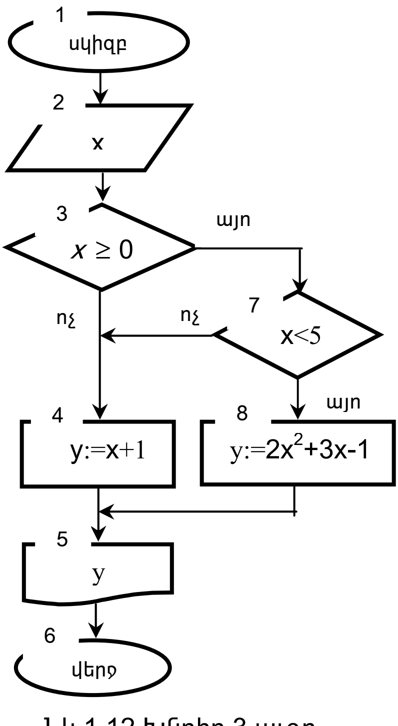
Հետագա խնդիրներում պայմանները հետզհետե ավելանալու են և պետք է կարողանալ ընտրել դրանց ստուգման ճիշտ հերթականությունը:

<u>Խնդիր 3:</u> Առաջարկվող ֆունկցիան որոշված է իրական թվերի ողջ բազմության վրա, սակայն տարբեր հատվածներում հաշվարկվում է տարբեր արտահայտություններով.

$$y = \begin{cases} 2x^2 + 3x - 1, & \text{եթե} \quad 0 \le x < 5; \\ x + 1, & \text{հակառակ դեպքում :} \end{cases} \qquad (1.2)$$

Հաշվի առնելով նախորդ դեպքում արված դիտողությունը՝ խնդրի ձևակերպումը ընթերցենք, սկսած պայմանի նշումով. ***եթե $x \ge 0$ և $x < 5$, ապա ֆունկցիայի արժեքը պետք է հաշվել առաջին բանաձևով, հակառակ դեպքում՝ երկրորդ բանաձևով:*** Այստեղ *և* շաղկապը ենթադրում է նշված երկու պայմանների միաժամանակ կատարում: Եթե պայմաններից որևէ մեկը տեղի չունենա, մենք հարկադրված կլինենք *y–ի* արժեքը հաշվել երկրորդ բանաձևով: Այսինքն նույն ալգորիթմը կարելի է ընթերցել այլ կերպ. ***եթե $x<0$ կամ $x\ge5$, ապա ֆունկցիայի արժեքը պետք է հաշվել երկրորդ բանաձևով, հակառակ դեպքում՝ առաջին բանաձևով:***

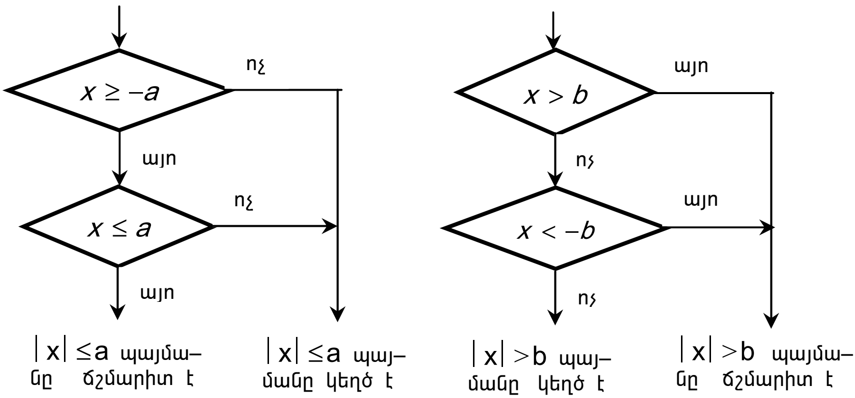
Նկ.1.12–ում ներկայացված է տվյալ խնդրի լուծման ալգորիթմի բլոկ–սխեման, որը համապատասխանում է վերը բերված յուրաքանչյուր ձևակերպմանը: Այստեղ, հնարավորություն չունենալով 3–րդ բլոկում ամբողջությամբ նշելու խնդրի առաջին պայմանը, մենք այն մասնատեցինք երկու տարբեր առնչությունների, որոնք պետք է ստուգվեն առանձին–առանձին, ընդ որում երկրորդ առնչությունը՝ *$x<5$* իմաստ ունի ստուգելու (բլ.7) միայն այն դեպքում, երբ առկա է առաջինը՝ *$x\ge0$* կամ, ինչպես ասում են, երբ առաջին պայմանի ստուգման արդյունքը **ճշմարիտ է** (’**այո**՛ շարունակություն): Հակառակ դեպքին համապատասխանում են նշված յուրաքանչյուր առնչության ստուգման ’**ոչ**’ շարունակությունները կամ, ինչպես ասում են, այն դեպքերը, երբ որևէ առնչության ստուգման արդյունքը **կեղծ է**:



Նկ.1.12.Խնդիր 3 ալգո–րիթմի բլոկ–սխեմա:

Հետագայում մենք հաճախակի կհանդիպենք փոփոխականների բացարձակ արժեքներով ներկայացված պայմանների հետ, ինչպես, օրինակ, $|x| \le a$ կամ $|x| > b$ առընչությունները (*a>0* և *b>0*): Ինչ խոսք, այսպիսի պարզ պայմաններից յուրաքանչյուրը կարելի է ներկայացնել մեկական շեղանկյան մեջ: Սակայն երբեմն, որոշ նկատառումներից ելնելով, նպատակահարմար է մեկ առնչությունը ներկայացնել երկուսի միջոցով: Դրա համար հիշենք, որ $|x| \le a$ առնչության ճշմարտությունը որոշվում է՝ միաժամանակ *(x≤–a)* և *(x≤a)* առնչությունների ճիշտ լինելով *(–a≥x≤a)*, իսկ $|x| > b$ առնչության ճշմարտությունը՝ (*x>b*) *կամ* (*x<–b*) առնչություններից որևէ մեկի ճիշտ լինելով:

Նկ.1.13ա)–ում բերված է $|x| \le a$ առնչության, իսկ նկ. 1.13բ)–ում՝ $|x| > b$ առնչության ստուգման և որոշման կա–



ա) $|x| \le a$ բ) $|x| > b$

Նկ.1.13 Բացարձակ արժեքների հետ կապված պայման–ների ստուգման և որոշում կայացնելու սխեմաներ:

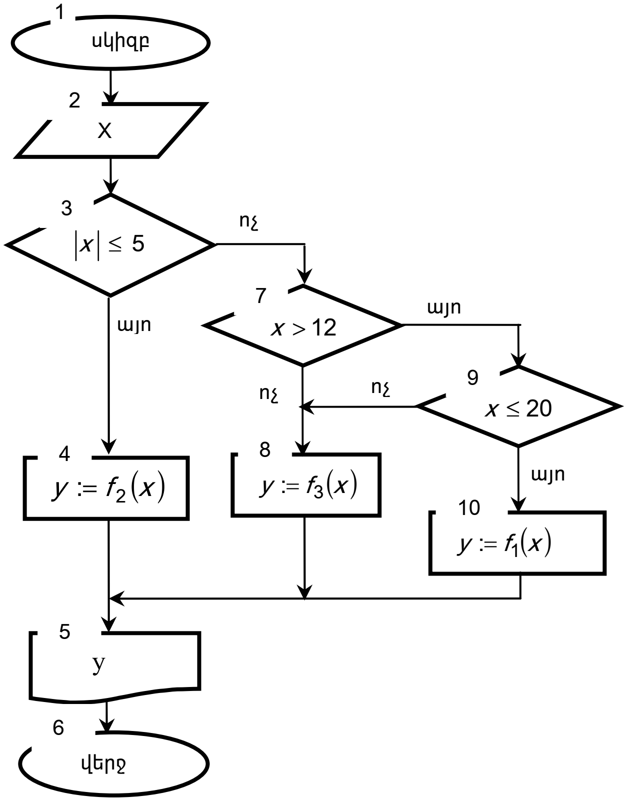
յացնելու սխեմաները, որոնք իրականացված են հիմնական պայմանի մասնատման և երկու պայմանների ստուգման եղանակով:

Այժմ հաջորդ օրինակով հիմնավորենք մասնատման անհրաժեշտությունը:

<u>Խնդիր 4:</u> *X* փոփոխականի կամայական արժեքների համար, ելնելով առկա պայմաններից, առաջարկվում է հաշվել հետևյալ ֆունկցիայի արժեքը.

$$y = \begin{cases} f_1(x), & \text{եթե } 12 < x \le 20, \\ f_2(x), & \text{եթե } |x| \le 5, \\ f_3(x), & \text{մնացած դեպքերում :} \end{cases} \qquad (1.3)$$

Այս անգամ y ֆունկցիայի արժեքները հաշվելու համար ընտրել ենք վերացական՝ $f_1(x)$, $f_2(x)$ և $f_3(x)$ արտահայտություններ, որոնցից մեկի արժեքը պետք է հաշվել x–ի արժեքի համապատասխան պայմանին բավարարելու դեպքում: Ինչպես կարելի էր նկատել նախորդ օրինակներից, նման խնդիրների լուծման ալգորիթմների կառուցվածքը որոշվում է ոչ թե հաշվարկվող արտահայտությունների տեսքով, այլ այն



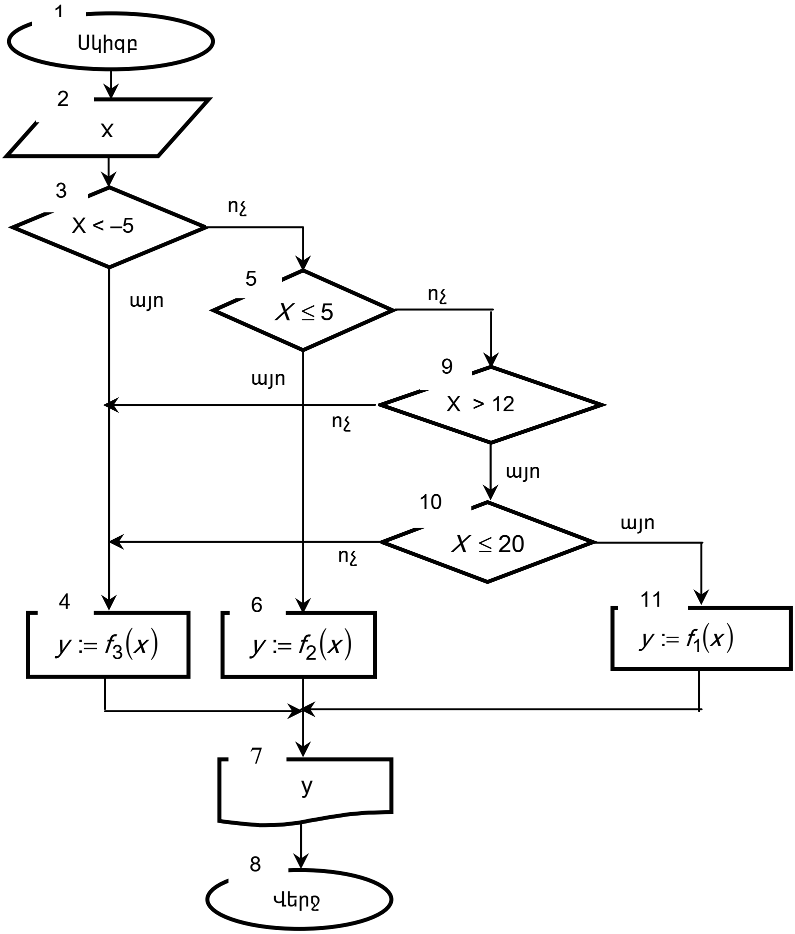
Նկ.1.14. Խնդիր 4 լուծման Ա4.1 ալգորիթմի բլոկ–սխեմա:

պայմաններով, որոնց հիման վրա կատարվում է համապատասխան արտահայտության ընտրությունը:

Նկ.1.14–ում ներկայացված է տվյալ խնդրի ալգորիթմի բլոկ–սխեման, որը կառուցված է անմիջապես արգումենտի բացարձակ արժեքի ստուգման հիման վրա: (1.3) պայման-ներից երկրորդը ստուգելուց հետո (բլ.3), եթե այն ճիշտ է, անցնում ենք 4–րդ բլոկին՝ *y* ֆունկցիային $f_2(x)$ արժեքը վերագրելու համար, հակառակ դեպքում (ինչը համապա-տասխանում է *x*–ի արժեքի **[–5;5]** տիրույթից դուրս գտնվելու պայմանին) նույն անորոշությունը պահպանվում է՝ այն առումով, որ պարզ չէ, թե *x*–ի արժեքը գտնվում է տվյալ հատվածի ձա՞խ, թե ա՞ջ կողմում: Եթե անգամ **x<–5**, միևնույն է անհրաժեշտ է ճշտել *x*–ի դիրքը **(12;20]** հատվածի նկատմամբ: Համեմատելով *x*–ի արժեքը 12–ի հետ (բլ.7), $x \leq 12$ դեպքում, մենք կարող ենք միանշանակ ասել, որ այն համապատասխանում է խնդրի պայմաններից 'մնացած դեպ-քերին', քանի որ մեկ քայլ առաջ պարզել էինք, որ *x*–ի արժեքը **[–5;5]** տիրույթից դուրս է: Հետևաբար, բլ.7–ում ներկայացված առնչության ստուգման '**ոչ**' արդյունքի դեպ-քում անցնում ենք բլ.8–ին՝ համապատասխան արտահայ-տության արժեքը հաշվելու և այն ***y*** ֆունկցիային վերագրելու համար: Նույն առնչության '**այո**' արդյունքի դեպքում մնում է ճշտել $x \in (12;20]$ հարցը, ինչի համար կատարվում է անցում բլ.9–ին, որտեղ վերջնականապես կպարզվի *x*–ի դիրքը թվային առանցքի վրա: x≤20 պարագայում ֆունկցիային վերագրվում է առաջին արտահայտության արժեքը, իսկ հակառակ դեպքում՝ երրորդ: Անկախ նրանից, թե որ արտահայտությամբ ենք հաշվում ֆունկցիայի արժեքը. բլոկներից 4–ում, 8–ում, թե 10–ում, հետագա ընթացքը նույնն է բոլորի համար՝ արդյունքի արտածում (բլ.5) և լուծման ավարտ:

Նկ.1.15–ում բերված է նույն խնդրի լուծման երկրորդ տարբերակը, որում հիմնական պայմանները մասնատված են երկու մասի և ստուգվում առանձին–առանձին: Ինչպես երևում է գծանկարից, Ա4.2 ալգորթմի բլոկ–սխեմայում բլոկների քանակը մեկով ավելին է, սակայն երկրորդ ալգո-րիթմը աշխատում է ավելի արագ, քան Ա4.1–ը: Դա բացա-տրըվում է նրանով, որ երբ *x*–ի արժեքը գտնվում է **[–5;5]** տիրույթից ձախ **(x<–5)**, ապա 3–րդ բլոկից հետո, միանշանակ, y ֆունկցիային պետք է վերագրել $f_3(x)$ արտահայտության արժեքը, անցնելով բլ.4–ին: Խըն–

դրի առաջին պայմանը ստուգվում է միայն, երբ *x*–ի ար–ժեքը գտնվում է **[–5;5]** տիրույթից աջ:



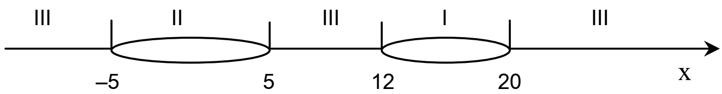
Նկ.1.15. Խնդիր 4 Ա4.2 ալգորիթմի բլոկ-սխեմա:

Նմանատիպ խնդիրներ լուծելիս ալգորիթմի վերջնա–կան տարբերակի ընտրությունը ծերն է: Պետք է միայն գծագրել և համբերատար վերլուծել կառուցված ալգո–րիթմները: Չե՞ որ այս խնդրի ալգորիթմը կառուցելիս մենք կարող էինք համեմատությունները սկսել ոչ թե երկրորդ պայմանից այլ առաջինից: Սակայն մենք վարվեցինք այդպես՝ հետապնդելով այլ նպատակներ:

Ուրեմն, որպես առաջին առաջադրանք՝ փորձեք կառուցել նույն խնդրի լուծման այնպիսի ալգորիթմ, որի աշխատանքը սկսվի առաջին պայմանի ստուգումով: Քանի որ այն բաղկացած է երկու առնչություններից, ապա դուք կարող եք կառուցել ալգորիթմների ևս երկու տարբերակ. մեկում սկսել x–ի արժեքը համեմատելով 12–ի հետ, իսկ մյուսում՝ *x*–ի արժեքը համեմատելով 20–ի հետ:

Ընդհանրապես, նման խնդիրներում լուծումը նպատակահարմար է սկսել երկու եզրային սահմանային կետերից մեկի հետ համեմատությամբ. մեր օրինակում դա –5 և 20 կետերն են: Շարժվելով դեպի մյուս եզրային կետը և հաջորդաբար համեմատելով տիրույթների սահմանային կետերի հետ, դուք ապահովագրված կլինեք որևէ պայմանի անտեսումից:

Այստեղ կարող է շատ բնական հարց առաջանալ. ինչպե՞ս ինքնուրույն համոզվել կառուցված ալգորիթմների աշխատունակության մեջ: Կարող ենք առաջարկել հետևյալ միջոցը: Եթե կրկին անդրադառնանք վերջին օրինակին, ապա նախ և առաջ պետք է $f_1(x)$, $f_2(x)$ և $f_3(x)$ պայմանական արտահայտությունները փոխարինել համապատասխանաբար 1, 2 և 3 արժեքներով: Որից հետո թվային առանցքը բաժանվում է տիրույթների, ինչպես ներկայացված է նկ. 1.16–ում, համարակալելով իրենց՝ համաձայն խնդրում բերված հերթականությանը:



Նկ.1.16. Թվային առանցքի բաշխումը տիրույթների:

Ալգորիթմի աշխատանքը ստուգելու համար անհրաժեշտ է այն ’աշխատեցնել’, հաջորդաբար վերագրելով *x* փոփոխականին կամայական արժեքներ՝ ստեղծված հինգ տարբեր տիրույթներից: Եթե *x*–ի յուրաքանչյուր արժեքի համար ձեր կազմած բլոկ–սխեմայի համաձայն դուք կհասնեք *y*–ին համապատասխան արժեքի վերագրման բլոկին, ապա համարեք, որ տվյալ ալգորիթմը աշխատունակ է: Եթե *x*–ի որևէ արժեքի համար չստանաք անհրաժեշտ արդյունք, ապա դուք պետք է վերանայեք ալգորիթմի գծանկարի այն հատվածը, որի պատճառով ստեղծվել է նման իրավիճակ: Եվ այսպես, քայլ առ քայլ շարժվելով, դուք կարող եք շտկել կազմված ալգորիթմը:

Այն արտահայտությունները, որոնց միջոցով բազմա–թիվ խնդիրներում հաշվարկվում են ֆունկցիայի արժեք–ները, կամ այն առնչությունները, որոնց հիման վրա կատարվում է համապատասխան արտահայտության ընտրություն, բացահայտ ձևով չեն տրվում: Նման դեպքերում մնում է ինքնուրույն կազմել անհրաժեշտ արտահայտությունները, եթե խնդիրը հաշվարկվող բնույթի է կամ որոշել այն գոր-ծողությունների ցանկը և տրամաբանական ընթացքը, որոնց իրականացման դեպքում կարող ենք հասնել նպատակին: Այս առումով դիտարկենք ևս մի քանի օրինակներ:

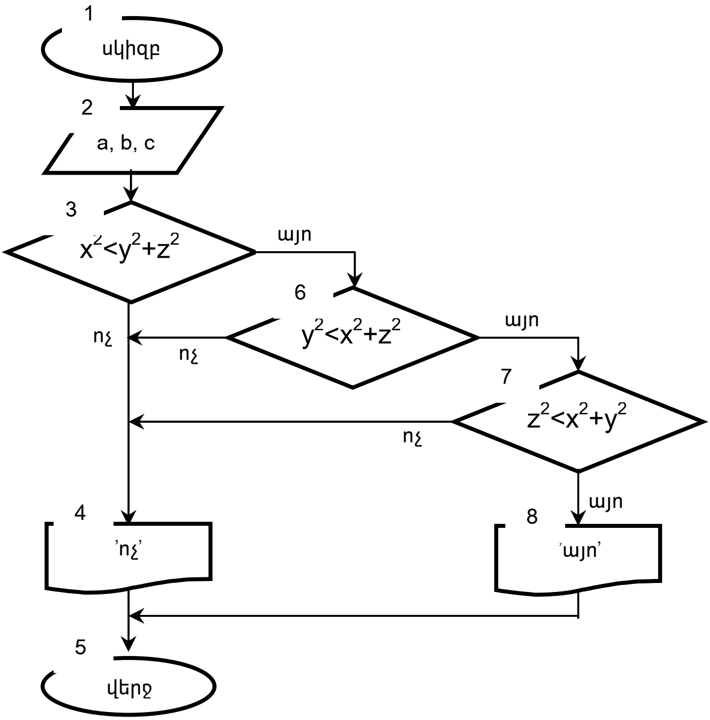
<u>Խնդիր 5:</u> Տրված են եռանկյան կողմերի *x, y*, *z* երկա–րությունները: Պետք է պարզել՝ տվյալ եռանկյունին սուր–անկյու՞ն է, թե՝ ոչ:

Վերհիշելով դպրոցական երկրաչափությունից եռան–կյունիների հետ կապված դրույթները՝ կարող ենք կազմել տվյալ խնդրի լուծման հիմնական պայմանը, այն է. **եթե եռանկյան կողմերից յուրաքանչյուրի քառակուսին փոքր է մնացած երկուսի քառակուսիների գումարից, ապա եռան–կյունին սուր է, հակառակ դեպքում ոչ**: Տվյալ պայմանը մաթեմատիկական առնչությունների ձևով ներկայացվում է այսպես. $(x^2<y^2+z^2)$ *և* $(y^2<x^2+z^2)$ *և* $(z^2<x^2+y^2)$, ինչը ենթա–դրում է երեք առնչությունների միաժամանակյա ճշմարիտ լինելը:

Այսպիսով, մեզ մնում է՝ եռանկյան կողմերի երկարու–թյունների ներմուծումից հետո հաջորդաբար ստուգել նշված առնչությունները, մեկը մյուսի ճշմարիտ լինելու դեպքում: Միայն այսպես կարող ենք ստանալ խնդրում դրված հար-ցի պատասխանը:

Նկարագրված գործընթացը ներկայացված է նկ.1.17–ում: Հիշեցնենք, որ ի սկզբանե մենք ենթադրեցինք, որ *x*, *y* և *z* երկարություններ ունեցող կողմերով հնարավոր է կառուցել եռանկյունի, այլապես մենք ստիպված կլինեինք վարվել, ինչպես Ա0.2 ալգորիթմը կառուցելիս, որի բլոկ–սխեման բերված է նկ.1.8–ում:

Հետագայում, եթե խնդրում բերված ենթադրությունները կհնչեն հարցի ձևով, ապա դրանք ստուգման ենթակա են, և ալգորիթմի աշխատանքը պետք է սկսել այդ ենթադրու-թյուններում համոզվելով, հակառակ դեպքում, մենք կընդունենք նման դիրքորոշումը որպես փաստ, ինչպես Նկ.1.7–ում բերված Ա0.1 ալգորիդմի դեպքում:



Նկ.1.17. Խնդիր 5 լուծման ալգորիթմի բլոկ-սխեմա:

<u>Խնդիր 6:</u> Որոշել տրված՝ *a, b, c* երեք թվերից մեծա–գույնը:

Մեծագույն արժեքի ֆունկցիան ունի իր մաթեմատիկա-կան տեսքը, որն է՝ **max{a,b,c}**, ինչպես նաև փոքրագույն արժեքի ֆունկցիան. **min{a,b,c}**: Սակայն ֆունկցիայի նման ներկայացումը լուծման եղանակի ընտրության հարցում ոչ մի պարզաբանում չի մտցնում: Եթե այս ֆունկցիան հաշ-վարկվող չէ, մնում է նրա արժեքը որոշել տրամաբանական գործողությունների միջոցով, այսինքն՝ համեմատություն-ներով:

Մտաբերենք, թե ինչպես ենք ընտրում առաջարկված երեք խնձորներից մեծագույնը: Հիշեցի՞ք: Այո, մեկը մյուսի հետ համեմատելով. երկուսից ամենամեծը համեմատվում է երրորդի հետ և, վերջապես, մենք ընտրում ենք երկրորդ զույգի հաղթողին: Ով շտապեց, ասելով, թե մեծագույնը ընտրվում է միաժամանակ երեք խնձորները համեմատելով, նա չարաչար սխալվում է: Այն, որ երեք խնձորները միասին գտնվում են մեր տեսողության դաշտում, դեռ չի նշանակում, որ մենք միանգամից համեմատեցինք բոլորն իրար հետ: Եթե ասածը համոզիչ չէ, փորձեք ձեր առջև դնել խնձորներով լի մեկ զամբյուղ և միանգամից ընտրել մեծագույնը: Ահա՜, չստացվե՞ց: Դուք նկատեցի՞ք, որ ձեր աչքերը հաջորդաբար անցնում են մեկ զույգից մյուսին, ընթացքում առանձնացնելով երկուսից ամենամեծը և հաջորդ պահին համեմատելով այն հարակից խնձորի հետ: Եթե ոչ, ապա փորձեք մեկ անգամ ևս, ավելի դանդաղ, և մեկնաբանելով ձեր գործողությունները:

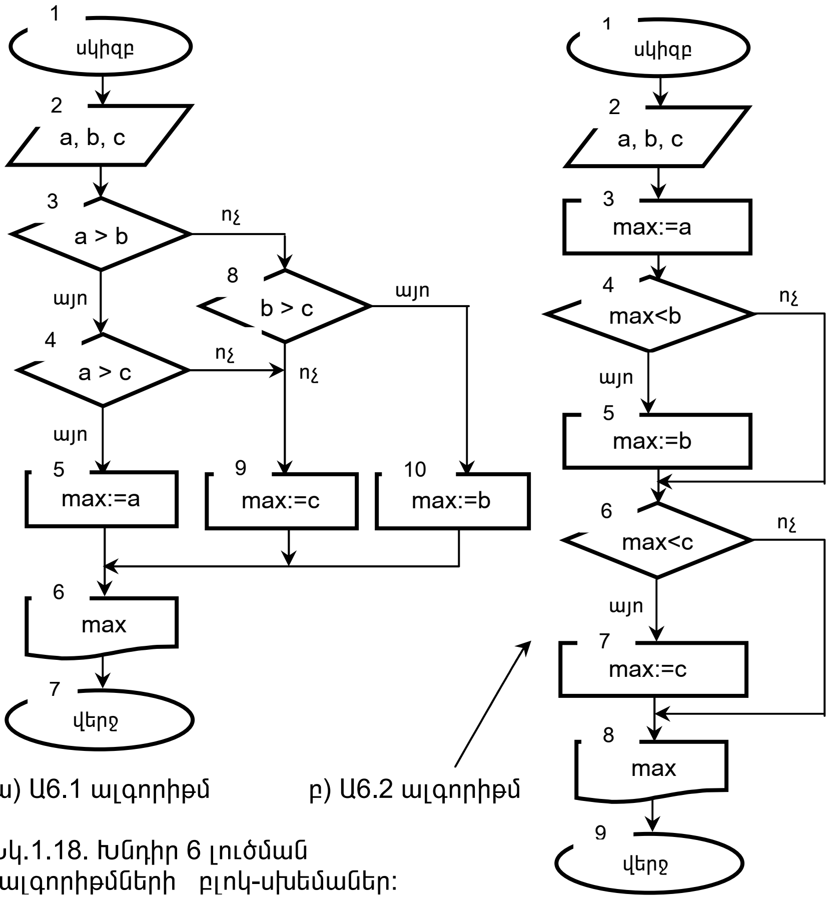
Նկ.1.18–ում բերված են այս խնդրի լուծման ալգո–րիթմների բլոկ–սխեմաների երկու տարբերակներ: Ինչպես տեսնում եք, գծանկարներում Ա6.1 ալգորիթմը կառուցված է վերը նկարագրված եղանակով, այսինքն, առաջին զույգ թվերի համեմատության արդյունքում (բլ.3) ընտրվում է մեծագույնը, որը հետո համեմատվում է երրորդ թվի՝ *c* փոփոխականի արժեքի հետ. եթե դա *a* արժեքն է, ապա անցնում ենք բլ.4–ին, հակառակ դեպքում՝ բլ.8–ին, որոնք և վերջնականապես ընտրում են մեծագույն արժեքը:

Ա6.2 ալգորիթմի գաղափարը, որը իրականացված է նկ. 1.18բ)–ում բերված բլոկ–սխեմայի տեսքով, մեզ արդեն ծանոթ է խնդիր 2–ից, երբ սկզբից ենթադրվում է, որ թվերից որևէ մեկը մեծագույնն է մյուսների նկատմամբ և վերագրվում է **max** ֆունկցիային (բլ.3): Այնուհետև արդեն ֆունկցիայի արժեքն է հաջորդաբար համեմատվում մյուս թվերի հետ՝ սկզբից *b* արժեքի հետ (բլ.4), հետո *c*–ի հետ (բլ.6): Երկու դեպքում էլ, եթե պարզվում է, որ ֆունկցիայի ընթացիկ արժեքը փոքր է համեմատվող թվից, ապա այն փոխարինվում է երկրորդ արժեքով (բլ.5 և բլ.7), հակառակ դեպքում մնում է անփոփոխ:

Մեծագույն և փոքրագույն արժեքների որոշման երկու ալգորիթմների գաղափարները հետագայում կօգտագոր–ծենք տվյալների ավելի մեծ բազմությունների, օրինակ, զանգվածների համար, իսկ առայժմ փորձեք ինքնուրույն պատասխանել Ա6.2 ալգորիթմին վերաբերող հետևյալ երկու հարցերին.

1) երեքից ո՞ր փոփոխականի արժեքը կվերագրվի **max** ֆունկցիային, եթե ներմուծած երեք արժեքները հավասար են (**a=b=c**);

2) նույն պայմանի դեպքում, ինչպե՞ս կփոփոխվի **max** ֆունկցիայի արժեքը, եթե նկ.1.18բ) սխեմայի 4 և 6 բլոկներում ‘<’ նշանը փոխարինենք ‘≤’ նշանով:



ա) Ա6.1 ալգորիթմ

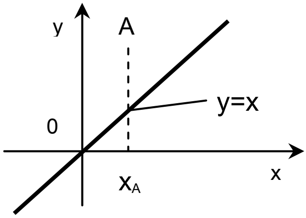
բ) Ա6.2 ալգորիթմ

Նկ.1.18. Խնդիր 6 լուծման ալգորիթմների բլոկ-սխեմաներ:

Պատասխանելով այս երկու հարցերին, համոզված եղեք, որ դուք արդեն կարող եք ինքնուրույն ’ընթերցել’ բլոկ–սխեմաները, պատկերացնելով այն գործընթացները, որոնք նկարագրված են գծանկարի միջոցով:

Այժմ դիտարկենք այլ բնույթի խնդիրներ, որտեղ ձե–զանից պահանջվում են տարրական գիտելիքներ մաթե–մատիկական ֆունկցիաների մասին և, ինչու՞ չէ՝ մի քիչ երևակայություն: Սկսենք պարզագույն օրինակից:

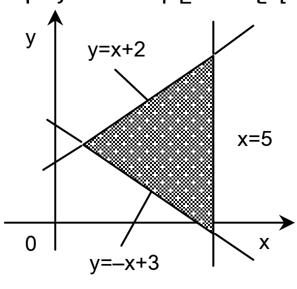
<u>Խնդիր 7:</u> Դիցուք, տրված են A կետի x_A և y_A կոորդի–նատները կամ, ինչպես ընդունված է ասել, տրված է $A(x_A,y_A)$ կետը: Պա-հանջվում է պատասխանել հետևյալ հարցին՝ գտնվու՞մ է արդյոք, տվյալ կետը **y=x** գծից վերև, թե՝ ոչ:



Այսպիսով, պատասխանը պետք է արտահայտվի ’այո’ կամ ՛ոչ’ տես–քով: Սակայն, մինչև հարցին պա-տասխանելը հիշենք, թե ինչպիսի պայմանին (կամ պայ-մաններին) են բավարարում երկրաչափական որևէ գծից վերև գտնվող կետերի կոորդինատները: Ինչպես երևում է խնդրի պայմանը պարզաբանող նկարում, միևնույն x_A կոորդինատ ունեցող երկու կետերից A կետը ավելի բարձր է, քան թե **y=x** ողղի վրա գտնվող կետը՝ այսինքն A կետի y_A կոորդինատի արժեքը ավելի մեծ է, քան **y** արժեքը: Տեղադրելով **y**-ի փոխարեն x_A-ն՝ կստանանք. $y_A > x_A$ պայմանը:

Այժմ խնդրի լուծման ալգորիթմը կարելի է ձևակերպել այսպես. ***եթե*** $y_A > x_A$ ***, ապա պատասխանել ’այո’, հակա–ռակ դեպքում՝ ’ոչ’***: Այս պարզագույն միտքը արտահայտող բլոկ–սխեման փորձեք կառուցել ինքնուրույն: Եթե կդըժվարանաք, ապա կարող եք օգտվել խնդիր 2–ի լուծման՝ Ա2.1 ալգորիթմի բլոկ–սխեմայի օրինակից (նկ.1.10):

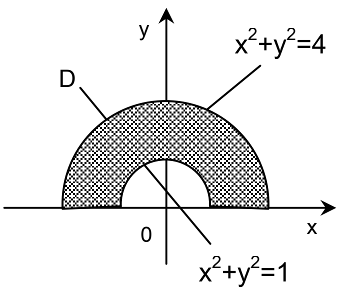
Յաջորդ օրինակներում (8–10) դիտարկվում են ինքն–ամփոփ տիրույթին կետի պատկանելության անհրաժեշտ պայմանները: Նշված բոլոր խնդիրներների ելակետային պայմանը նույնն է. **տրված են՝ D տիրույթը և կամայական $A(x_A,y_A)$ կետը; եթե $A \in D$ (A կետը պատկանում է D տիրույթին) պետք է պատասխանել ’այո’, հակառակ դեպքում՝ ’ոչ’**: D տիրույթը բոլոր խնդիրներում ընդգծ-ված է գունավորման միջոցով:



Խնդիր 8: Այս օրինակում D տիրույթը եռանկյունի է, որի եզրագիծը կազմված է երեք ուղիղ գծերի հատումով (ուղիղների հավասարումները բեր
ված են նկարում):

Օգտվելով նախորդ խնդրում արված դատողություններից՝ կարող ենք ասել, որ եթե A կետը պատկանում է D տիրույթին $(A \in D)$, ապա այն գտնվում է միաժամանակ՝ **y=x+2** ուղղից ներքև, **y=–x+3** ուղղից վերև և, վերջապես, **x=3** ուղղից ձախ: Այսինքն, տվյալ խնդրի լուծման ալգորիթմը հնչում է այսպես. ***եթե*** $y_A < x_A + 2$, $y_A > -x_A + 3$ ***և*** $x_A < 5$, ***ապա A կետը պատկանում է D տիրույթին, հակառակ դեպքում՝ չի պատկանում:*** Բնական է, որ եթե կետը որևէ ուղղի նկատմամբ գտնվի վերը նշված անհրաժեշտ դիրքին հակառակ կողմում, այսինքն, եթե նշված երեք պայմաններից որևէ մեկը (կամ ավելին) տեղի չունենա, ապա կետը կհամարվի տիրույթից դուրս:

Այս խնդրի լուծման բլոկ–սխեմայի կառուցումը, ինչ–պես նաև նախորդինը, իրենց պարզության պատճառով հանձնարարում ենք ձեզ, իսկ մենք անցնենք հաջորդ նմանատիպ օրինակին:

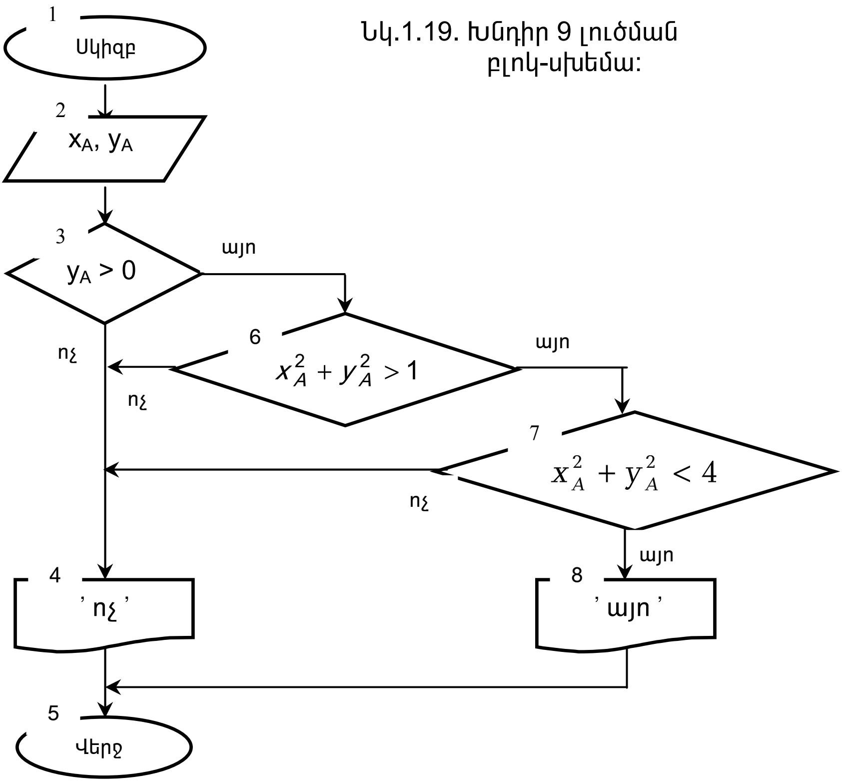


Խնդիր 9: Այստեղ **D** տիրույթը կիսաօղակ է, որի եզրագիծը կազմված է երկու տիպի գծերից. կիսաշրջանակ և ուղիղ գիծ, որի հավասարումն է՝ **y=0:** Արտաքին և ներքին կիսաշրջանագծերի հավասարումները բերված են գծագրում, որտեղից եզրակացնում ենք, որ արտաքին շրջանագծի շառավիղը՝ $R_{\text{ա}}=2$, իսկ ներքին շրջանագծինը՝ $R_{\text{ն}}=1$:

Խնդրի լուծումը ստացվում է պարզ դատողությունների շնորհիվ. նախ ընդգծված տիրույթին պատկանող յուրաքանչյուր **A** կետի **y** կոորդինատը պետք է լինի դրական (y>0), մյուս կողմից՝ տվյալ կետով անցնող կիսաշրջանագծի R_A շառավիղը բավարարում է հետևյալ պայմանին. $R_{\text{ն}}<R_A<R_{\text{ա}}$: Տեղադրելով R_A մեծության փոխարեն՝ $x_A^2 + y_A^2$ արտահայտությունը, նույն երկակի առնչությունը կընդունի հետևյալ տեսքը. $1 < x_A^2 + y_A^2 < 4$:

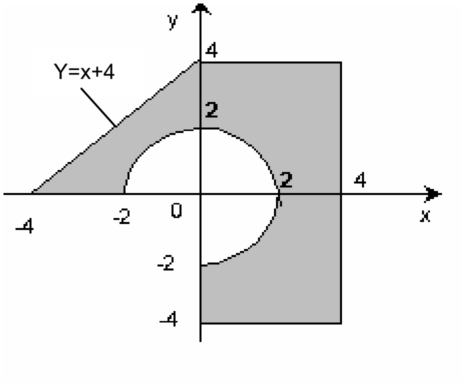
Այսպիսով, խնդրի լուծման ալգորիթմներից մեկ տար– բերակը կարելի է ձևակերպել հետևյալ պայմանական նա– խադասությամբ. ***եթե ($y_A > 0$) և ($1 < x_A^2 + y_A^2 < 4$), ապա 'այո', հակառակ դեպքում՝ 'ոչ',*** որի իրականացման բլոկ–սխեման բերված է նկ.1.19–ում: Իսկ ո՞րն է 'հակառակ դեպքը: Դժվար չէ կռահել, որ 'ոչ' պատասխանը ստացվում է, եթե 1.19 գծանկարում նշված երեք պայմաններից **որևէ մեկը** ճշմարիտ չէ: Հետևաբար, նույն ալգորիթմի այլ ձևակեր-պումը կինչի այսպես. ***եթե ($y_A \le 0$) կամ ($x_A^2 + y_A^2 \le 1$) կամ ($x_A^2 + y_A^2 \ge 4$), ապա 'ոչ', հակառակ դեպքում՝ 'այո':***

Սովորաբար եզրագիծը ընդգրկվում է տիրույթի մեջ: Եթե մեր օրինակում ավելացնենք նման պայման, ապա նկ.1.19–ի 3, 6 և 7 բլոկներում ներկայացված առնչություններում պետք է ավելացնել '=' նշանը:



Նկ.1.19. Խնդիր 9 լուծման բլոկ-սխեմա:

Դաջորդ օրինակում տիրույթի եզրագիծը նույնպես կարելի է բաժանել հատվածների, սակայն նման մոտեցու–մը միայն դժվարացնում է ալգորիթմի ձևակերպումը: Նման դեպքերում ամբողջ տիրույթն են տրոհում մի քանի չհատ–վող ենթատիրույթների, որից հետո կազմում են՝ յուրա–քանչյուր ենթատիրույթին կետի պատկանելության պայ–մանները, որոնք հետո կցվում են միմյանց **'կամ'** շաղկապով, որովհետև միևնույն կետը միաժամանակ չի կարող պատկանել երկու չհատվող ենթատիրույթներին:



Խնդիր 10: Նկարում, D տիրույթը ներքևից սահ-մանափակված է շրջանա-գծի երեք քառորդով, իսկ արտաքին եզրագիծը բե-կյալ է, որը հնարավոր չէ նկարագրել մի հավասա-րումով: Այդ պատճառով ամբողջ տիրույթը տրոհ-վում է ենթատիրույթների: Նկատելի է, որ տիրույթը կարելի է տրոհել *y* առանցքով երկու մասի. ձախ՝ D_1 և աջ՝ D_2: Եթե A կետը պատկանում է ձախ ենթատիրույթին $(A \in D_1)$, ապա նրա կոորդինատները բավա-րարում են միաժամանակ հետևյալ առնչություններին.

$$(x_A < 0),\ (y_A > 0),\ (y_A < x_A + 4) \text{ և } \left(x_A^2 + y_A^2 > 4\right): \qquad (1.4)$$

Եթե A կետը պատկանում է աջ ենթատիրույթին $(A \in D_2)$, ապա նրա կոորդինատները բավարարում են միաժամանակ այլ առնչություններին.

$$(x_A > 0),\ (x_A < 4),\ (|y_A| < 4) \text{ և } \left(x_A^2 + y_A^2 > 4\right): \qquad (1.5)$$

Վերջնականապես A կետի՝ D տիրույթին պատկանե–լության պայմանը ստացվում է վերը բերված երկու պայ–մանների կցումով **'կամ'** շաղկապի միջոցով և հնչում է այսպես. ***եթե (1.4) կամ (1.5) պայմանները առկա են, ապա A կետը պատկանում է D տիրույթին, հակառակ դեպքում՝ ոչ:***

Կարելի է շարունակել նմանատիպ խնդիրների քննար–կումը, սակայն նախընտրելի է, որ դուք ինքնուրույն փորձարկեք ձեռք բերած ունակությունները ճյուղավորված ալգորիթմներ կառուցելու ասպարեզում: Ինքնավարժանքի նպատակով ստորև առաջարկում ենք զանազան տեսակի խնդիրներ, որոնց ալգորիթմների կառուցումը կնպաստի ստացած գիտելիքների ամրապնդմանը և հետագա նյութի հեշտ յուրացմանը:

ՎԱՐԺՈՒԹՅՈՒՆՆԵՐ

Ստորև առաջադրված բոլոր վարժությունների համար պահանջվում է կազմել յուրաքանչյուրի լուծման ալգո–րիթմի բլոկ–սխեման:

1–3 խնդիրներում տառային պարամետրերի կամայա–կան թվային արժեքների համար հաշվել տրված ֆունկցիայի արժեքը.

1) $$y=\begin{cases} x+1, & \text{եթե } -10 < x \le -6, \\ x^2, & \text{եթե } |x| \le 3, \\ x, & \text{մնացած դեպքերքում :} \end{cases}$$

2) $$y=\begin{cases} ax+b, & \text{եթե } a+b > 10, \\ bx-a, & \text{եթե } -15 \le a+b < 2, \\ a+b, & \text{մնացած դեպքերում :} \end{cases}$$

3) $$y=\begin{cases} x+3a, & \text{եթե } |x| \le 4, \\ ax-2, & \text{եթե } 5 < x \le 8, \\ x^2, & \text{եթե } |x| > 20, \\ 3x, & \text{մնացած դեպքերում :} \end{cases}$$

4) Տարեթիվը կոչվում է ՛նահանջ՛, եթե այն անմնացորդ բաժանվում է 4–ի, բացառությամբ այն տարեթվերի, որոնք պատիկ են 400–ին: Պարզել՝ տրված **N** տարեթիվը նահա՞նջ է, թե՝ ոչ:

5) Տրված են՝ **x**, **y** իրական թվերը: Պահանջվում է եր–կուսից նվազագույնը փոխարինել նրանց կիսագումարով, իսկ մեծագույնը՝ արտադրյալի կես արժեքով: Եթե **x=y**, ապա թվերը թողնել անփոփոխ:

6) Տրված են եռանկյան կողմերի x, y, z երկարությունները: Պարզել և արտածել. **3** – եթե եռանկյունը հավասարասրուն է, **2** – եթե այն հավասարակողմ է և **1** – մնացած դեպքերում:

7) Տրված են. $ax^2+bx+c=0$ քառակուսի հավասարման **a**, **b** և **c** գործակիցների արժեքները $(a \neq 0)$: Հաշվել և արտածել հավասարման արմատների արժեքները, իսկ եթե այն լուծում չունի, ապա պատասխանել ’**լուծում չկա**’:

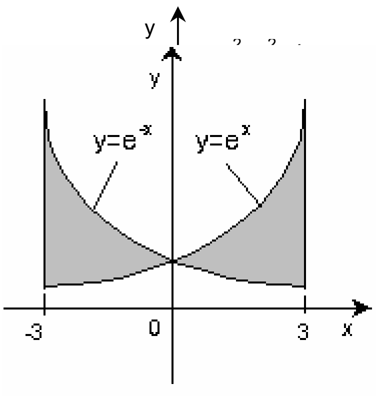
8) Տրված են դրական *a, b, c, d* թվերը: Պարզել. հնարավո՞ր է արդյոք, *a, b* կողմերով ուղղանկյունը ամբողջությամբ տեղադրել *c, d* կողմերով ուղղանկյան մեջ այնպես, որ յուրաքանչյուր ուղղանկյան կողմերը լինեն զուգահեռ կամ ուղղահայաց մյուս ուղղանկյան կողմերի նկատմամբ: Արտածել համապատասխան պատասխանը:

9) Տրված են **ABCD** զուգահեռագծի **A**, **B**, **C**, **D** գագաթների (x_A,y_A), (x_B,y_B), (x_C,y_C) և (x_D,y_D) կոորդի–նատները՝ համապատասխանաբար: Պարզել զուգահեռա–գծի տեսակը և արտածել. **0** – եթե այն ուղղանկյուն է, **1** - եթե այն քառակուսի է և **2** – եթե այն շեղանկյուն է:

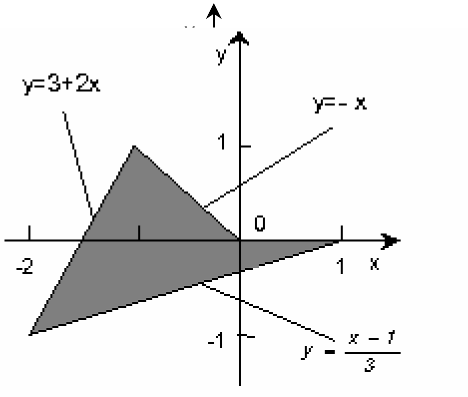
10) Տրված են երկու ոչ համակենտրոն շրջանների O_1 և O_2 կենտրոնների կոորդինատները՝ (x_1,y_1),(x_2,y_2) և շառավիղները՝ R_1, R_2: Պահանջվում է հաշվել և արտածել շրջանագծերի ընդհանուր կետերի քանակը:

11–14 խնդիրներում պահանջվում է պարզել. պատկանու՞մ է, արդյոք, A(x,y) կետը համապատասխան տիրույթներին, թե ոչ:

11)

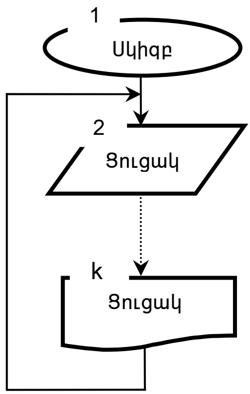


12)



3. ՊԱՐԶ ՑԻԿԼԵՐ

Մինչ այժմ քննարկված և կառուցված ճյուղավորված ալգորիթմներում ներկայացված գործողությունների մի մասը խնդրի լուծման ընթացքում կատարվում է **մեկ** անգամ կամ **չի** կատարվում՝ ելնելով մուտքային տվյալների կոնկրետ արժեքներից: Իսկ եթե դուք խնդիրը լուծում եք մուտքային տվյալների որոշակի տիրույթում, ապա գործողությունների մի խումբ ընդհանրապես չի կատարվի: Այնուամենայնիվ տվյալ գործողությունների առկայությունը ալգորիթմում անհրաժեշտ է, քանի որ մենք պետք է նկարագրենք խնդրի լուծման ընթացքը տվյալների բոլոր հնարավոր արժեքների համար:



Նկ.2.1. Ցիկլ պարզաբանող բլոկ-սխեմա:

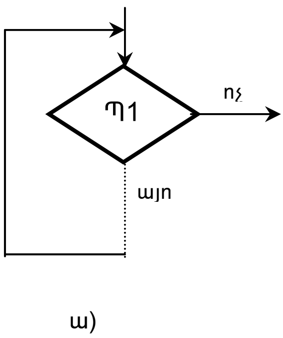
Երբ հարկ է լինում խնդիրը լուծել մուտքային տվյալների այլ արժեքների համար, միևնույն ալգորիթմը պետք է ՛աշխատեցնել՛, կրկին սկսած թիվ 1 բլոկից: Եվ այսպես պետք է վարվել միշտ, անգամ եթե մուտքային տվյալների տարբեր արժեքները ներմուծվում են անընդհատ: Տվյալ գործընթացը ավտոմատացնելու նպատակով կարելի է վարվել, ինչպես նշված է նկ.2.1–ում, այսինքն. յուրաքանչյուր անգամ արդյունքը հաշվելուց և ատածելուց հետո պարտադրել կատարողին՝ վերսկսել ալգորիթմի իրականացումը մուտքային տվյալների արժեքների ներմուծումից: Ըստ բերված սխեմայի ներմուծումը և հետագա ընթացքը, որը նշված է կետագծով, կիրականացվեն ավտոմատորեն: Մեր միջամըտությունը կպահանջվի այն դեպքերում, երբ մուտքային տվյալների ներմուծումը իրականացվի մեր կողմից: Այս դեպքերում, բնական է, ***2÷k*** գործողությունների կրկնությունը կախված է մեր ցանկությունից, ինչը բխում է խնդրի պայմանների թելադրանքից, այլապես նման կառուցվածքով ալգորիթմը ենթակա է անվերջ կրկնվելու:

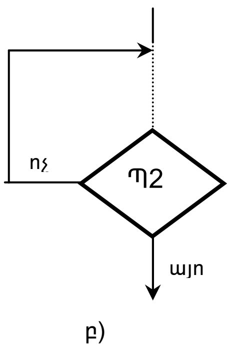
Սակայն միշտ չէ, որ անհրաժեշտ է մուտքային տվյալները ներմուծել: Շատ դեպքերում բավական է նշել հաշվարկվող ֆունկցիայի արգումենտի սկզբնական արժեքը և նրա փոփոխման օրենքը, որպեսզի հետագա հաշվարկներն իրականացվեն ավտոմատորեն՝ առանց մարդու միջամտության՝ արգումենտի մեկ արժեքից ստանալով մյուսը և տեղադրելով արտահայտության մեջ:

Ալգորիթմի այն հատվածը, որն ալգորիթմի իրակա– նացման ընթացքում կրկնվում է մի քանի անգամ, ընդունված է անվանել **ՑԻԿԼ**, իսկ ամբողջ ալգորիթմը՝ **ՑԻԿԼԱՅԻՆ** (**շրջափուլային**): Պարզենք ցիկլերին բնորոշ հատկություն-ները:

Ակնհայտ է, որ ցիկլերը պետք է կրկնվեն վերջավոր քանակությամբ, որի պատճառով ամեն անգամ ցիկլը վեր– սկսելուց առաջ պետք է համոզվել նրա կրկնության ան– հրաժեշտության մեջ: Այդ նպատակով յուրաքանչյուր ցիկլ կազմելիս նախապես պարզում են տվյալ ցիկլի կրկնության կամ ավարտի պայմանները, որոնք, սովորաբար, ստուգվում են ցիկլի սկզբում կամ վերջում, որից հետո միայն ընդուն-վում է համապատասխան որոշում, ինչպես ներկայացված է նկ.2.2–ում: Բերված սխեմաներից ցիկլի ա) տարբերակը կոչվում է **նախապայմանով ցիկլ**, իսկ բ) տարբերակը՝ **հետպայմանով ցիկլ:** Երկու դեպքում էլ միայն պայմանի առկայությունը չի ապահովի ցիկլի վերջավորությունը, եթե ցիկլի կատարման ընթացքում պայմանը բնորոշող փոփո-խականներից որևէ մեկը չփոխի իր արժեքը:

Այսպիսով, ցիկլի աշխատանքը կարելի է նկարագրել հետևյալ կերպ՝ ցիկլը սկսելուց հետո որևէ փոփոխականի արժեք այնպես է փոփոխվում, որ ի վերջո ցիկլի պայմանը ընդունում է ցիկլի ավարտին համապատասխանող արժեք, ինչի արդյունքում ցիկլի աշխատանքը ընդհատվում է:





Նկ.2.2. Ցիկլերի կազմակերպման տարբերակներ:

Հետագայում դիտարկված ա) և բ) կառուցվածքներով ցիկլերի նկարագրման համար կօգտագործենք հետևյալ ա) և բ) դարձվածքները համապատասխանաբար.

ա) *Քանի դեռ Պ1 պայմանը ճշմարիտ է, կատարել ցիկլում ընդգրկված գործողությունները* և

բ) *Կրկնել ցիկլում ընդգրկված գործողությունները մինչև Պ2 պայմանի ճշմարիտ դառնալը:*

Այսուհետև ցիկլում ընդգրկված գործողությունները, բացառությամբ *Պ* պայմանի, կանվանենք **ցիկլի մարմին**:

Այժմ տարբեր օրինակներով ցուցադրենք զանազան տիպերի ցիկլեր կառուցելու հնարքները և համոզվենք, որ նրանք, հիմնականում, կախված չեն խնդիրների բովանդակությունից ու դրանց կիրառման ոլորտից: Սկզբունքը մեկն է՝ որոշել ցիկլի մարմինը կազմող գործողությունները և դրանց իրականացման հաջորդականությունը, պարզել ցիկլի որևէ փոփոխականի, որը այսուհետ կանվանենք **ցիկլի պարամետր**, փոփոխման օրենքը և այն պայմանը, որի դեպքում ցիկլը կրկին պետք է վերսկսել կամ ավարտել: Միայն այսպիսի հստակ դիրքորոշման պարագայում դուք կնվաճեք նաև այս բարձունքը:

<u>Խնդիր 11:</u> Հաշվել և արտածել *y=f(x)* ֆունկցիայի բոլոր արժեքները $D=[x_1,x_2]$ հատվածի վրա Δx քայլով:

Այստեղ ենթադրվում է, որ *y* ֆունկցիան *D* տիրույթում անընդհատ է, այսինքն՝ *x* արգումենտի յուրաքանչյուր արժեքի համար որոշված է: Հակառակ դեպքում *D* հատվածը կարելի է տրոհել չհատվող ենթահատվածների, որոնցում ֆունկցիան խզում չունի, և խնդիրը լուծել առանձին–առանձին՝ յուրաքանչյուր նման հատվածի համար:

Կարևոր չէ, թե ինչի՞ համար են հաշվվում ֆունկցիայի արժեքները որևէ տիրույթում, մեզ համար կարևոր է, թե ինչպե՞ս: Այնուամենայնիվ, հաշվարկված կետերը կարելի է օգտագործել տվյալ ֆունկցիայի գրաֆիկը կառուցելու համար: Հանրահաշվի տարրական կուրսից հայտնի է, որ նման դեպքերում անհրաժեշտ է արգումենտի արժեքը փոփոխել սկզբնական արժեքից մինչև իր վերջնականը Δx դրական քայլով և յուրաքանչյուրի համար հաշվել տվյալ ֆունկցիայի արժեքը: Այստեղ էական չէ, թե *D* հատվածի վրա ի՞նչ ուղղությամբ ենք ընթանում՝ x_1–ից դեպի x_2–ը, թե հակառակ: Դրանից հարևան կետերի հարաբերական դիրքը չի փոխվում և միացնելով հարևան կետերը միմյանց՝ կստանանք պահանջվող կորը: Դիցուք, *x* արգումենտի սկզբնական արժեքը` *D* հատվածի ձախ եզրն է, այսինքն` x_1 արժեքը:

Այժմ պարզենք` թե ինչ գործողություններ են կատարվում արգումենտի *x* կամայական արժեքի դեպքում.

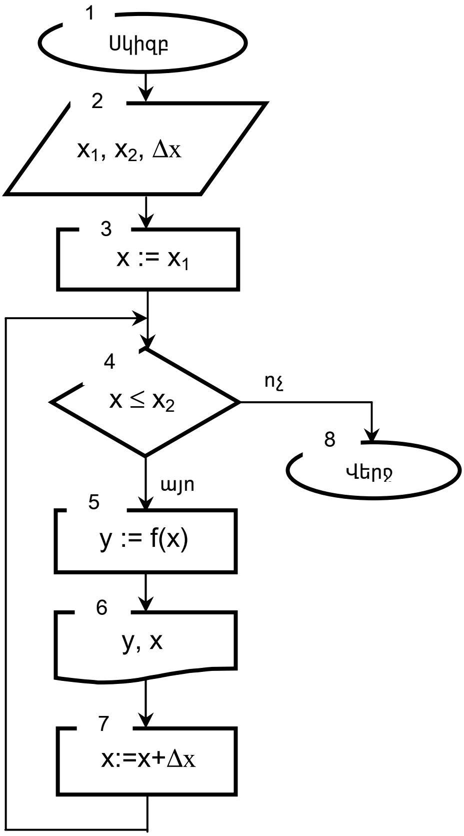
1) տեղադրելով *x*–ի արժեքը *f(x)* արտահայտության մեջ` հաշվում ենք *y* ֆունկցիայի արժեքը ($y:=f(x)$);

2) գրանցում ենք (արտածում ենք) ֆունկցիայի հաշ–վարկված և նրան համապատասխանող արգումենտի ար–ժեքները, որոնք միասին ներկայացնում են մեկ կետի կո–որդինատները;

3) անցնում ենք արգումենտի հաջորդ արժեքին, որը ստացվում է ընթացիկ արժեքից` գումարելով $\Delta x$ քայլը, այսինքն ($x+\Delta x$) նոր արժեքը վերագրում ենք միևնույն` *x* փոփոխականին ($x:=x+\Delta x$). այստեղ է, որ ՛վերագրում՛ գործողությունը ավելի ճիշտ է արտահայտում գործ–ընթացը, քան թե ՛հավասար՛ գործողությունը;

Այսպիսով, եթե *D* տիրույթի սահմանային` $x=x_1$ արժեքի նկատմամբ կիրառենք նշված երեք գործողությունները, ապա կստանանք. ֆունկցիայի $y=f(x_1)$ և արգումենտի $x=x_1+\Delta x$ արժեքները: Եթե կրկին կատարենք նույն գործո–ղությունները, ապա, բնականաբար, կստանանք. ֆունկ–ցիայի $y=f(x_1+\Delta x)$ և արգումենտի հաջորդ` $x=(x_1+\Delta x)+\Delta x$ արժեքները: Ինչպես տեսնում եք, յուրաքանչյուր անգամ ֆունկցիայի հերթական արժեքը հաշվելուց հետո *x*-ի արժեքը աճում է Δx չափով, ապահովելով գործընթացի անընդհատությունը: Այժմ կարող ենք վստահորեն ասել, որ վերը թվարկված երեք գործողություններից կարելի է կազմել **ցիկլ**, որի ավարտը պայմանավորված է *x*-ի և *D* տիրույթի երկրորդ սահմանային կետի` $x_2$–ի հարաբերությամբ: Մեզ մնում է որոշել, թե որ տիպի ցիկլն է ավելի ճիշտ իրականացնում դիտարկված գործընթացը. սկզբնա՞պայմանով, թե` հետպայմանով, կազմել համապատասխան առնչությունը և տեղադրել այն ցիկլի սկզբից կամ էլ վերջից:

Եթե դուք համոզված եք, որ միշտ, $x_1 \le x_2$, ապա վըս–տահորեն կարող եք կիրառել վերջնապայմանով ցիկլ: Սակայն նման խնդիր լուծելիս հնարավոր են այնպիսի վրիպումներ, երբ տվյալները ներմուծելիս դուք ոչ միտումնավոր կերպով ներմուծեք սահմանային կետերի այնպիսի արժեքներ՝ $x_1>x_2$: Տվյալ պարագայում ձեր ցիկլը կհասցնի **մեկ** անգամ իրագործվել (հակառակ մուտքային տվյալների արժեքների), որի արդյունքում կստանանք $y=f(x_1)$ արժեքը, կարտածենք այն և հետո կանցնենք արգումենտի հաջորդ արժեքին, որը x_2–ից ավելի մեծ կլինի: Դրա հետևանքով ցիկլի աշխատանքը կավարտվի: Ալգորիթմը այսպիսի սխալ լուծումներից ապահովագրելու նպատակով տվյալ դեպքում նպատակահարմար է կիրառել **նախապայմանով** ցիկլ:



Նկ.2.3. Խնդիր 11 լուծման ալգորիթմի բլոկ–սխեմա:

Խնդրի լուծման ալգո–րիթմի բլոկ–սխեման բեր–ված է նկ.2.3–ում: Գծա–նկարից և վերը բերված դատողություններից պարզ է, որ այն ցիկլային է և ընդգրկում է 4…7 բլոկներից կազմված ցիկլը: Համաձայն վերը բերված ա) դարձվածքի՝ կազմած ցիկլը կնկարագրվի այսպես. *քանի դեռ $x \le x_2$ կատարել 5, 6 և 7 բլոկները:* Բնական է, որ երբ տվյալ պայմանը խախտվի, այսինքն *x*–ի արժեքը դուրս գա *D* տիրույթի սահմաններից, ցիկլը պետք է ավարտել:

Այժմ փորձեք պատկերացնել, թե ի՞նչ տեղի կունենա, եթե ինչ–ինչ պատճառներով.

1) ցիկլում չնախատեսենք *x* փոփոխականի արժեքի անընդհատ աճ (բլ.7) կամ
2) ցիկլը կազմենք առանց 4–րդ բլոկում նշված պայմանի:

՞...: Այո, դուք իրավացի եք: Առաջին դեպքում կըստանանք **անվերջ** ցիկլ, երբ անընդհատ կհաշվարկվի և կարտածվի ֆունկցիայի միևնույն՝ $y=f(x_1)$ արժեքը: Երկրորդ դեպքում կխախտվի խնդրի հիմնական պայմաններից մեկը, այն է՝ ֆունկցիայի արժեքները հաշվել *D* վերջավոր տիրույթում, ինչը նույնպես կհանգեցնի անընդհատ հաշվարկների:

Եթե նման խնդիր լուծելիս ձեզ կհետաքրքրի իրականացվող ցիկլի կրկնությունների թիվը, ապա դուք այն կարող եք հաշվել հետևյալ բանաձևով.

$$N=\left[\frac{x_2-x_1}{\Delta x}\right]+1, \qquad (2.1)$$

որտեղ միջակ (ուղղանկյուն) փակագծերը նշանակում են՝ նրանցում փակցված արտահայտության ամբողջ մասը: Իսկ իմանալով յուրաքանչյուր ցիկլի կատարման ժամանակաընթացքը, անհրաժեշտության դեպքում կարելի է նաև հաշվել ամբողջ ալգորիթմի կատարման կամ խնդրի լուծման ժամանակը: Ալգորիթմների ժամանակային գնահատականը բազմաթիվ դեպքերում կարող է լինել այս կամ այն ալգորիթմի ընտրության չափանիշ, երբ միևնույն խնդիրը լուծելու նպատակով առաջադրված են լինում մեկից ավելի ալգորիթմներ:

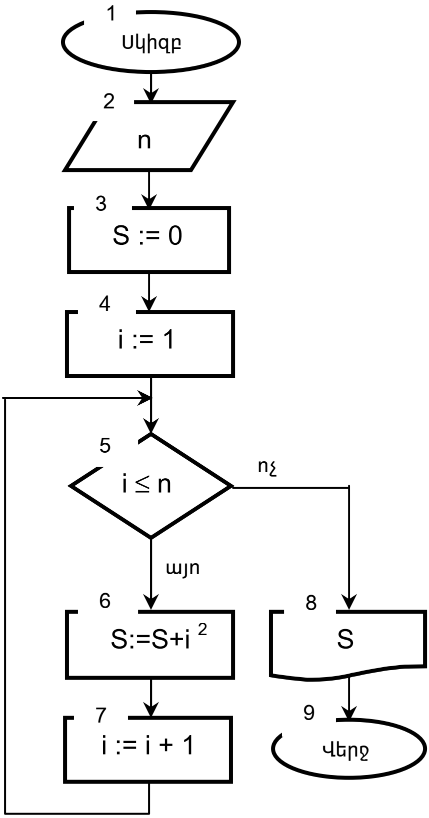
Յաջորդ խնդրում ցիկլի կրկնությունների թիվը նախապես տրվում է, սակայն դա ոչ մի ձևով չի անդրադառնում ալգորիթմի կառուցվածքի վրա. փոխվում է միայն ցիկլի պայմանը:

<u>Խնդիր 12:</u> Տրված **n** բնական թվի համար հաշվել ու արտածել $S=\sum_{i=1}^{n} i^2$ գումարի արժեքը:

Եթե ներկայացնենք գումարը բացված տեսքով, ապա կունենանք հետևյալ բանաձևը.

$$S=1^2+2^2+3^2+\ldots+i^2+\ldots+n^2: \qquad (2.2)$$

Դժվար թե որևէ մեկը բերված արտահայտության ար–ժեքը հաշվելու նպատակով սկսի գումարել վերջից, առա–վել ևս՝ կամայական **i**–րդ անդամից: Սովորաբար գումարի արժեքը հաշվվում է հաջորդաբար՝ գումարելով առաջին անդամին երկրորդը, հետո՝ երրորդը և այսպես մինչև վերջին՝ **n**–րդ անդամը: Իսկ դրա համար **i** փոփոխականը հաջորդաբար ընդունում է **1**–ից **n** ամբողջ արժեքները և յուրաքանչյուր անգամ ըն-թացիկ արժեքի քառակուսին գումարվում է նախորդ քայ-լերից ստացված՝ (**i–1**) ան-դամների մասնակի գու-մարին: Այսպիսով, կարելի է վստահությամբ ասել, որ յուրաքանչյուր **i**–րդ քայլում կատարվում են հետևյալ գործողությունները.



Նկ.2.4. Խնդիր 12 լուծման ալգորիթմի բլոկ–սխեմա:

1) $S:=S+i^2$ – նախորդ քայլերի արդյունքում ստաց-ված մասնակի գումարին ավելանում է i^2 արժեքը և կրկին վերագրվում է **S** փոփոխականին՝ որպես առաջին **i** անդամների մասնակի գումար;

2) **i:=i+1** – հաջորդ անդամին անցնելու նպա-տակով **i** համարին գումարում ենք **1** արժեք:

Այսպիսով, առկա է գոր-ծողությունների կրկնություն, հետևաբար, գործ ունենք շրջափուլային գործընթացի հետ: Քանի որ **i** փոփոխականի արժեքն ան-ընդհատ աճում է և չի գերազանցելու **n** մեծությու-նը, ապա որպես ցիկլի կրկնության անհրաժեշտ պայման ընտրում ենք **i≤n** առնչության ճշմարիտ լինելը:

Անկախ կիրառվող ցիկլի տեսակից անհրաժեշտ է մինչև առաջին ցիկլի իրականացումը որոշել այն փոփոխականների նախնական արժեքները, որոնք կրկնությունից կրկնություն փոփոխվելու են: Մեր օրինակում դա՝ **S** և **i** փո–փոխականներն են: Եթե **i** փոփոխականի առաջին արժեքը ակնհայտ է և հավասար է **1**–ի, ապա գումարի սկզբնական արժեքը կարելի է ստանալ հետևյալ դատողություններից. գումարման՝ $S:=S+i^2$ բանաձևը պետք է գործի **i** փոփո–խականի բոլոր թույլատրելի արժեքների համար, ինչպես նաև՝ **i=1** սկզբնական արժեքի համար: Իսկ առաջին անդամը գումարելուց հետո պետք է ստանանք. $1=S+1^2$ նույնությունը, որտեղից հետևում է, որ **S=0**, այսինքն, գումարի առաջին արժեքը պետք է հավասար լինի զրոյի:

Վերը նկարագրված ալգորիթմի ամբողջական բլոկ–սխեման բերված է նկ.2.4–ում: Այստեղ 5...7 բլոկներից կազմված ցիկլը նույնպես սկզբնապայմանով է և նկա–րագրվում է այսպես. ***քանի դեռ i≤n, կատարել 6 և 7 բլոկները***:

Ինչպես տեսնում եք, այս և նախորդ խնդիրների լուծման ալգորիթմները չնչին, ոչ էական բացառությամբ, նման են միմյանց՝ չնայած պայմանների էական տարբերությանը: Այդ տարբերությունը պայմանավորված է ոչ թե կառուցվածքային որևէ հարցով, այլ ընդամենը՝ արտածման գործողության կիրառմամբ: Նախորդ խնդրում պահանջվում էր արտածել ֆունկցիայի յուրաքանչյուր հաշվարկված արժեք, այդ պատ-ճառով արտածման բլոկը ընդգրկվեց ցիկլի մեջ, իսկ վերջին խնդրում ֆունկցիայի միակ՝ վերջնական արժեքը ստացվում է վերջին կրկնությունից հետո, որի պատճառով արտածման բլոկը տեղադրվեց ցիկլից դուրս:

Յաջորդական գումարումը կարելի է նմանեցնել կու–տակման գործընթացին, երբ դատարկ զամբյուղը լցվում է, հաջորդաբար ավելացնելով որոշակի՝ պարտադիր չէ հա–վասար, քանակությամբ ապրանքատեսակով: Այդ պատճառով “գումարման” ֆունկցիան երբեմն անվանում են “կուտակման” ֆունկցիա և զամբյուղն էլ նախապես պետք է դատարկել (**S=0**):

Բացարձակապես նույն կառուցվածքն ունի “բազմա–պատկման” ֆունկցիայի հաշվման ալգորիթմը, ինչպես, օրի-նակ, հետևյալը.

$$y=\prod_{i=1}^{n} i = 1\times 2\times 3\times\cdots\times n \text{ :} \qquad (2.3)$$

Այստեղ, ի տարբերություն գումարման գործընթացի, յու–րաքանչյուր **i**–րդ քայլում առաջին **(i–1)** բազմապատկե–լիների արտադրյալը բազմապատկվում է **i**–րդ անդամի վրա, ստանալով առաջին i բազմապատկելիների արտադրյալը, այսինքն` y:=y×i: Որպեսզի ցիկլի առաջին կատարման արդյունքում ստանանք 1=y×1 նույնությունը, բնական է ընդունել ֆունկցիայի նախնական արժեքը հավասար y=1: Ի դեպ, 1–ից n բնական թվերի արտադրյալը մաթեմատիկայից հայտնի է որպես “**ֆակտորիալ**” անունով ֆունկցիա և նշանակվում է` y=n!, որի համար ընդունված է. 0!=1, երբ n=0:

Դիտարկենք արտադրյալի հաշվման մեկ այլ օրինակ:

<u>Խնդիր 13:</u> Կազմել i աստիճանացույցի կամայական բնական արժեքի համար **y=2**^{i} աստիճանային ֆունկցիայի արժեքը հաշվող ալգորիթմի բլոկ–սխեմա:

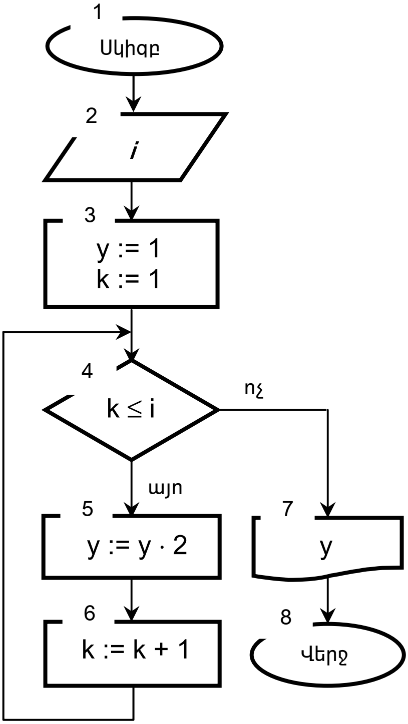
Ինչպես հայտնի է, նշված ֆունկցիայի արժեքը հաշվարկվում է` i անգամ երկուսը բազմապատկելով իր հետ: Հետևելով (2.3) օրինակին, ֆունկցիան կարելի է ներկայացնել արտադրյալի տեսքով.

$$y = 2^i = \underbrace{2 \cdot 2 \cdot 2 \cdots 2}_{i\ \text{հատ}} = \prod_{k=1}^{i} 2 : \qquad (2.4)$$

Վերը նշված դատողությունների հիման վրա աստիճանային ֆունկցիայի արժեքի հաշվման գործընթացը կարելի է նկարագրել նկ.2.5–ում բերված բլոկ–սխեմայով, ըստ որի y և k փոփոխականների նախնական արժեքները որոշելուց հետո (բլ.3) ***քանի դեռ k ≤ i կատարվում են. y–ը 2–ով բազմապատկման և k–ին 1 գումարման գործողությունները*** (բլ.5,6):

Ինչպես երևում է գծանկարից, k փոփոխականը անմիջապես չի մասնակցում ֆունկցիայի հաշվմանը, որն իրականացվում է բլ.5–ում, սակայն այս փոփոխականը անհրաժեշտ է ցիկլի կրկնությունների քանակը հաշվելու համար: Չէ՞ որ մեզ հարկավոր է երկուսով բազմապատկումները կատարել i անգամ: Հետևաբար, տվյալ դեպքում k փոփոխականը կատարում է հաշվիչի դեր: Նման առաքելությամբ օժտված փոփոխականը, ընդահանրապես, կոչվում է **հաշվիչ**:

Հաջորդ օրինակները քննարկելիս յուրաքանչյուր դեպքում դուք կրկին կհամոզվեք, որ ամեն անգամ ցիկլ կառուցելիս կարելի է կիրառել երկու տեսակի ցիկլերից մեկը (նախապայմանով կամ հետպայմանով): Հիմնական բարդությունը երբեմն առաջանում է՝ ցիկլում ֆունկցիան հաշվարկող գործողությունների որոշման հարցում: Եթե խնդիր 11–ում ենթադրվում էր, որ ֆունկցիան տրված է բացահայտ՝ **f(x)** տեսքով, ապա խնդիր 12–ում այն բացահայտ ձևով տրված չէր, և մենք միայն դատողությունների հիման վրա ստացանք. **S:=S+i²** մեկ և **y:=y×i** մյուս դեպքում: Այլ պարագայում ֆունկցիայի արժեքը հաշվարկող արտահայտությունը դուրս բերելու համար կարող են պահանջվել հատուկ մասնագիտական գիտելիքներ և ալգորիթմների կառուցման որոշակի հմտություն, որոնք ունենալու դեպքում հաջողությամբ կկիրառեք ձեզ առաջադրված խնդիրը լուծելու համար:



Նկ.2.5: Խնդիր 13 լուծման ալգորիթմի բլոկ–սխեմա:

Այժմ քննարկենք արդեն դիտարկված խնդիրների այլ, ձևափոխված տարբերակներ, երբեմն միավորելով մեկից ավել խնդիրները մեկի մեջ: Օրինակ, ինչպե՞ս կձևափոխվի խնդիր 12–ի ալգորիթմը, եթե գումարելիների **n** քանակը փոխարինենք այլ պայմանով և խնդիրը ներկայացնենք հետևյալ ձևակերպմամբ:

<u>Խնդիր 14</u>: Տրված է **A** բնական մեծ թիվը: Հաշվել՝ $S = \sum_{i=1} i^2 = 1^2 + 2^2 + \cdots + i^2$, մինչև այն անդամը ներառյալ, որի դեպքում ստացված գումարը կգերազանցի **A** թիվը:

Դժվար չէ կռահել, որ այսպիսի հարցադրմամբ խնդրի լուծումը ստանալու համար բավական է նկ.2.4–ում բերված ալգորիթմի բլոկ–սխեմայում **n** փոփոխականի արժեքի փոխարեն ներմուծել **A** մեծությունը և թիվ 5 բլոկը փոխարինել՝ **S≤A** առնչությունը ստուգող պայմանական բլոկով: Նման պարագայում գումարման գործընթացը կկրկնվի՝ ***քանի դեռ S≤A*** և կավարտվի, երբ **i** փոփոխականի հերթական արժեքի քառակուսին գումարելուց հետո ստանանք՝ **S>A:**

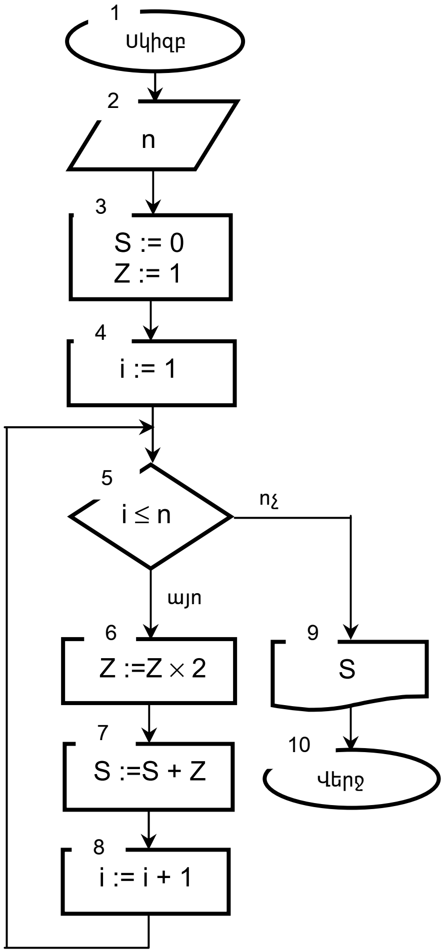
Այժմ ձևափոխենք գումարելի արտահայտության տես–քը, վերցնելով $\mathbf{i^2}$ փոխարեն $\mathbf{2^i}$ և լուծենք հետևյալ խնդիրը:

<u>Խնդիր15:</u> Տրված է **n** բնական թիվը: Պահանջվում է կառուցել հետևյալ գումարի հաշվման և արտածման ալգորիթմի բլոկ–սխեման.

$$S = \sum_{i=1}^{n} 2^{i} : \qquad (2.5)$$

Առաջին հայացքից թվում է, թե ալգորիթմը մնում է անփոփոխ, բացառությամբ ֆունկցիան հաշվող արտա–հայտության, որը կընդունի. $\mathbf{S:=S+2^i}$ տեսքը: Սակայն ստացված արտահայտությունը ճիշտ չի արտացոլում իրականացվող գործընթացը և ահա թե ինչու: Գրելով $\mathbf{2^i}$ մենք ի նկատի ունենք, որ տվյալ պահին անհրաժեշտ է հաշվել երկուսի նշված աստիճանը, որը, ինչպես ասվեց նախորդ խնդրում, հաշվարկվում է՝ **i** անգամ երկուսը բազմապատկելով ինքն իրենով: Սակայն, հաշվի առնելով աստիճանացույցի հաջորդական աճը ցիկլի մի կրկնությունից մյուսը, բնական է թվի **i**–րդ աստիճանը հաշվել օգտվելով իր նախորդ՝ **(i–1)**–րդ աստիճանից, այսինքն. $\mathbf{2^i=2^{i-1}\times 2}$ **ռեկուրենտ** բանաձևից: Եթե այժմ երկուսի աստիճանը նշանակենք, օրինակ, **Z** անունով, ապա նույն ռեկուրենտ բանաձևը կարելի է փոխարինել հետևյալ՝ համարժեք վերագրման գործողությամբ. **Z:=Z×2,** որը ցիկլի **i**–րդ կրկնության սկզբում, ինչպես գիտեք, նշանակում է՝ **Z**–ի հերթական (**i**–րդ) արժեքը ստանալ իր ընթացիկ ((**i–1**)–րդ) արժեքից, այն բազմապատկելով երկուսով: Ինչպես տեսնում եք, անկախ ցիկլի համարից, կամ աստիճանացույցի արժեքից, մենք աստիճան բարձրացման գործողությունը փոխարինեցինք մեկ հատ երկուսով բազմապատկման գործողությունով: Բնական է, որ **Z** փոփոխականի, որպես արտադրյալի, նախնական արժեքը պետք է ընդունվի հավասար մեկի:

Վերը կատարված դատողություններից կարող ենք եզրակացնել, որ (2.5) արտահայտությունը, որպես գումար, պետք է հաշվարկվի համաձայն 2.4 նկարում բերված բլոկ–սխեմայի, այն տարբերությամբ միայն, որ ցիկլի մարմնի աշխատանքը պետք է անմիջապես սկսել **Z**–ի հերթական արժեքի հաշվումով, որից հետո միայն այն կարող ենք գումարել **S**–ին: Խնդրի ամբողջական բլոկ–սխեման բերված է Նկ.2.6–ում:



Նկ.2.6. Խնդիր 15 լուծման ալգորիթմի բլոկ–սխեմա:

Ասածն առավել պարզաբանելու նպատակով կատարենք գծանկարում նըշված բոլոր գործողությունները համաձայն ներկայացված հերթականության: Դիցուք, ներմուծել ենք **n=3** արժեքը (բլ.2): Որից հետո որոշում ենք 3 և 4 բլոկներում նշված նախնական արժեքները ու մտնում ցիկլ: Տեսնենք, թե ի՞նչ է տեղի ունենում ցիկլի երեք կրկնություններից յուրաքանչյուրում:

Ստուգում ենք. $1 \le 3$ առնչությունը (բլ.5): Ստուգման արդյունքն է՝ ՛**այո**՛ պատասխանը: Յետևաբար ցիկլը կատարում ենք առաջին անգամ.

Z:=1×2=2; (բլ.6)
S:=0+2=2; (բլ.7)
i:=1+1=2: (բլ.8)

Ստուգում ենք. $2 \le 3$ առնչությունը (բլ.5): Ստուգման արդյունքն է՝ ՛**այո**՛ պատասխանը: Յետևաբար ցիկլը կատարում ենք երկրորդ անգամ.

Z:=2×2=4; (բլ.6)
S:=2+4=6; (բլ.7)
i:=2+1=3: (բլ.8)

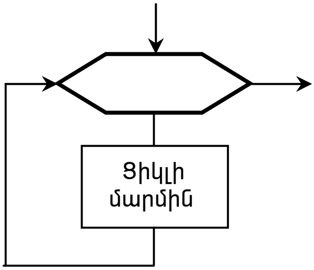
Ստուգում ենք. 3≤3 առնչությունը (բլ.5): Ստուգման արդյունքն է՝ ՛**այո**՛ պատասխանը: Հետևաբար ցիկլը կատարում ենք երրորդ անգամ.

Z:=4×2=8;   (բլ.6)

S:=6+8=14; (բլ.7)

i:=3+1=4:   (բլ.8)

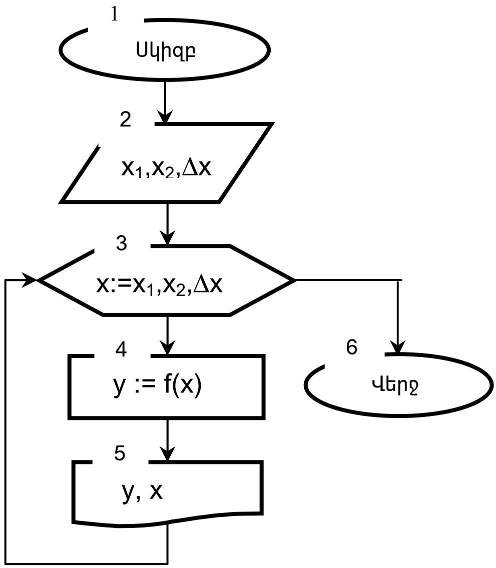
Ստուգում ենք. 4≤3 առնչությունը (բլ.5): Ստուգման արդյունքն է՝ ՛**ոչ**՛ պատասխանը: Հետևաբար անցնում ենք թիվ 9 բլոկին, գրանցում ենք **S** փոփոխականի վերջին արժեքը, որը 14–ն է, և ավարտում աշխատանքը: Ինչպես տեսաք, 6...8 բլոկներից կազմված ցիկլի մարմինը կատարվեց ուղիղ **n=3** անգամ, որովհետև **i** հաշվիչի վերջին՝ 4 արժեքը գերազանցեց **n**–ի արժեքին:



Նկ.2.7: “Մոդիֆիկացիա” բլոկ:

Այսուհետև մենք հաճախակի կհանդիպենք այնպիսի նախապայմանով ցիկլերի, որոնց կրկնությունների քանակը նախապես հայտնի է կամ նախապես կարելի է որոշել, ինչպես նախորդ օրինակներում (բացառությամբ խնդիր 14–ի): Ամեն անգամ բլոկ–սխեմաներում երեք պարտադիր բլոկները չկրկնելու համար, այն է. ցիկլի փոփոխականի նախնական արժեքի որոշումը, փոփոխականի ընթացիկ արժեքի համեմատումը իր վերջին արժեքի հետ և ցիկլում փոփոխականին քայլի գումարումը, հետագայում մենք կկիրառենք ընդհանրացված՝ “**մոդիֆիկացիա**” բլոկը, որով սկսվելու է ցիկլը (նկ. 2.7): Բլոկին կից մուտքային և ելքային գծերը բլոկի հետ հանդերձ առանձին-առանձին մեկնաբանելու փոխարեն նկ.2.8–ում ներկայացված է վերը կազմած 11, իսկ նկ. 2.9–ում՝ 12 խնդիրների լուծման ալգորիթմների ձևափոխված բլոկ–սխեմաները, որտեղ օգտագործած է “մոդիֆիկացիա” բլոկը:

Երկու գծանկարներում էլ “մոդիֆիկացիա” բլոկում նշված են ցիկլի փոփոխականի՝ առաջին, վերջին և քայլի արժեքները, որոնց վրա ոչ մի սահմանափակում չի դրվում: 2.8 բլոկ–սխեմայում ցիկլի վերնագիրը ընթերցվում է այսպես. *փոփոխելով x-ի արժեքը x_1–ից մինչև x_2 Δx քայլով, յուրաքանչյուր արժեքի համար կատարել ցիկլի մարմինը*,



Նկ.2.8: Խնդիր 11 լուծման ալգո–րիթմի բլոկ–սխեմա:

որը տվյալ դեպքում կազմված է 4 և 5 բլոկներից: Քանի որ "մոդիֆիկացիա" բլոկը ենթադրում է նախապայմանով ցիկլ, ապա այստեղ բացառվում է ավելորդ ցիկլի իրականացումը փոփոխման քայլի դրական արժեքի և $x_1>x_2$ դեպքում կամ քայլի բացասական արժեքի և $x_1<x_2$ դեպքում: Իսկ ինչ՞ տեղի կունենա, եթե $x_1<x_2$ պայմանի դեպքում ներմուծենք քայլի արժեքը $\Delta x<0$ կամ $x_1>x_2$ պայմանի դեպքում վերցնենք $\Delta x>0$: Փորձեք պատասխանել առաջադրված հարցերին:

Ցիկլի վերնագրում միշտ չէ, որ պետք է նշել փոփոխման քայլի արժեքը՝ միայն մեկ դեպքում, երբ $\Delta x=+1$, այն կարելի է չնշել, ինչպես նկ.2.9–ում, մնացած դեպքերում քայլի նշումը պարտադիր է:

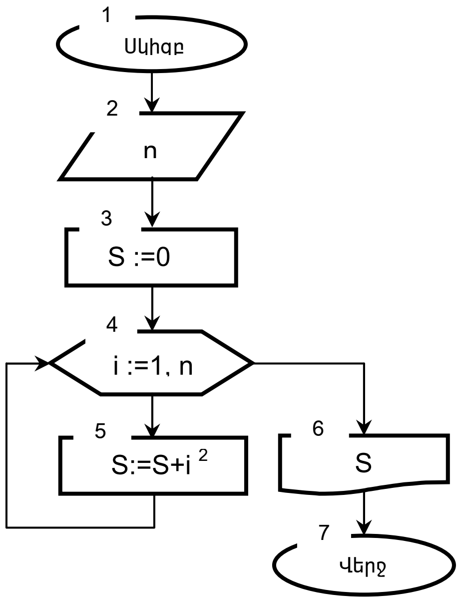
Դիտարկենք պարզագույն ցիկլերի հետ կապված ևս մի շարք խնդիրներ, որտեղ առաջին հայացքից թվացող բազմացիկլային գործընթացը իրականացվում է մեկ ցիկլով: Դա հնարավոր է դառնում, եթե կիրառում ենք, ինչպես խնդիր 15–ում, ռեկուրենտ բանաձևեր:

<u>Խնդիր 16:</u> Տրված **N** բնական թվի համար հաշվել.

$$y = \frac{1}{\sin 1} + \frac{2}{\sin 1 + \sin 2} + \cdots + \frac{N}{\sin 1 + \cdots + \sin N}; \qquad (2.6)$$

Եթե կամենում եք, առաջադրված արտահայտությունը կարելի է ներկայացնել ավելի կարճ տեսքով, որպես.

$$y = \sum_{i=1}^{N} \frac{i}{\sum\limits_{k=1}^{i} \sin k}, \qquad (2.7)$$



Նկ.2.9: Խնդիր 12 լուծման ալգոռիթմի բլոկ–սխեմա:

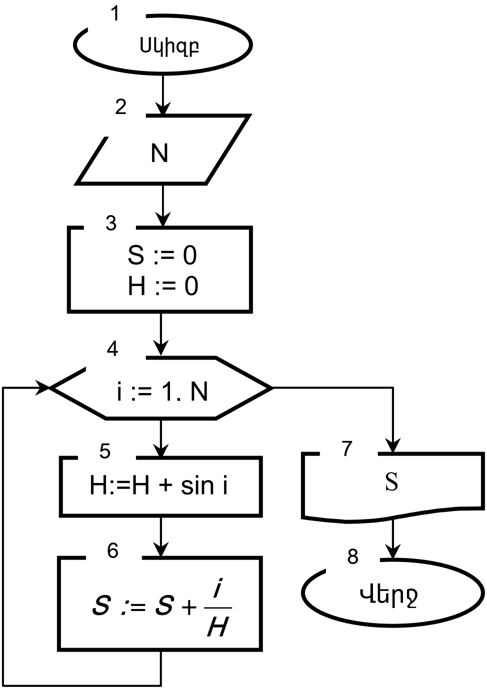
սակայն քիչ հետո կհամոզվեք, որ տվյալ պարագայում նման "պարզեցումը" սկսնակին կարող է միայն խանգարել գործընթացի հաջորդականությունը ըմբռնելու հարցում: Այդ իմաստով ֆունկցիայի առաջին տեսքը նախընտրելի է և ահա թե ինչու: Եթե համեմատեք նախորդ խնդրի հետ, ապա կտեսնեք, որ ի տարբերություն (2.5) արտահայտության, այս դեպքում ոչ թե ընդհանուր անդամն է հաշվարկվում ռեկուրենտ բանաձևով, այլ ընդհանուր անդամի հայտարարը: Այսպես, եթե հայտարարը փոխարինենք **H** անունն ունեցող փոփոխականով, ապա ռեկուրենտությունը ($H_i=H_{i-1}+\sin i$) համարժեք է՝ **H:=H+sini** վերագրման գործողությանը, չէ որ գումարի **i**–րդ անդամի հայտարարը կարելի է ստանալ նախորդ՝ **(i–1)**–րդ անդամի հայտարարից, գումարելով ընդամենը մեկ թիվ. **sin i**: Նկ.2.10–ում բերված է խնդրի լուծման ալգորիթմի բլոկ–սխեման, որից պարզ երևում է, որ 4…6 բլոկներից կազմված ցիկլում, փաստորեն, հաջորդաբար հաշվարկվում են երկու գումարներ՝ **H** և **S**: Այստեղ կարևոր է չխախտել նրանց հաշվման հերթականությունը, որովհետև **H**–ի արժեքը տեղադրվում է **S**–ի արտահայտության մեջ:

Քանի որ ֆունկցիայի արժեքը ստացվում է ցիկլի միայն վերջին՝ **N**-րդ կրկնությունից հետո, արտածման բլոկն էլ տեղադրվում է ցիկլը ավարտելուց հետո (բլ.7): Չի բացառվում, որ նման խնդիրներում հետազոտման նպատակով պահանջվի արտածել գումարի միջանկյալ արժեքները՝ այս դեպքերում արտածման բլոկը անհրաժեշտ է տեղադրել ցիկլի ներսում, բնական է, թիվ 6 բլոկից հետո: Իսկ եթե պահանջվի նաև արտածել հայտարարի միջանկյալ արժեքները, ապա համապատասխան արտածման բլոկը պետք է տեղադրել նույն ցիկլում թիվ 5 բլոկից հետո ցանկացած տեղում:

Խնդիր 17: Տրված N բնական թվի համար հաշվել.

$$y = \underbrace{\sqrt{2+\sqrt{2+\sqrt{2+\cdots\sqrt{2}}}}}_{N\ \text{արմատներ}} \quad : \qquad (2.8)$$

Միայն չշտապեք հայտարարել, որ այս դեպքում էլ գործ ունենք գումարման ֆունկցիայի հետ: Քիչ մտածեք և հետո՝ ծանոթացեք հետագա մեր դատողություններին: Եթե ուշադիր զննեք (2.8) արտահայտությունը, ապա կտեսնեք, որ ֆունկցիայի արժեքը հաշվվում է սկսած ներքին արմատից՝ հաջորդաբար տեղադրելով մեկ արմատի արժեքը մյուսի մեջ: Այսինքն արմատն այստեղ հանդիսանում է ռեկուրսիվ ֆունկցիա և, խոսելով ռեկուրենտ բանաձևերի լեզվով, կարելի է արձանագրել, որ յուրաքանչյուր արմատի արժեքը ստացվում է՝ նախորդ արմատի արժեքին գումարած երկու և կրկին վերցված արմատ՝ գործողությամբ, ինչը արտահայտվում է հետևյալ ձևով. $y := \sqrt{2+y}$: Տվյալ բանաձևից ակնհայտորեն հետևում է. եթե ֆունկցիայի նախնական արժեքը հավասար լինի զրոյի, ապա ցիկլի առաջին իսկ կատարման արդյունքում կստանանք $y = \sqrt{2}$ արժեքը, որը երկրորդ կրկնության ժամանակ տեղադրվում է մյուս արմատի տակ, ստանալով $y = \sqrt{2+y} = \sqrt{2+\sqrt{2}}$ արժեքը և այլն: N անգամ կրկնելով տեղադրման գործողությունը, կստանանք (2.8) արտահայ-



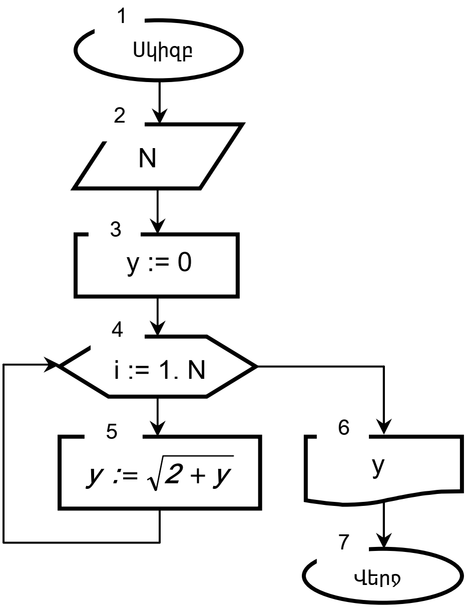
Նկ.2.10: Խնդիր 16 լուծման ալգորիթմի բլոկ–սխեմա:

տության արժեքը: Նկարագրված գործընթացը լավագույնս նկարագրում է նկ.2.11–ում բերված բլոկ–սխեման:

<u>Խնդիր 18</u>: Հաշվել հետևյալ արտահայտության արժե–քը. (2.9)

$$y = \cfrac{1}{1 + \cfrac{1}{3 + \cfrac{1}{5 + \cfrac{1}{\cfrac{\vdots}{101 + \cfrac{1}{103}}}}}}$$

Եթե նախորդ օրինակների քննարկման հետևանքով դուք արդեն տիրապետել եք ռեկուրսիայի ու ցիկլի գաղափարներին, ապա կարելի է ասել, որ դուք հասել եք այնպիսի մակարդակի, երբ ցանկացած նոր խնդիր առաջադրելիս պարտավոր եք սկզբից փորձել լուծել այն ինքնուրույն: Եթե այս անգամ ձեզ մոտ չստացվեց, մի վշտացեք, կփորձեք հաջորդ անգամ: Իսկ եթե անհաջողությունների շարքը երկար տևեց, ապա խորհուրդ կտանք ընդհատել ձեռնարկի հետագա ընթերցումը և կրկին անդրադառնալ նախորդ բացատրություններին: Միայն այսպիսի ստեղծագործական մոտեցմամբ կարելի է հասնել զգալի արդյունքի:

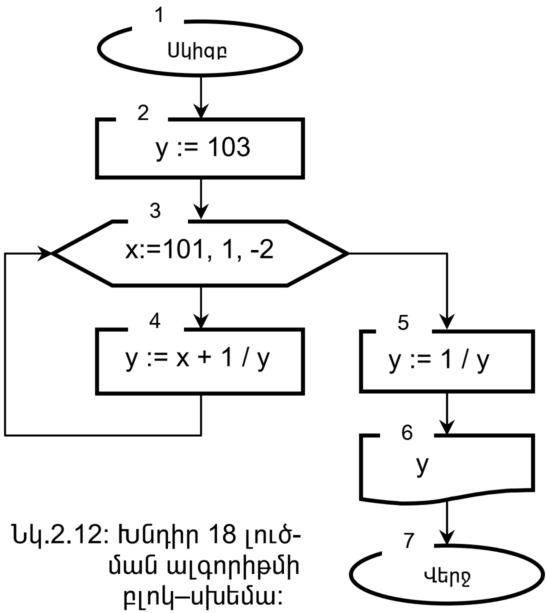


Նկ.2.11: Խնդիր 17 լուծման ալգորիթմի բլոկ–սխեմա:

... Եթե պատրաստ է վերջին խնդրի լուծման ալգորիթմի ձեր տարբերակը, ապա այժմ ծանոթացեք առաջարկվող մեր տարբերակին: Համեմատվող ալգորիթմների միջև տարբերությունը անխուսափելի է, թեկուզ նշանակումների, կամ փոփոխականների նախնական արժեքների ընտրության մեջ: Սակայն ամենակարևոր մասում՝

ռեկուրսիվ բանաձևի և այն իրականացնող ցիկլի ընտրության մեջ մենք պետք է լինենք, հիմնականում, համախոհ:

Վերլուծելով (2.9) արտահայտությունը, եզրահանգում ենք այն մտքին, որ փոփոխականի դերում հանդես է գալիս հայտարարը: Դժվար չէ տեսնել, որ հաշվարկը կատարվում է ներքևից վերև. մեկ հայտարարից նոր հայտարար ստանալով: Հայտարար–փոփոխականը նշանակենք *y* տառով և իր առաջին արժեքը ընդունենք *y=103*: Բացի այդ, հաշվարկներում մասնակցում է ևս մեկ փոփոխական, որի դերում հայտարարում հանդես է գալիս գումարելիներից մեկը: Նշանակենք նրան *x* անունով և փոփոխենք նրա արժեքը *101*–ից մինչև *1* *Δx=–2* քայլով: Գուցե դուք կռահեցիք, թե ո՞րն է հայտարար հաշվող ռեկուրսիվ բանաձևը: Այո, դուք միանգամայն իրավացի եք՝ դա *y=x+1/y* բանաձևն է, որի իրականացումը բերված է նկ.2.12–ում: Ինչպես երևում է գծանկարից,



Նկ.2.12: Խնդիր 18 լուծման ալգորիթմի բլոկ–սխեմա:

ներմուծվող տվյալներ չկան, քանի որ բոլոր փոփոխականների նախնական և վերջնական արժեքները նախապես հայտնի են: Մյուս կողմից՝ 3 և 4 բլոկներից կազմված ցիկլի իրականացման արդյունքում հաշվարկվում է միայն (2.9) արտահայտության հայտարարը, որից հետո վերջնական արդյունքը ստանալու համար կատարվում է բլ.5–ում ներկայացված բաժանման գործողությունը:

<u>Խնդիր 19:</u> Տրված են՝ *x*, *a* իրական և *n* բնական թվերը: Հաշվել հետևյալ արտահայտության արժեքը.

$$\underbrace{(((\cdots\cdots\cdots(((}_{n\ Փակագծեր}x+a)\cdot 10+a)\cdot 10+\cdots+a)\cdot 10+a \qquad (2.10)$$

Մի քանի էջ հետո մենք կզգանք այս խնդրի լուծման ալգորիթմի կարիքը, իսկ առայժմ տվյալ արտահայտության արժեքի հաշվման ալգորիթմի կառուցումը առաջարկում ենք կատարել ինքնուրույն՝ հիմնվելով նախորդ երկու խնդիրների լուծման սկզբունքի վրա: Եթե դուք չկարողանաք, ապա մենք միասի՞ն կվերլուծենք խնդրի ալգորիթմը:

<u>Խնդիր 20</u>: Տրված են դրական, իրական a, x, ε թվերը:

$$y_0 = a; \qquad y_i = \frac{1}{2}\left(y_{i-1} + \frac{x}{y_{i-1}}\right), \quad i = 1,\ 2,\ \ldots\ , \quad (2.11)$$

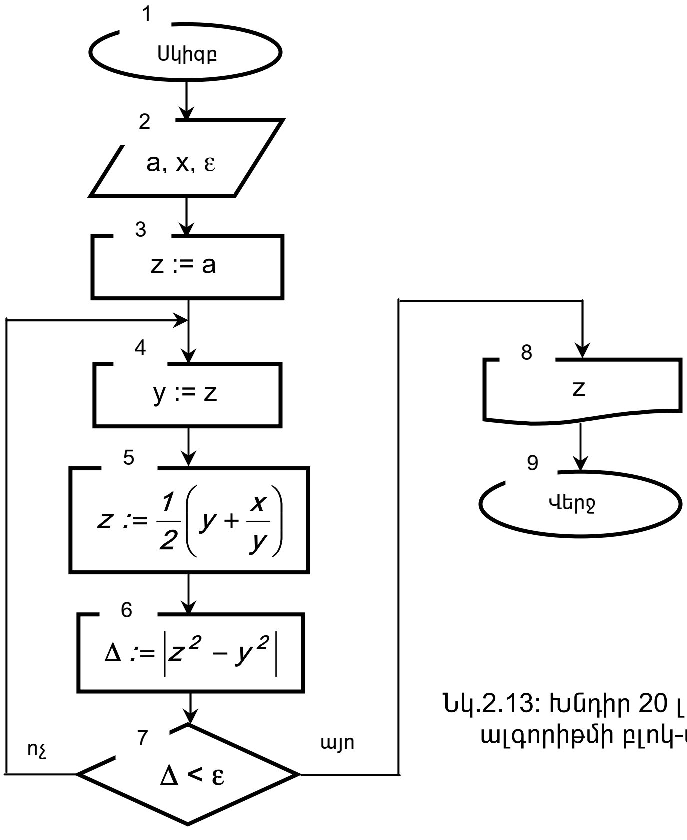
օրենքով առաջացած $y_1,\ y_2, \ldots,$ հաջորդականությունում որոշել այն առաջին y_n անդամը, որը կբավարարի հետև–յալ անհավասարությանը. $\left|y_n^2 - y_{n-1}^2\right| < \varepsilon$: (2.12)

Առաջին հայածքից թվում է, թե նախորդ խնդիրների օրինակով այստեղ նույնպես կարելի է հրաժարվել ինդեքս-ների գործածումից և կիրառել համապատասխան՝ $y := \frac{1}{2}\left(y + \frac{x}{y}\right)$ վերագրման գործողությունը. այնքան ժամա-նակ, մինչև խնդրում բերված (2.12) անհավասարությունը դառնա ճշմարիտ: Սակայն այս և նախորդ պայմանների մեջ գոյություն ունի էական տարբերություն, որը անմիջապես հերքում է մեր նախնական ենթադրությունը: Եթե նախորդ օրինակների յուրաքանչյուր ցիկլում մեզ անհրաժեշտ էր ունենալ ռեկուրսիվ փոփոխականի ընդամենը մեկ ընթացիկ արժեք՝ հաջորդը հաշվելու համար, ապա այստեղ ան-հրաժեշտ են փոփոխականի վերջին երկու արժեքները, ինչը պահանջում է ալգորիթմի մեջ կատարել որոշակի լրացում-ներ:

Մինչև ալգորիթմի կառուցելը կարող ենք հստակ ասել, որ ռեկուրսիան իրականացնող ցիկլը պետք է լինի հետ–պայմանով, քանի որ ըստ խնդրի պայմանի՝ լուծումը ան–հրաժեշտ է ավարտել (2.12) անհավասարության ճշմարիտ լինելուն պես: Իսկ առնչությունը կարող ենք ստուգել միայն ցիկլի վերջում, երբ կունենանք հաջորդականության ընթա-ցիկ երկու արժեքները:

Նկ.2.13–ում բերված խնդրի լուծման ալգորիթմի բլոկ–սխեմայից երևում է, որ հաջորդականության վերջին ար–ժեքը նշանակված է z–ով, իսկ նախավերջինը` y–ով: Քանի որ յուրաքանչյուր ցիկլում վերջին արժեքի միջոցով հաշվարկվում է հաջորդը, այսինքն տվյալ պահի վերջինը հաջորդ պահին դառնում է նախորդ, ապա ցիկլը սկսում է աշխատանքը նրանից, որ y–ին վերագրում է z արժեքը (բլ.4):

Եթե որևէ մեկը կասկածում է, թե (2.11) ռեկուրսիվ բա-



Նկ.2.13: Խնդիր 20 լուծման ալգորիթմի բլոկ-սխեմա:

նաձևի միջոցով ստացվող թվային հաջորդականությունում կգտնվեն երկու հարևան այնպիսի թվեր, որ ցանկացած ε դրական թվի համար կբավարարվի (2.12) անհավասա-րությունը, ապա կարող ենք հավաստիացնել, որ ստացվող թվային հաջորդականությունը զուգամիտում է և, հետևաբար, ձգտում է որոշակի սահմանի: Իսկ (2.12) առնչությունը` սահմանի առկայության հետևանքն է: Նման բանաձևերը, որոնցով հաշվարկվում է ֆունկցիայի մոտավոր արժեքը, ինչպես ասում են, ε ճշտությամբ, այլ կերպ անվանում են “**իտերացիոն**” բանաձևեր: Ավելի մանրամասն իտերացիոն բանաձևերի, ինչպես նաև խնդիր 20–ում ստացվող հաջորդականության զուգամիտության հարցերի հետ կարող եք ծանոթանալ [1] գրականությունից: Նույն այդ գրականությունում կգտնեք այլ տիպի շարքերի զուգամիտության ապացույցը, իսկ այժմ մենք կօգտագործենք այդ փաստը հաջորդ խնդիրը լուծելու համար:

Այս օրինակում մենք առնչվեցինք բլոկ–սխեմաների կառուցման այլ տարբերակի հետ, երբ տարածք տնտեսելու նպատակով երկրորդ սյունը գծագրվում է առաջին սյանը զուգահեռ, սկսելով թիվ 1 բլոկի մակարդակից: Թիվ 7 և 8 բլոկները միացնող սլաքի երեք հատվածով բեկյալի տեսքը բացատրվում է նրանով, որ յուրաքանչյուր բլոկի մուտքային սլաքը պետք է տեղադրվի միմիայն իրենից վերև: Ավելի ընդարձակ բլոկ–սխեմաների դեպքում, երբ բազմաթիվ են պայմանական բլոկների քանակը և նրանցով պայմանավորված անցումները, սլաքների հատումները անխուսափելի են: Նման դեպքերում հատումները թույլ են տրվում՝ միայն հատման կետերը հատուկ կետով նշել հարկավոր չէ:

<u>Խնդիր 21:</u> Տրված են իրական **x** և ε թվերը: Որոշել հետևյալ շարքի գումարը.

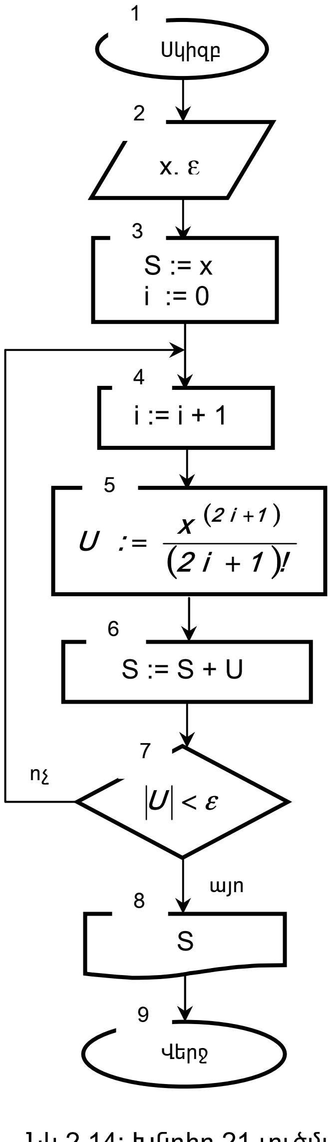
$$S = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \cdots + \frac{x^{(2n+1)}}{(2n+1)!} + \cdots \qquad (2.13)$$

ε ճշտությամբ:

Քանի որ բերված շարքը զուգամիտում է, ապա ցանկացած դրական ε թվի համար կգտնվի այնպիսի **n**–րդ անդամ U_n, որից սկսած բոլոր գումարելիների արժեքները ձգտում են զրոյի, այսինքն.

$$U_i = \frac{x^{(2i+1)}}{(2i+1)!} \to 0, \quad i = n, n+1, n+2, \ldots, \qquad (2.14)$$

իսկ դա նշանակում է, որ *n*–րդ և նրան հաջորդող բոլոր անդամների համար ճիշտ է $|U_n| < \varepsilon$ անհավասարությունը: Հետևաբար, շարքի գումարի հաշվման գործընթացը վեր է ածվում կլասիկ, հետպայմանով ցիկլ պարունակող սխեմայի,



Նկ.2.14: Խնդիր 21 լուծման
Ա21.1 ալգորիթմի բլոկ-սխեմա:

որտեղ որպես ցիկլի ավարտի պայման կիրառվում է վերը նշված առնչությունը, իսկ որպես գումարի նախնական արժեք ընդունվում է առաջին գումարելի՝ *x*–ը:

Ալգորիթմի նկարագրված տարբերակը անվանենք **Ա21.1**, որի բլոկ–սխեման բերված է նկ.2.14–ում:

Սակայն, ծանոթ լինելով ռեկուրսիայի գաղափարի հետ, նշենք, որ շարքի ընդհանուր անդամը կազմված է աստի–ճանային և ֆակտորիալ ֆունկ-ցիաներից, որոնցից յուրա-քանչյուրը հաշվվում է ռե-կուրսիվ բանաձևով, հետե–վաբար, ալգորիթմում կարիք չկա ընդհանուր անդամը հաշ–վել (2.14) բանաձևով: Այն կա-րող ենք ստանալ, առանձին–առանձին հաշվելով արտա-հայտության համարիչն ու հայտարարը և բաժանելով մեկը մյուսին: Հիմնվելով այս դիտո-ղության վրա՝ դիտարկենք տվյալ խնդրի ռեկուրսիվ բանա-ձևերով լուծման երկու տարբերակներ:

1) Ա21.2 ալգորիթմ:

Նախ համարակալենք շարքի անդամները բնական հերթականությամբ, այսինքն՝ հաջորդական համարներով, սկսած մեկից. $U_1=x$; $U_2=\frac{x^3}{3!}$; $U_3=\frac{x^5}{5!}$ և այլն: Հարկավոր է որոշել՝ $U_{i+1}=F(U_i)$ ռեկուրսիվ բանաձևը, այսինքն՝ i–րդ անդամից $(i+1)$–րդ անդամի հաշվարկման անցողիկ բանաձևը: Այդ նպատակով կիրառենք ինդուկցիայի մեթոդը, որն ասում է. եթե որևէ բանաձև ճիշտ է առաջին i քայլերում, ապա այն ճիշտ է նաև $(i+1)$–րդ քայլում: Այս առումով քննարկենք առաջին երկու–երեք քայլերը: Դիցուք, ունենք.

1. $U_1=x$ կամ $U_1=x/1$;

Առաջին անդամից երկրորդը ստանալու համար բավական է համարիչը բազմապատկել x^2-ով իսկ հայտարարը՝ $2\cdot3$ արտադրյալով, այսինքն՝

2. $U_2=U_1\cdot\frac{x^2}{2\cdot3}=\frac{x\cdot x^2}{1\cdot2\cdot3}=\frac{x^3}{3!}$: Եթե շարունակենք նման ոճով, ապա երկրորդ անդամից երրորդը ստանալու համար բավական է համարիչը բազմապատկել կրկին x^2-ով իսկ հայտարարը՝ $4\cdot5$ արտադրյալով, այսինքն՝

3. $U_3=U_2\cdot\frac{x^2}{4\cdot5}=\frac{x^3\cdot x^2}{3!\cdot4\cdot5}=\frac{x^5}{5!}$: Հաջորդ անդամների վերլուծությունը դժվար թե նոր պարզաբանումներ ավելացնի ստեղծված իրավիճակում, քանի որ արդեն կարող ենք կատարել որոշ եզրակացություններ: Դրանք են. ա) ընդհանուր անդամի համարիչը միշտ բազմապատկվում է x^2–ով; բ) հայտարարը բազմապատկվում է երկու այնպիսի թվերով, որոնք կարելի է հաշվարկել անմիջապես ընթացիկ անդամի ինդեքսի արժեքից ելնելով: Այսպես, առաջին բազմապատկելին՝ ինդեքսի կրկնապատիկն է, իսկ երկրորդը՝ կրկնապատիկից մեկով ավելի: Այսպիսով անցողիկ բանաձևը կընդունի հետևյալ տեսքը.

$$U_{i+1}=U_i\cdot\frac{x^2}{2i\cdot(2i+1)},\quad \text{որտեղ } i=1,\ 2,\ 3,\ ... \qquad (2.15)$$

Տեսա՞ք, թե ինչ հաջողությամբ Ա20.1 ալգորիթմի ցիկլի յուրաքանչյուր կրկնողության ընթացքում կատարվելիք՝ աստիճան բարձրացման և ֆակտորիալի հաշվման գործողությունները, անկախ աստիճանացույցի և ֆակտորիալի հիմքի արժեքներից, փոխարինեցինք հետևյալ չորս թվերի բազմապատկումով. *x*–ի քառակուսի աստիճանով (x^2), երկուսը *i*–ով ($2\cdot i$), *2i*–ով և *(2i+1)*–ով: Գործողությունների նման պարզեցումը, այն էլ ցիկլում, ի վերջո հանգեցնում է ժամանակի զգալի տնտեսման, եթե հաշվի առնենք ε ճշտությունը նվաճելու նպատակով՝ իրագործվող ցիկլի կրկնությունների աhռելի քանակը: Ցիկլի յուրաքանչյուր, անգամ պարզագույն, ’ավելորդ’ գործողություն աններելի կերպով երկարաձգում է ալգորիթմի իրականացումը: Այդ իսկ պատճառով նկ.2.15–ում բերված Ա21.2 ալգորիթմի x^2 արժեքը, որպես ’ավելորդ’, հաշվվում է ցիկլից դուրս և վերագրվում **z** փոփոխականին:

Ցանկացած խնդիր լուծելիս մի բավարարվեք ալգո–րիթմի մեկ տարբերակով՝ փորձեք այն բազմակողմանի վերլուծել և փնտրել պարզեցման նոր ուղիներ՝ կիրառելով այլ մեթոդներ կամ հնարքներ, ինչպես, օրինակ, նույն խնդրի Ա21.3 ալգորիթմում:

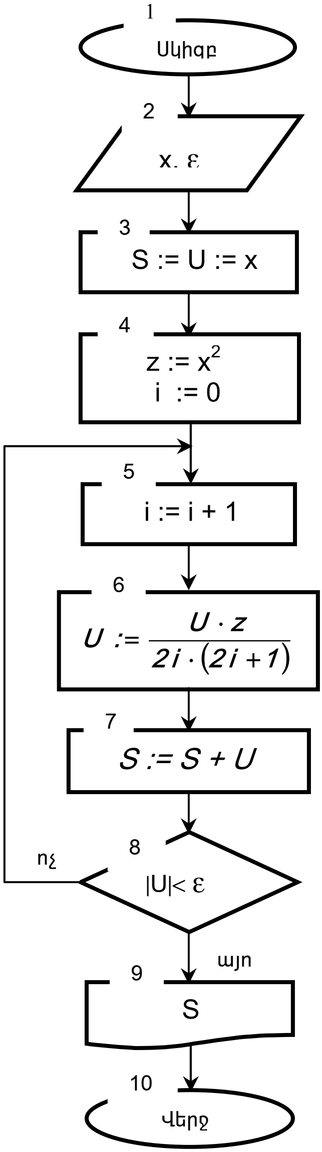
2) <u>Ա21.3 ալգորիթմ</u>

Նկատենք, որ (2.15) անցողիկ բանաձևը ստացանք այն ենթադրությունից, որ շարքի անդամները համարակալված են հաջորդական համարներով, այսինքն՝ **Δi=1** քայլով: Սակայն ոչինչ չի խանգարում համարակալումը իրականացնել այլ քայլով, օրինակ, **Δi=2**: Այդ դեպքում ժամանակի առումով կորուստներ չենք ունենա, մնում է պարզել, թե ինչպես կձևափոխվի անցողիկ բանաձևը:

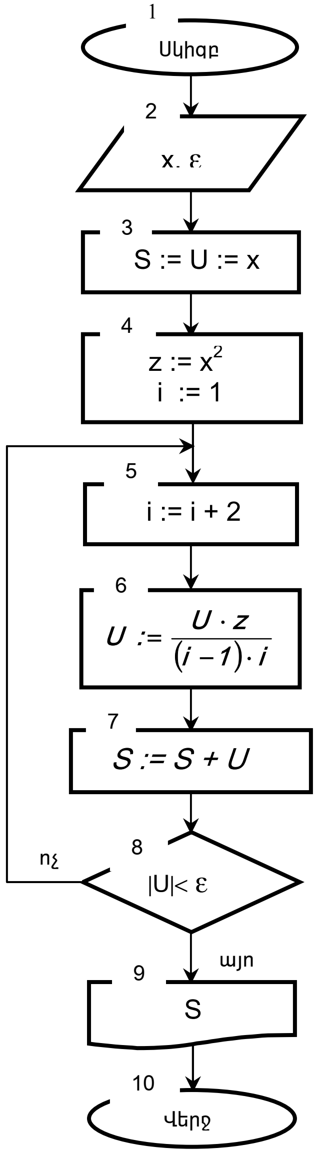
Համարակալենք նույն շարքի անդամները ’2’ քայլով, սկսած մեկից, այսինքն. *1,3,5,...* թվերով: Այս դեպքում կունենանք. $U_1=x$; $U_3=\frac{x^3}{3!}$; $U_5=\frac{x^5}{5!}$ և այլ անդամները: Եթե $U_1=\frac{x}{1}$, ապա հաջորդաբար կստանանք.

$$U_3=U_1\cdot\frac{x^2}{2\cdot 3}=\frac{x\cdot x^2}{1\cdot 2\cdot 3}=\frac{x^3}{3!},\quad U_5=U_3\cdot\frac{x^2}{4\cdot 5}=\frac{x^3\cdot x^2}{3!\cdot 4\cdot 5}=\frac{x^5}{5!}$$ և այլն:

Ինչպես տեսնում ենք, յուրաքանչյուր քայլում. ա) ընդհանուր անդամի համարիչը նույնությամբ բազմապատկվում է x^2–ով; բ) հայտարարը բազմապատկվում է երկու թվերով, որոնցից երկրորդի արժեքը համընկնում է հաշվարկվող (հաջորդ) անդամի ինդեքսի հետ, իսկ առաջինի արժեքը՝



Նկ.2.15: Խնդիր 21 լուծման
Ա21.2 ալգորիթմի բլոկ-սխեմա:



Նկ.2.16: Խնդիր 21 լուծման
Ա21.3 ալգորիթմի բլոկ-սխեմա:

հաջորդ անդամի ինդեքսից մեկով պակաս է: Այսպիսով, անցողիկ բանաձևի պարզագույն տեսքը կլինի.

$$U_i = U_{i-2} \cdot \frac{x^2}{(i-1)\cdot i}\ , \quad \text{որտեղ } i = 3,\ 5,\ 7,\ \ldots \qquad (2.16)$$

Ահա այսպիսի պարզեցված ռեկուրենտ բանաձևով շարքի ընդհանուր անդամ հաշվարկող ցիկլով Ա21.3 ալգորիթմի բլոկ–սխեման բերված է նկ.2.16–ում: Ճիշտ է, մենք շահեցինք ընդամենը մեկ բազմապատկման գործողություն` *2·i*, այն էլ մեկ ցիկլում: Սակայն ցիկլի բազմաթիվ կրկնությունների պարագայում ժամանակի տնտեսումը դառնում է ակնհայտ:

Այս մեկ օրինակով մենք ցանկանում էինք ընդարձակ ձևով ներկայացնել յուրաքանչյուր խնդրի լուծման ալգո–րիթմի կառուցման գործընթացը: Եթե ամփոփենք ար–դյունքները, ապա կարող ենք եզրակացնել, որ ալգորիթմի կլասիկ տարբերակը ընտրվում է միայն որպես հիմք ամբողջ շինության (ալգորիթմի) կառուցման գործում, ինչպես գումարման սխեման վերջին խնդրում: Դրանից հետո սկսվում է հիմնական ստեղծագործական աշխատանքը, երբ պահանջվում են ձեր մաթեմատիկական և տրամաբանական ողջ ունակությունների ներդրումը ընտրված սխեմայի կատարելագործման հարցում: Դրա ցայտուն օրինակն էր` ռեկուրսիայի կիրառումը շարքի ընդհանուր անդամի հաշվման գործում: Իսկ վերջնական պարզեցված տարբերակը շտկըվում է որոշ հնարքներ կիրառելուց հետո. եթե հիշում եք, վերջին օրինակում բավական էր փոփոխել շարքի անդամների համարակալման քայլի չափը, որպեսզի ցիկլի յուրաքանչյուր կրկնության ընթացքում շահեինք մեկ բազմապատկման գործողություն:

Հետագայում նման խնդրի հանդիպելիս, բնական է, որ դուք, հիմնվելով ձեռք բերած փորձի վրա, որպես ալ–գորիթմի հիմնային սխեմա ընտրեք Ա21.3 ալգորիթմի սկզբունքը: Ինքնավարժանքի նպատակով կառաջարկենք լուծել սույն բաժնի վերջում ներկայացված մի շարք խնդիրներ:

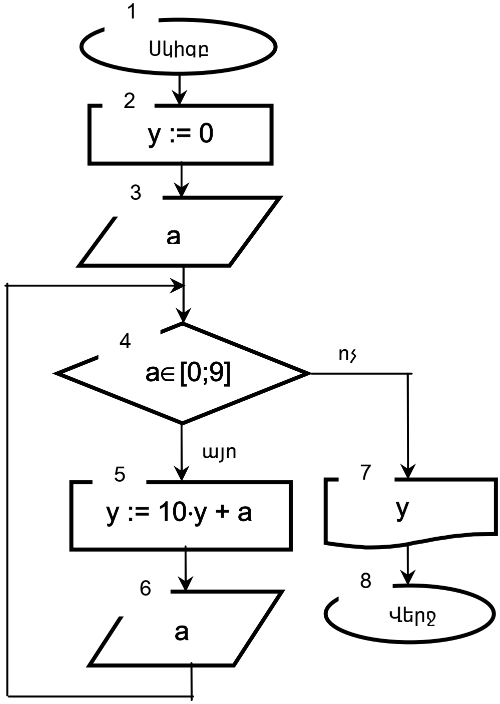
Իսկ այժմ դիտարկենք այլ տիպի խնդիրների լուծման եղանակները, որոնք հաճախակի են կիրառվում ծրագրա–վորման պրակտիկայում: Մինչ բուն խնդրին անցնելը բերենք որոշ պարզաբանումներ:

Գիտե՞ք, արդյոք, թե ինչպես է առանձին թվանշաննե–րից ձևավորվում բնական թիվը: Ինչպե՞ս է, օրինակ, ’5’ (հինգ) թվանշանից ստացվում ’51’ (հիսունմեկ) թիվը, երբ հինգի կողքին ավելացնում ենք ’1’ թվանշանը: Եվս մեկ թվանշան, օրինակ, ’3’ (երեք) ավելացնելով, կստանանք՝ ’513’ (հինգ հարյուր տասներեք) թիվը և այլն: Այս գործընթացի նկարագրման հարցում մեզ կարող է օգնել խնդիր 19–ում առաջադրված (2.10) արտահայտության հաշվման ալգորիթմը, որը ձեզ վաղօրոք հանձնարարվել էր կազմելու: Եթե կազմել եք, ապա այս ցուցումը բավական է, որպեսզի դուք կռահեք բնական թվերի ձևավորման եղանակը, իսկ եթե ոչ, ապա առաջարկում ենք միասին լուծել հետևյալ ձևակերպմամբ խնդիրը.

<u>Խնդիր 22:</u> Կազմել թվանշանների հաջորդականությունից բնական թվի ձևավորման ալգորիթմը:

Անդրադառնալով (2.10) արտահայտությանը, կարող ենք արձանագրել, որ նրա արժեքը հաշվարկվում է ցիկլի կրկնությունների արդյունքում՝ մեկ փակագծի արժեքից մյուսը ստանալով, ինչի համար մեկ փակագծի արժեքը բազմապատկվում է տասով և արտադրյալին գումարվում **a** թիվը: Նկատենք, որ նույն սկզբունքը գործում է նաև վերը բերված թվային օրինակում: Այստեղ փակագծի դերում հանդես է գալիս մինչև ընթացիկ պահը ներմուծված թվանշաններից կազմված թիվը, իսկ գումարվող **a** թվի դերում՝ հերթական թըվանշանը: Դիցուք, ’51’ թիվը ստացվում է ’5’–ը բազմապատկելով ’10’-ով և գումարելով ’1’ թվանշանը, իսկ ’513’ թիվը՝ արդեն ստացած ’51’–ը բազմապատկելով ’10’–ով և գումարելով ’3’ թվանշանը: Այսպես պետք է շարունակել մինչև ավարտվի թըվանշանների ցանկը: Իսկ սկզբում կարելի է ենթադրել, որ թվի արժեքը զրո է, որից առաջին իսկ քայլում ստացվում է ’5’ թիվը. $0 \cdot 10+5=5$:

Նկ.2.17–ում բերված է խնդիր 22-ի լուծման ալգորիթմի բլոկ–սխեման, որտեղ որպես *y* թվի նախնական արժեք (առաջին փակագիծ) ընտրված է զրոն, իսկ յուրաքանչյուր գումարվող թվանշանի արժեքը (**a**–ն), բացառությամբ առաջինի, ներմուծվում է ցիկլի հերթական կրկնության ընթացքում: Ինչպես երևում է գծանկարից, հաշվարկող ցիկլը նախապայմանով է, այնպես որ այն հերթական անգամ կկատարվի, եթե մուտքային նշանը թվանշան է, այսինքն **a**–ի արժեքը պատկանում է **[0;9]** բազմությանը: Այս հանգամանքը բլոկ–սխեմայում արտահայտված է թիվ 4 պայմանական բլոկում: Իսկ թե ինչպես է այդ փաստը ստուգվում, հուսով ենք՝ չեք մոռացել. **a∈[0;9]** պայմանը ճըշմարիտ է, եթե միաժամանակ **a≥0** և **a≤9**: Մնացածը մեկնաբանվում է սովորական ձևով. ***քանի դեռ a∈[0;9] կատարել 5 և 6 բլոկներում***



Նկ.2.17: Խնդիր 22 լուծման ալգորիթմի բլոկ–սխեմա:

նշված գործողությունները, հակառակ դեպքում թվի հաշվումը կավարտվի:

Յետագայում դուք կրկին կհանդիպեք (2.10) տիպի արտահայտության և մեկ անգամ ևս կհամոզվեք նրա պրակտիկ մեծ նշանակության մեջ: Ի դեպ, դուք արդեն պատրաստ եք տվյալ հանդիպմանը և, հուսով ենք, ի վիճակի եք ինքնուրույն կառուցելու նման, թեկուզ և ընդհանրացված ալգորիթմ:

Դիտարկված խնդիրն ունի իր հակադարձը, երբ անհրաժեշտ է որոշել տրված բնական թիվը կազմող թվանշանները, կամ թվի կարգայնությունը: Առաջին հայացքից առաջադրված խնդիրը անիմաստ է, քանի որ, եթե թիվը գրված է և, հետևաբար, տեսանելի է, ապա թվանշանների կազմը, ինչպես նաև նրանց քանակը, ակնհայտ են: Իսկ եթե թիվը տեսանելի չէ՞, կարո՞ղ եք, արդյոք, թվարկել այն գործողությունները, որոնք պետք է կատարվեն բաղադրիչ թվանշանները, այն է` միավորները, տասնավորները և այլն, որոշելու համար: Այսինքն առաջարկում ենք լուծել հետևյալ խնդիրը:

<u>Խնդիր 23:</u> Տրված է բնական **N** թիվը: Որոշել այդ թվի պատկերը կազմող թվանշանները:

Տրված թիվը տասնորդական հաշվարկային համակար–գից է, որը պատկանում է դիրքային համակարգերի դասին: Իսկ այդ տիպի համակարգերում, ինչպես հայտնի է, ցանկացած թվի պատկերը կազմող թվանշանների իրական արժեքը կախված է թվանշանի զբաղեցրած դիրքից: Դի–ցուք, մեր թիվը (p+1) կարգանի է, հետևաբար նրա պատ–կերը տասնորդական թվանշանների վերջավոր հաջորդականություն է՝ $a_p a_{p-1} \cdots a_2 a_1 a_0$, որտեղ a_p, a_{p-1}, … , a_1, a_0 թվանշաններից յուրաքանչյուրը պատկանում է **[0;9]** բազմությանը, իսկ նրանց քաշերը ավելանում են 10 անգամ աջից ձախ: Օրինակ, վերը նշված '513' թիվը երեք կարգանի է (p=2) և նրա պատկերը կազմված է ´**5**´, ´**1**´ և ´**3**´ տասական թվանշաններից, որտեղ $a_0=3$, $a_1=1$, $a_2=5$: Սակայն, եթե a_0–ի քաշը հավասար է մեկի, ապա a_1-ի քաշը հավասար է 10, իսկ a_2–ինը՝ 100 և այլն: Հաշվի առնելով այս հանգամանքը, 513 թվի արժեքը հաշվում են, որպես $(5{\cdot}10^2+1{\cdot}10^1+3)$ բազմանդամի արժեք: Ընդհանուր դեպքում բնական թվի արժեքը կարելի է հաշվել իր բաղադրիչ թվանշաններից հետևյալ բազմանդամի միջոցով.

$$N = a_p \cdot 10^p + a_{p-1} \cdot 10^{p-1} + \cdots + a_2 \cdot 10^2 + a_1 \cdot 10^1 + a_0 : \quad (2.17)$$

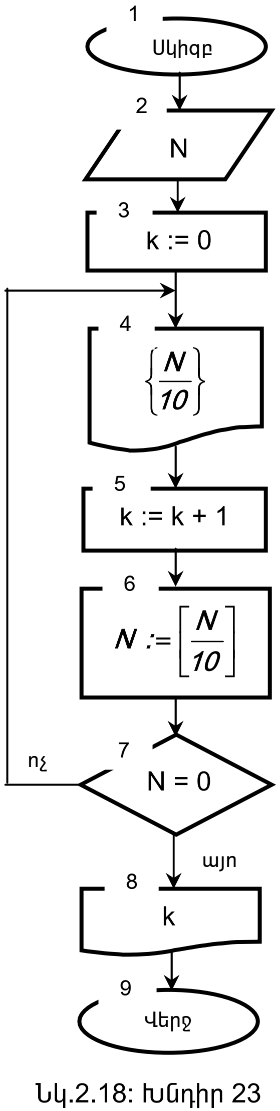
Մեր խնդիրն է՝ ունենալով բնական թիվ, որոշել նրա բաղադրիչ թվանշանները: Այդ նպատակով դիմենք (2.17) արտահայտությանը որտեղից երևում է, որ բոլոր գումարելիները, բացառությամբ վերջինի՝ a_0–ն, անմնացորդ բաժանվում են 10-ի: Այստեղից հետևում է, որ եթե **N** թիվը բաժանենք 10-ի, ապա առաջացած մնացորդը հավասար կլինի a_0–ի: Այս հանգամանքը կարող ենք գրանցել այսպես. $a_0 = \left\{\frac{N}{10}\right\}$: Բաժանման արդյունքի՝ քանորդի ամբողջ մասը, բնականաբար, a_0 չի պարունակի, իսկ մնացած a_i (i=1,2,…,p) թվանշաններին կից տասի աստիճանացույցները մեկով կնվազեն: Եթե ամբողջ մասը նշանակենք $\mathbf{N_1}$, ապա կունենանք.

$$N_1 = a_p \cdot 10^{p-1} + a_{p-1} \cdot 10^{p-2} + \cdots + a_2 \cdot 10^1 + a_1 :$$

Եթե շարունակենք նույն ոճով, ապա կարող ենք ասել, որ բաժանելով $\mathbf{N_1}$–ը տասի կստանանք a_1 մնացորդը, իսկ $\mathbf{N_2}$ ամբողջ մասը կընդունի հետևյալ տեսքը.

$$N_2 = a_p \cdot 10^{p-2} + a_{p-1} \cdot 10^{p-3} + \cdots + a_3 \cdot 10^1 + a_2 :$$

Այսպես շարունակելով, կհասնենք $N_p = a_p$ թվին: Մեկ անգամ ևս բաժանելով ընթացիկ N_p թիվը տասի, որպես մնացորդ, կստանանք a_p թվանշանը, իսկ ամբողջ մասը կհավասարվի զրոյի՝ դրանով թվանշանների որոշման գործընթացը կավարտվի:



Նկ.2.18: Խնդիր 23 լուծման ալգորիթմի բլոկ–սխեմա:

Ինչպես տեսանք, խնդրի լուծումը ավարտվում է, երբ հերթական N_i բնական թիվը, որը ստացվել է նախորդ քայլի N_{i-1} բնական թվից, բաժանելով այն տասի վրա և վերցնելով քանորդի ամբողջ մասը, այսինքն՝ $N_i = \left[\frac{N_{i-1}}{10}\right]$, հավասարվում է զրոյի: Այս հանգամանքից ելնելով, վերը ներկայացված գործընթացը վստահորեն կարող ենք նկարագրել հետպայմանով ցիկլի միջոցով, ինչը բերված է նկ.2.18–ում: Խնդիրը լուծելիս, զուգահեռաբար կարող ենք հաշվել նաև այլ պարամետրերի արժեքներ՝ դիցուք ալգորիթմի բերված տարբերակում մենք միաժամանակ հաշվում ենք **N** թվի k կարգը, որի արժեքը արտածվում է ցիկլի աշխատանքը ավարտելուց հետո: Իսկ ցիկլում ընդգրկված են՝ կրտսեր թվանշանի որոշման և արտածման (բլ.4), կարգը որոշող k փոփոխականին մեկ գումարման (բլ.5) և **N**–ը տասի բաժանած քանորդի ամբողջ մասով փոխարինող (բլ.6) բլոկներով:

Կառուցված ցիկլի նկարագրու–թյունը նույնպես ոչնչով չի տարբերվում նախորդ օրինակներում կիրառված դարձվածքներից և հնչում է այսպես. ***կրկնել 4, 5 և 6 բլոկները մինչև N=0 պայմանի ճիշտ լինելը:*** Տեսնու՞մ եք, արդեն որերորդ անգամ նույն ոճով նկարագրում ենք տարբեր բովանդակությամբ ցիկլեր: Տարբերությունը միայն ցիկլի պարունակությունն է:

Ինչպես երևում է ալգորիթմի բլոկ–

սխեմայից, ցիկլի յուրաքանչյուր կրկնության ընթացքում հաշվարկվող a_i (i=1,2,...,p) թվանշանը միայն արտածվում է և ոչ մի տեղ չի պահպանվում: Այնպես որ, եթե որևէ խնդրում հարկ լինի ինչ–որ ձևով դրանց օգտագործել, ապա անհրաժեշտ է ցիկլի սկզբում ավելացնել ընդամենը մեկ վերագրման գործողություն՝ $a_k := \left\{ N/_{10} \right\}$: Այս դեպքում ցիկլի առաջին կատարման ժամանակ, երբ k=0, հաշվարկվող մնացորդը կվերագրվի a_0–ին, կարտածվի և k–ի արժեքը կաճի մեկով, ստանալով մեկ արժեքը: Ցիկլի հաջորդ կրկնության ժամանակ մնացորդը կվերագրվի a_1–ին, կարտածվի և k–ն կընդունի երկու արժեքը և այլն: Այսպիսով մենք կստանանք $A(a_0, a_1, \cdots, a_{k-1}, a_k)$ վեկտոր, որի տարրերի արժեքներն են՝ **N** թվի միավորների, տասնավորների, հարյուրավորների և այլն քանակները, այսինքն՝ բաղադրիչ թվանշանները: Ցիկլի աշխատանքի ավարտից հետո k–ն ցույց կտա ոչ միայն **N** թվի կարգը, այլ նաև **A** վեկտորի երկարությունը:

Յաջորդ խնդիրը լուծելիս առաջին հայացքից կարող է թվալ, որ բոլոր գործողությունները կարելի է իրականաց–նել մեկ ցիկլի կրկնությունների շնորհիվ: Սակայն դա այդպես չէ:

<u>Խնդիր 24:</u> Տրված է **N** բնական թիվը: Եթե այն կազ–մող բոլոր թվանշանները փոքր են ’5’–ից, ապա ստանալ $\mathbf{N_1}$ բնական թիվը, կրկնապատկելով **N** թվի թվանշանները և պահպանելով թվանշանների հարաբերական դիրքը: Եթե **N** թվի պատկերում առկա են ’5’–ից մեծ կամ հավասար թվանշաններ, ապա թիվը թողնել անփոփոխ:

Մինչև խնդրի լուծմանն անցնելը հստակեցնենք խնդրի պայմանները, արդյունքը և վերջինիս հասնելու հնարավոր ուղիները: Դիցուք, N=3014 թվի բոլոր a_i թվանշանները փոքր են ’5’ թվանշանից, ինչի պատճառով, կրկնապատ–կելով յուրաքանչյուր թվանշան, պետք է ձևավորել N_1=6028 թիվը: N=92518 թվի կազմում առկա են ինչպես ’5’–ից փոքր, այնպես էլ՝ մեծ կամ հավասար թվանշաններ, հետևաբար, թիվը պետք է թողնել անփոփոխ:

Ցանկացած խնդիր կարելի է լուծել տարբեր եղանակ–ներով, և այս խնդիրը նույնպես բացառություն չի կազմում: Սակայն եթե ալգորիթմի նկատմամբ ոչ մի հատուկ պահանջ չի ներկայացված, ապա նպատակահարմար է կիրառել ծեզ հայտնի մեթոդները, ինչը կբերի ժամանակի զգալի տնտես-մանը: Նման մոտեցմամբ ալգորիթմը կարծես թե ’հավաքվում է՝ առանձին մոդուլներից, արդյունքում պարզեցնելով ալգորիթմի կառուցման գործընթացը: Ասածը ցուցադրենք վերջին օրինակով:

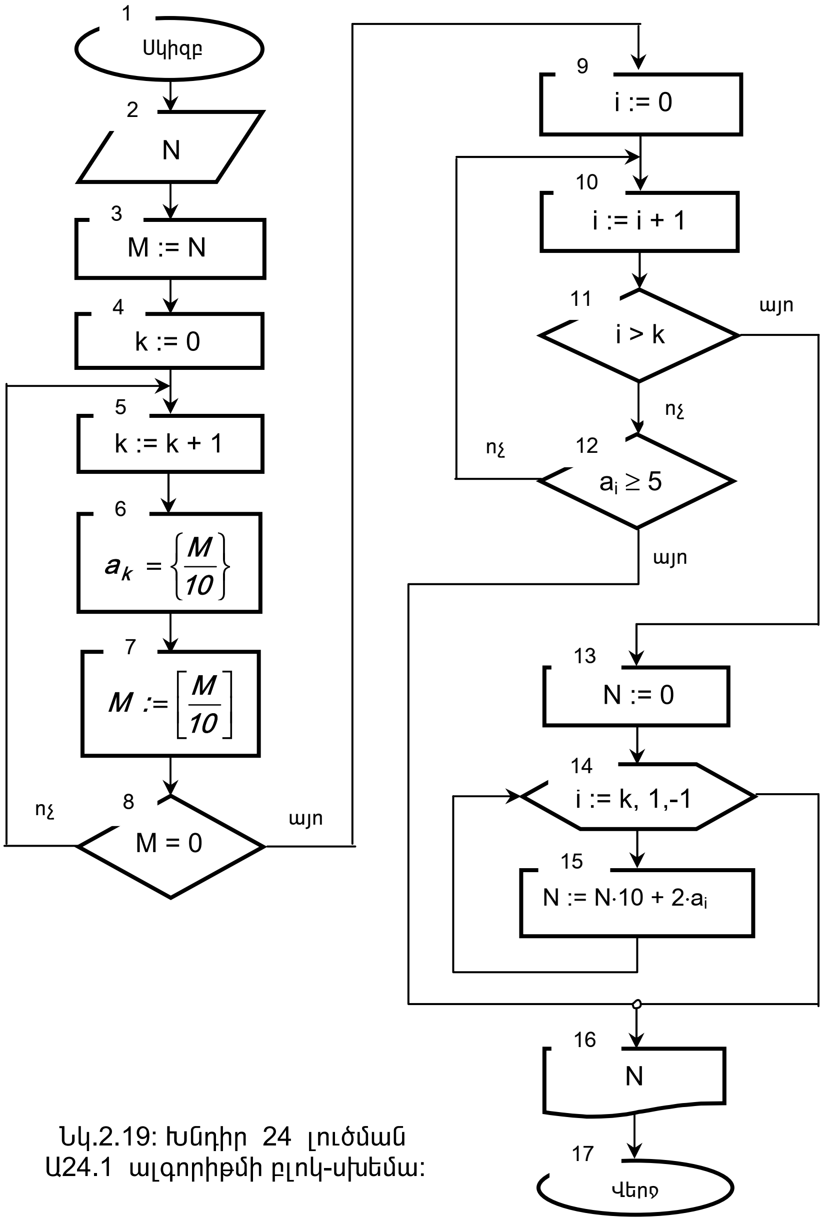
Նոր թիվը մենք կարող ենք ձևավորել խնդիր 22–ում ներկայացված ալգորիթմով (**Ա22**), որտեղ պահանջվում է թվանշանները ներմուծել՝ սկսած ավագ կարգից: Սակայն այդ թվանշանները **Ա23** ալգորիթմի միջոցով ՚արտաքըս–վելու՚ են տրված **N** թվից հակառակ հերթականությամբ՝ սկսած միավորներից: Յետևաբար, այստեղ պետք է դիմենք վերը առաջարկված **A** զանգվածի օգնությանը՝ առաջին փուլում հաջորդաբար գրանցելով այնտեղ բոլոր $\mathbf{a}_i$ բաղադրիչ թվանշանները:

Խնդրում առաջադրված հարցի պատասխանը ստանա–լու ենք երկրորդ փուլում: Դրա համար ամենևին էլ պար–տադիր չէ ստուգել բոլոր թվանշանները՝ բավական է դրանք ստուգել հաջորդաբար՝ **մինչև** այն պահը, երբ կհանդիպենք հարակից այնպիսի երկու թվանշանների, որոնք գտնվում են ՚4՚–՚5՚ սահմանագծի երկու կողմերում: Եթե այդպիսի իրավիճակի չհանդիպենք, ապա N թիվը կթողնենք անփոփոխ, հակառակ դեպքում՝ երրորդ փուլում կձևավորենք նոր թիվ:

Նկ.2.19–ում բերված է խնդիր 24–ի լուծման վերը նշված տարբերակի՝ Ա24.1 ալգորիթմի բլոկ–սխեման, որը տրամա-բանորեն բաղկացած է երեք մասերից (մոդուլներից)՝ համա-պատասխան նշված փուլերի, որոնցից յուրաքանչյուրը իրականացվում է իրեն բնորոշ ցիկլի միջոցով: Այժմ փուլերի մասին ավելի մանրամասն:

Ինչպես երևում է գծանկարում, մուտքային N փոփո–խականի արժեքը կրկնօրինակված է M փոփոխականում (բլ.3): Դա կատարել ենք, հաշվի առնելով այն հանգամանքը, որ առաջին փուլը իրականացնող Ա23 ալգորիթմի իրագործման արդյունքում (բլ.5...8) մուտքային թիվը չի պահպանվում, իսկ համապատասխան պայմանի դեպքում այն կարող է վերջում պետք գալ: Մյուս կողմից, խնդրի պայմաններից ելնելով, առաջին փուլում Ա23 ալգորիթմի համեմատ կատարել ենք աննշան փոփոխություն. մուտքային թվի յուրաքանչյուր թվանշան պահպանվում է **A** վեկտորի համապատասխան դիրքում (բլ.5,6):

Երկրորդ փուլը իրականացվում է 9...12 բլոկների մի-ջոցով, որտեղ 10...12 բլոկներից կազմված հետպայմանով ցիկլում որոշվում է բաղադրիչ a_i թվանշանների պատ-կանելության տիրույթը: Ինչպես տեսնում եք, ցիկլը կարող է ընդհատվել երկու դեպքում. մեկը բնական ձևով, երբ **A** վեկ-



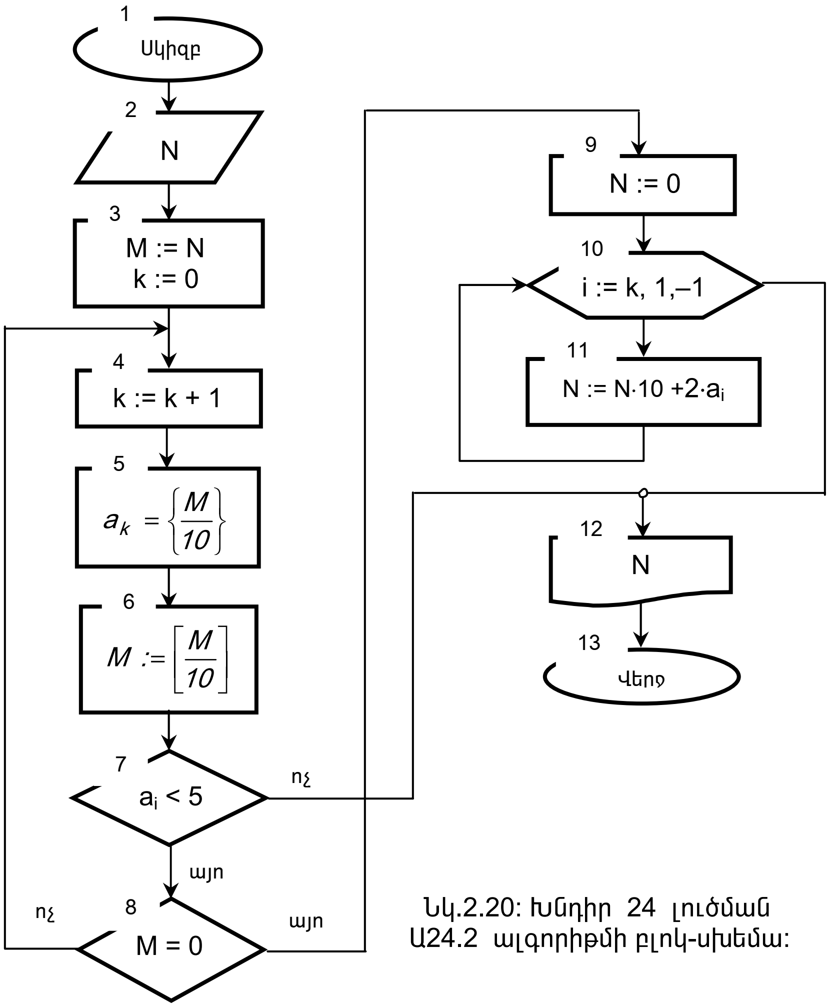
Նկ.2.19: Խնդիր 24 լուծման Ա24.1 ալգորիթմի բլոկ-սխեմա:

տորի բոլոր տարրերը փոքր են ‘5’–ից՝ այս դեպքին համա-պատասխանում է **i>k** պայմանը (բլ.11); մյուս պատճառն է՝ ’5’–ից մեծ կամ հավասար թվանշանի հանդիպելը (բլ.12), որի դեպքում պահանջվում է մուտքային թիվը թողնել անփոփոխ. ահա՜ որտեղ մեզ օգնեց **N** թվի **M** կրկնօրինակը: Քանի որ ցիկլի ավարտի պայմանները երկուսն են, ապա այն կնկարագրվի հետևյալ դարձվածքով. ***կրկնել թիվ 10 բլոկը մինչև (i>k) կամ (a_i ≥5)*** պայմաններից մեկի ճիշտ լինելը: Այս դեպքում, բնական է, որ ցիկլը կշարունակի կատարվել, ***քանի դեռ (i≤k) և (a_i<5)*** պայմանները միաժամանակ առկա են: Եթե ցիկլը ավարտվում է բնական ձևով, դա նշանակում է, որ բոլոր թըվանշանները փոքր են հինգից և ինչպես գծանկարում է ըստ խնդրի պահանջի, կատարվում է անցում երրորդ փուլին՝ նոր թիվ ձևավորելու նպատակով (բլ. 13…15):

Ինչպես վերը նշվեց, այս փուլում կիրառել ենք Ա22 ալգորիթմը, որը **N** թվի բաղադրիչ թվանշանները վերցնում է **A** վեկտորի վերջից՝ $a_k, a_{k-1}, \dots, a_2, a_1$ հերթականությամբ: Այստեղ որպես նոր թվի անուն վերցված է կրկին **N**–ը, ինչը թույլ է տալիս ձևավորված թիվը արտածելու համար դիմել նույն՝ 16 բլոկին:

Դժվար չէ նկատել, որ կառուցված ալգորիթմի իրակա–նացման ամենաերկար ժամանակը սպասվում է, երբ բոլոր թվանշանները ’5’–ից փոքր են: Այս դեպքում կգործեն բո–լոր երեք փուլերը, այսինքն երեք ցիկլերը՝ յուրաքանչյուրը k անգամ, որտեղ k–ն թվանշանների քանակն է: Ալգորիթմի աշխատանքը կարելի է արագացնել, կրճատելով ցիկլերի քանակը՝ եթե դա հնարավոր է, համատեղելով երկու և ավելի ցիկլեր մեկի մեջ: Մեր դեպքում ալգորիթմի տրամա-բանությունը թույլ է տալիս այդպիսի համատեղում: Մնում է պարզել՝ ո՞ր երկուսը:

Երեք ցիկլերի համատեղումը հնարավոր չէ վերը նշված պատճառով, այն է. թվանշանների ’արտաքսման’ և հետագա օգտագործման հերթականությունները տարբեր են: Երկրորդ և երրորդ փուլերի (ցիկլերի) համատեղումը իմաստ չունի, որովհետև, եթե ոչ բոլոր թվանշաններն են փոքր ’5’–ից, ապա նոր թվի ձևավորման վրա կորցրած ժամանակը կլինի ավելորդ: Մնում են առաջին երկու փուլերը: Այս երկու փու-լերի համատեղումը հնարավոր է, եթե փոխենք ալգորիթմի տրամաբանությունը: Եթե կառուցված ալգորիթմում բոլոր թվանշանները սկզբում ’արտաքսվում’ էին հետո նոր ստուգվում, ապա նոր ալգորիթմում կարելի է յուրաքանչյուր թվանշան ստուգել անմիջապես այն որոշելուց հետո, այսինքն՝ առաջին փուլում՝ հրաժարվելով երկրորդ փուլից: Այս տարբերակում առաջին ցիկլն է դառնում երկու ելքով ցիկլ, որոնցից մեկը բնական է, իսկ մյուսը՝ ստիպողական: Ցիկլը կավարտի իր աշխատանքը բնական ձևով, եթե բոլոր թվանշանները լինեն ’5’–ից փոքր, և ժամանակից շուտ, հենց որ հանդիպենք ’5’–ից մեծ կամ հավասար թվանշանի: Նկ.2.20–ում բերված է խնդիր 24–ի ցիկլների համատեղմամբ՝ Ա24.2 ալգորիթմի բլոկ–սխեման: Բերված սխեմայից պարզ



Նկ.2.20: Խնդիր 24 լուծման
Ա24.2 ալգորիթմի բլոկ-սխեմա:

երևում է, որ ստիպողական ելքի դեպքում շրջանցվում է երկրորդ փուլը և կատարվում է անցում արդյունքի արտածման բլոկին:

Համեմատելով կառուցված ալգորիթմների երկու տարբերակները, կարող ենք արձանագրել, որ բլոկների քանակի առումով նրանց միջև գրեթե ոչ մի տարբերություն չկա: Սակայն, ինչպես նշվել է, դա՜ չի արժեքավորում ցիկլիկ ալգորիթմները, այլ նրանց իրականացման ժամանակը, որը նաև կախված է ցիկլերի քանակից: Երկրորդ տարբերակում ցիկլերի քանակը մեկով պակաս է, իսկ ցիկլերի կըրկընություննների թիվը մնացել է անփոփոխ::

Պարզագույն ցիկլերով ալգորիթմների կառուցման օրինակները կարելի է անվերջ շարունակել, սակայն, ինչպես բազմիցս նշվել է, նման առատությունը դժվար թե որակ ավելացնի: Մենք ձեզ հետ բավականին շատ տարաբնույթ ցիկլեր կառուցեցինք, որպեսզի դուք ինքնուրույն լուծեք ներքո առաջարկվող խնդիրները և միասին շարունակենք ուսումնասիրել ավելի բարդ՝ ներդրված ցիկլերի կառուցման սկզբունքները:

ՎԱՐԺՈՒԹՅՈՒՆՆԵՐ

1.Փոփոխելով **x** փոփոխականի արժեքը **[1;8]** հատվածում **Δx=0.2** քայլով հաշվել և արտածել **y** ֆունկցիայի արժեք–ները.
$$y = \begin{cases} 2x+1 \ , & \textit{եթե} \ x < 5, \\ x^2 - 1 \ , & \textit{հակառակ դեպքում :} \end{cases}$$

2. Փոփոխելով **x** փոփոխականի արժեքը **[1;12]** հատ–վածում **Δx** քայլով հաշվել և արտածել **y** ֆունկցիայի արժեքները.
$$y = \begin{cases} 5x+2 & , \ \textit{եթե} \ x < 3, \\ x^2 + x - 1 & , \ \textit{եթե} \ 3 \le x \le 10, \\ 1 & , \ \textit{մնացած դեպքերում :} \end{cases}$$

Խնդրի լուծման ալգորիթմը կառուցել մեկ ցիկլով:

3. Տրված **N** բնական թվի համար հաշվել և արտածել
$$S = \frac{\sin 1}{\cos 1} + \frac{\sin 1 + \sin 2}{\cos 1 + \cos 2} + \cdots + \frac{\sin 1 + \cdots + \sin N}{\cos 1 + \cdots + \cos N} :$$

4. Տրված են՝ բնական **n** և իրական **x** թվերը: Հաշվել

$$S = \sin x + \sin^2 x + \sin^3 x + \cdots + \sin^n x \text{ :}$$

5. Տրված են՝ բնական **N** և իրական **x** թվերը: Հաշվել

$$S = \sin x + \sin \sin x + \cdots + \underbrace{\sin \sin \cdots \sin x}_{N} \text{ :}$$

6. Տրված **N** բնական թվի համար հաշվել

$$y = \sqrt{3 + \sqrt{6 + \cdots + \sqrt{3(N-1) + \sqrt{3N}}}} \text{ :}$$

7. Տրված **x** իրական և **n** բնական թվերի համար հաշվել

$$S = \frac{\sin(x+1)}{x} - \frac{\sin(x+2)}{x^2} + \cdots + (-1)^{n+1} \cdot \frac{\sin(x+n)}{x^n} \text{ :}$$

8. Տրված **x** իրական և **n** բնական թվերի համար հաշվել

$$y = \prod_{i=1}^{n} \frac{\cos(ix)}{2^i \cdot i!} \text{ :}$$

9. N բնական թվի համար ընդունենք, որ N!! նշանակում է. **1·3·5·…·N** կենտ N–ի համար և **2·4·6·…·N** զույգ N–ի համար: Հաշվել y=N!! կամայական N–ի համար:

10. Տրված է A ամբողջ թիվը: Որոշել այն ամենամեծ i ամբողջ թիվը, որի դեպքում 4^i<A:

11. Տրված է N բնական թիվը: Հաշվել

y=1·2 + 2·3·4 + … + N·(N+1)·…·2N:

12.Որոշել տրված N բնական թվի թվանշանների գումարը:

13. Տրված N բնական թվի թվանշանների հերթականու–թյունը փոխել հակադարձով: Օրինակ՝ 3275 թվից ստանալ 5723 թիվը:

3. ՆԵՐԴՐՎԱԾ ՑԻԿԼԵՐ

Նախորդ բաժնում դիտարկված խնդիրների լուծման ալգորիթմները ցիկլիկ բնույթի էին և, որպես կանոն, ոչ մի ցիկլի մարմին իր կազմում չէր պարունակում այլ ցիկլեր: Դրա անհրաժեշտությունը չի էլ նկատվել, որովհետև հաշվարկվող ֆունկցիաները ռեկուրենտ բնույթի էին, և վերջնական արդյունքը ստացվում էր ցիկլի կրկնությունից կրկնություն՝ մեկ արժեքից մյուսը ստանալով: Եթե ֆունկ–ցիան կախված էր այլ, նույնպես ռեկուրենտ, ֆունկցիայից, ապա հնարավորություն գտանք միևնույն ցիկլում հաշվել և ներդրված ֆունկցիայի հերթական արժեքը, և հիմնական ֆունկցիայինը: Սակայն բազմաթիվ խնդիրներում հանդիպում է հակառակ իրավիճակ, երբ ներդրված ֆունկցիան հնարավոր չէ հաշվել զուգահեռ հիմնականին, իսկ եթե այն նույնպես ցիկլիկ բնույթի է, ապա մենք ստիպված ենք հիմնական ցիկլի մարմնում կիրառել մեկ այլ ցիկլ՝ երկրորդ ֆունկցիան հաշվելու համար: Այս դեպքում գործ ունենք **ներդրված** ցիկլերի հետ:

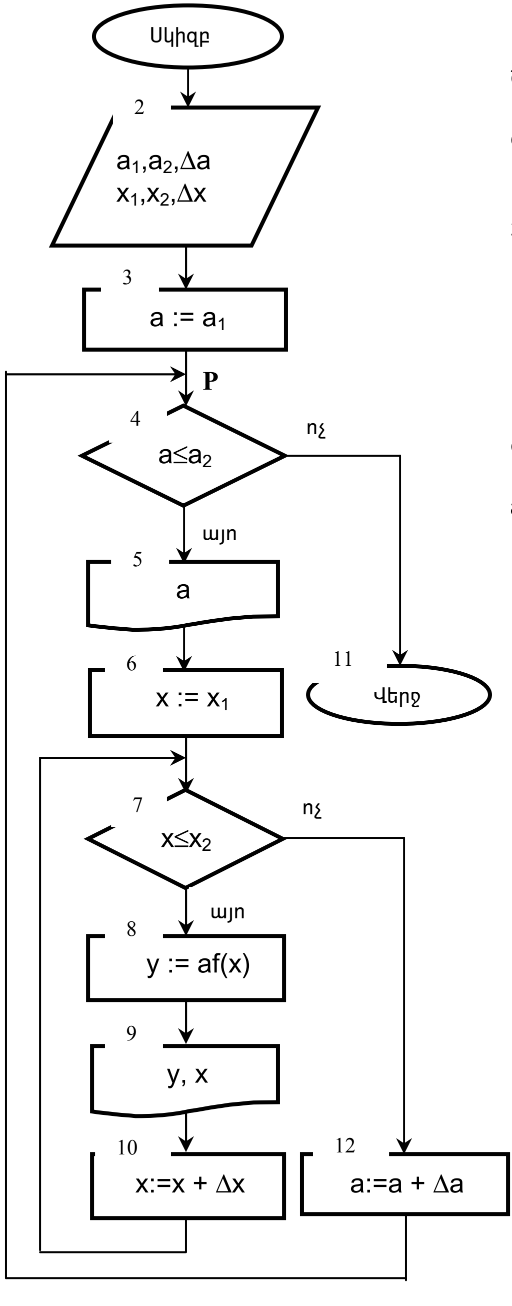
Ներդրված ցիկլերի անհրաժեշտությունը առաջանում է ոչ միայն նշված դեպքերում, այլ նաև բազմապարամետ–րական ֆունկցիաների արժեքները հաշվելիս: Ընդհանուր դեպքում՝ $g(z_1,z_2,...,z_n)$ ֆունկցիան կախված է **n** պարա–մետրերից, որոնց արժեքները փոփոխվում են տարբեր տիրույթներում զանազան քայլերով: Բնական է, որ այս դեպքում անիմաստ է միևնույն ցիկլում միաժամանակ փո–փոխել բոլոր պարամետրերի արժեքները, քանի որ նրանց թույլատրելի արժեքների քանակները տարբեր են և միա–ժամանակ սպառելու հավանականությունը գրեթե հավա–սար է զրոյի: Բազմապարամետրական ֆունկցիաների ընդհանուր դեպքին մենք կանդրադառնանք քիչ ուշ, իսկ առայժմ մեկ օրինակ:

<u>Խնդիր 25:</u> Հաշվել և արտածել **y=a·f(x)** ֆունկցիայի արժեքները, երբ **a** և **x** պարամետրերը ընդունում են ար–ժեքներ համապատասխանաբար՝ $A=[a_1,a_2]$ և $X=[x_1,x_2]$ հատվածներից ($a\in A$, $x\in X$) Δa և Δx քայլերով:

Միակ պահանջը, որ կարող ենք ներկայացնել ֆունկ–ցիային՝ պարամետրերի նշված տիրույթներում այն պետք է լինի անընդհատ, առանց խզումների: Հաջորդ հարցը վերաբերվում է ֆունկցիայի բովանդակությանը, ինչը և որոշում է կազմակերպվող գործընթացի տրամաբանությունը: **a** գործակցի կամայական արժեքի դեպքում y ֆունկցիայի գրաֆիկը այնպիսի կոր է, որը կարող ենք ստանալ Ա11 ալգորիթմի կիրառմամբ՝ ֆիքսելով գործակցի արժեքը: Փոփոխելով գործակցի արժեքը, մենք, բնական է, կստանանք զուգահեռ f(x) կորերի բազմություն: Հետևաբար, խնդրի դրվածքը կարելի է վերաձևակերպել հետևյալ տեսքի. ***փոփոխելով գործակցի արժեքը A հատվածում Δa քայլով, յուրաքանչյուր անգամ կառուցել y=a·f(x) կորը:*** Իսկ այսպիսի ձևակերպմամբ խնդրի լուծման ալգորիթմը նույն Ա11 ալգորիթմն է՝ այն տարբերությամբ, որ **a** պարամետրով ցիկլում (**արտաքին** ցիկլ) **y** ֆունկցիան հաշվարկվում է ոչ թե մեկ վերագրման գործողության շնորհիվ (նկ.2.3), այլ նույնպիսի շրջափուլային գործընթացի (**ներքին** ցիկլ) արդյունքում, փոփոխելով **x** պարամետրի արժեքը **X** հատվածի վրա որոշակի քայլով: Այսպիսով, բավական է **a** պարամետրով ցիկլում տեղադրել նմանատիպ՝ **x** պարամետրով ցիկլ և մենք կստանանք տվյալ խնդրի լուծման ալգորիթմը: Ահա՜ ձեզ ալգորիթմների մոդուլային սկզբունքով կառուցման ևս մեկ օրինակ:

Նկ.3.1–ում բերված է տվյալ խնդրի լուծման ալգորիթ–մի համառոտ տարբերակը, որտեղ նշված են բոլոր իրա–կանացվող գործողությունները, իսկ նկ.3.2–ում՝ նույն ալ–գորիթմի երկրորդ տարբերակը 'մոդիֆիկացիա' բլոկի կիրառմամբ: Ակնհայտ է, որ երկու գծանկարների համեմատությունը երկրորդ տարբերակի օգտին է, սակայն առայժմ ավելի հարմար է կատարել ալգորիթմի առաջին տարբերակի բացատրությունը: Հետագայում նման համեմատություններ չեն արվի, հարգելով ձեր ձեռք բերած փորձն ու ունակությունները:

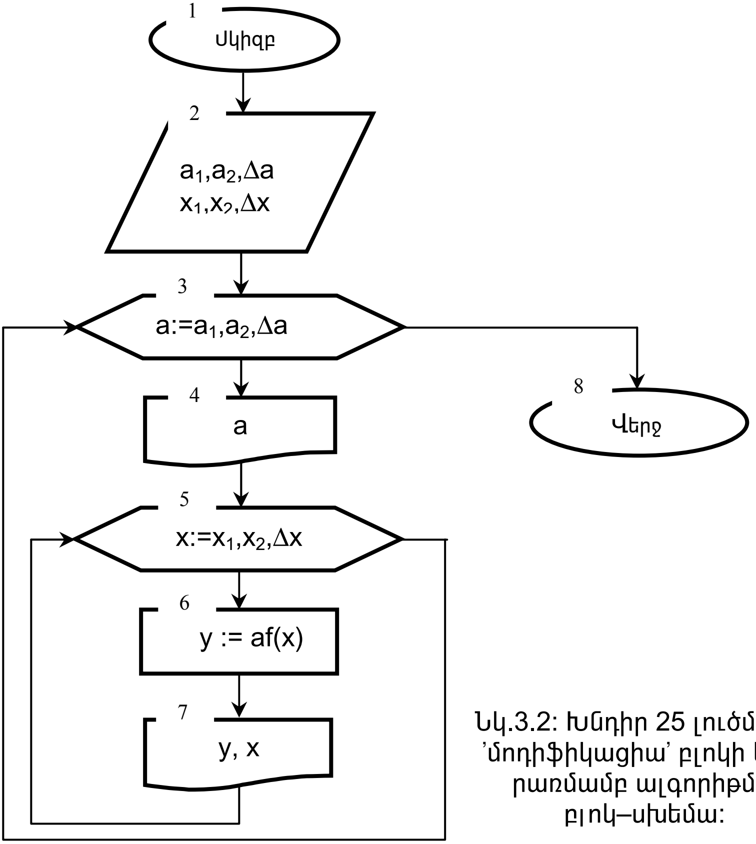
Եվ այսպես, տվյալների ներմուծումից հետո (բլ.2) **a** պարամետրին վերագրում ենք առաջին՝ a_1 արժեքը (բլ.3): Այնուհետև, ***քանի դեռ $a \le a_2$ կատարում ենք 5, 6...10 և 12 բլոկները:*** Այստեղ 6...10 բլոկները իրականացնում են **y** ֆունկցիայի արժեքների հաշվումը **X** հատվածի վրա **Δx** քայլով և նրանց արտածումը, որի ընթացքում **a** պարա–մետրի արժեքը մնում է անփոփոխ: Այսինքն, արդյունքում ստանում ենք կորերից մեկը՝ **a** պարամետրի որոշակի արժեքին համապատասխան: Հաջորդ կորը ստանալու հա–մար անհրաժեշտ է գործակցի ընթացիկ արժեքին գումարել **Δa** քայլի արժեքը (բլ.12) և թույլատրելի արժեքների սահմանում գտնվելու դեպքում կրկնել ***5, 6÷10 և 12 բլոկները***, այսինքն՝ արտաքին ցիկլի աշխատանքը:



Նկ.3.1: Խնդիր 25 լուծման ալգորիթմի ընդարձակ բլոկ-սխեմա:

Այստեղ շատ կարևոր է ճիշտ որոշել թիվ 6 բլոկի տեղը ընդհանուր գործընթացի սահմաններում: Խոսքն այն մասին է, որ այս բլոկում ինչպես և 3–ում, կատարվում է նախնական արժեքների վերագրում: Այդ հանգամանքը շատ դեպքերում հանդիսանում է սխալ որոշման պատճառ, որի հետևանքով **$x:=x_1$** գործողությունը հայտնվում է սխեմայի **P** կետից վերև՝ **$a:=a_1$** գործողության կողքին:

Սակայն չմոռանանք, որ 7…10 բլոկներից կազմված ցիկլի աշխատանքը ավարտելուց հետո **x** փոփոխականի արժեքը գերազանցում է սահմանային x_2–ը: Հետևաբար, անմիջապես հաջորդ կորը հաշվարկելուց առաջ անհրաժեշտ է վերականգնել **x** պարամետրի սկզբնական արժեքը: Ահա այս նկատառումից ելնելով տվյալ բլոկի տեղը սխեմայում որոշվել է՝ անմիջապես թիվ 7 բլոկից առաջ:



Նկ.3.2: Խնդիր 25 լուծման 'մոդիֆիկացիա' բլոկի կիռառմամբ ալգորիթմի բլոկ–սխեմա:

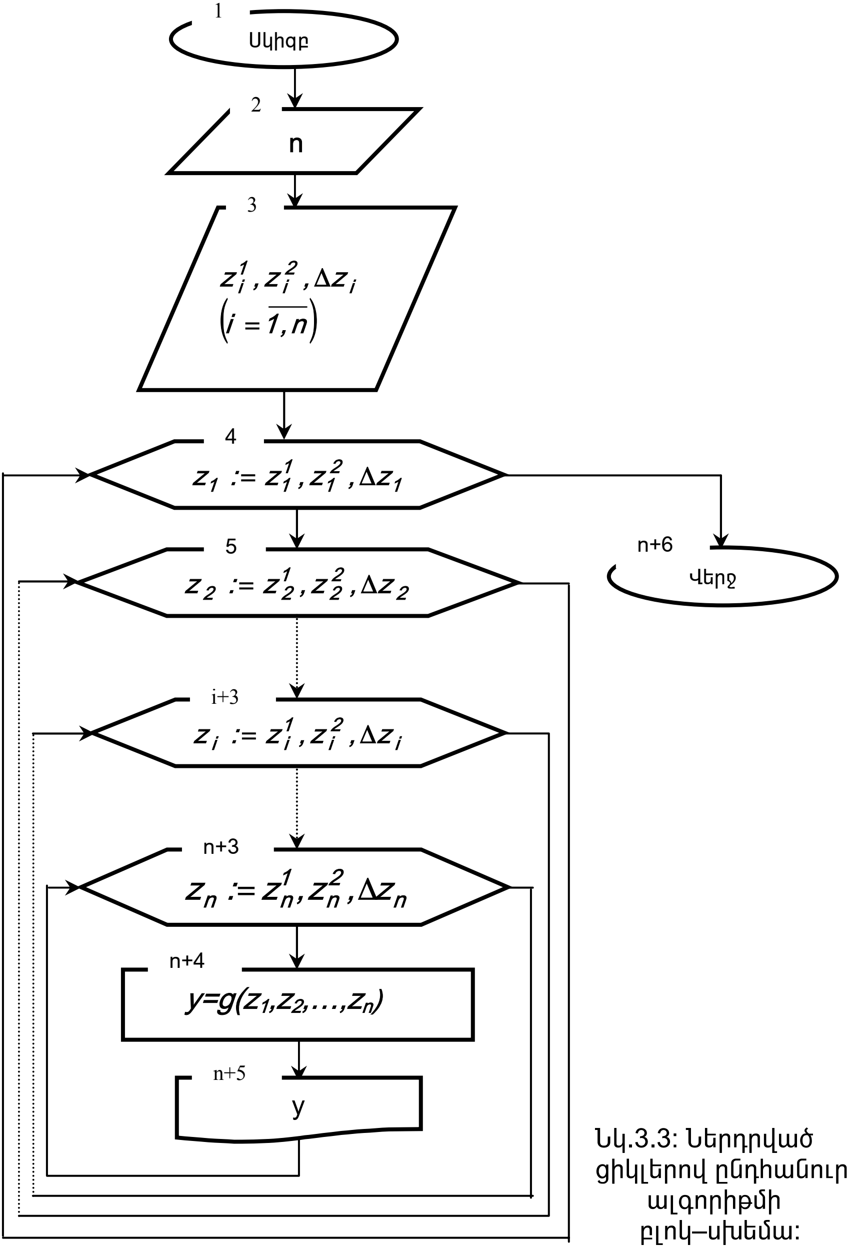
Նկ.3.2-ի վերաբերյալ միայն նշենք, որ սկսելով 5…7 բլոկներից կազմված ցիկլը, այն կավարտվի, երբ նրա **x** պարամետրը կսպառի բոլոր թույլատրելի արժեքները: Միայն այս դեպքում արտաքին ցիկլը կշարունակի գործ–ընթացը, նախապատրաստելով ներքին ցիկլի համար **a** գործակցի նոր արժեք: Մնում է արձանագրել ներքին ցիկլի պարամետրի ընտրության մեկ բնորոշիչ սկզբունք, որով պետք է այսուհետև ղեկավարվել ներդրված ցիկլեր կազմակերպելիս. *երկու պարամետրերից այն, որը մյուսի նկատմամբ ավելի հաճախ է փոփոխում իր արժեքը, պետք է հանդիսանա ներքին ցիկլի պարամետր, իսկ մյուսը՝ արտաքին:* Այս փաստը ցայտուն կերպով արտահայտված է վերջին խնդրի լուծման ալգորիթմում:

Քննարկված ֆունկցիան կարելի է համարել վերը բերված ընդհանուր ֆունկցիայի մասնավոր դեպք, այսինքն՝

$g(z_1,z_2)=a\cdot f(x)$, որտեղ $z_1=a$ և $z_2=x$: Եթե պարամետրերի քանակը ավելացնենք, վերցնելով երեք, չորս կամ ավել, ապա խնդրի բովանդակությունը չի փոխվի՝ միայն կերկարի: Ստացած արդյունքը ընդհանրացնելու նպատակով ենթադրենք, որ պարամետրերից յուրաքանչյուրը փոփոխվում է առանձին տիրույթում իրեն բնորոշ քայլով, օրինակ՝ $z_1 \in Z_1 = [z_1^1; z_1^2]$ Δz_1 քայլով, $z_2 \in Z_2 = [z_2^1; z_2^2]$ Δz_2 քայլով և այլն: Ընդունելով պարամետրերի քանակը հավասար **n** բնական թվի, կարող ենք գրել, որ z_i պարամետրը ընդունում է արժեքներ $Z_i = [z_i^1; z_i^2]$ հատվածից Δz_i քայլով (**i=1, 2,...,n**) և ձևակերպել ընդհանուր խնդիրը:

<u>Խնդիր 26:</u> Տրված է **n** պարամետրերից կախված $y=g(z_1,z_2,...,z_n)$ ֆունկցիա, որտեղ յուրաքանչյուր z_i պարամետրի արժեքը փոփոխվում է Z_i հատվածում Δz_i քայլով (**i=1,2,...,n**): Հաշվել և արտածել ֆունկցիայի բոլոր հնարավոր արժեքները:

Թվում է, թե անելանելի վիճակ է. հնարավո՞ր է, ար–դյոք, կազմակերպված ձևով, քայլ առ քայլ, անցնել ար–գումենտների բոլոր կոմբինացիաների վրայով, չկրկնելով ոչ մեկը: Չէ՞ որ այդպիսի հերթականության ընտրությունից է կախված մեր հաջողությունը: Քիչ խորհելուց հետո կարող ենք հանգել այն մտքին, որ, կիրառելով վերը քննարկված ներդրված ցիկլերի գաղափարը, ներքին ցիկլի պարամետրի ընտրության սկզբունքը, տվյալ խնդիրը կարող է ստանալ շատ կարճ և գեղեցիկ լուծում, որը բերված է նկ.3.3–ում: Այստեղ տրված **n** պարամետրերից յուրաքանչյուր ցիկլի պարամետրի ընտրությունը կախված է կոնկրետ խնդրի պայմաններից և պարամետրերի բովանդակությունից: Մյուս կողմից, ներդրված ցիկլեր պարունակող ալգորիթմներով հաշվման հետևանքով երբեմն ստացվում են, կարելի է ասել, շատ մեծ քանակությամբ արժեքներ: Նման պարագայում կարևոր դեր է հատկացվում պարամետրերի արտածման հարցին, թե ո՞ր պարամետրերը և ե՞րբ պետք է արտածվեն: Անգամ ֆունկցիայի հաշվարկված ոչ բոլոր արժեքներն են ենթակա արտածման: Բոլոր դեպքերում արտածման հարցերը որոշվում են ծրագրավորողի հայեցողությամբ:



Նկ.3.3: Ներդրված ցիկլերով ընդհանուր ալգորիթմի բլոկ–սխեմա:

Հաջորդ խնդիրը, թվում է, լիովին կպարզաբանի ընդ–

հանուր սխեման: Սակայն մինչև խնդրի ձևակերպումը բերենք մեկ սահմանում: Զույգ կարգանի բնական թիվը կոչենք 'երջանիկ', եթե այդ թվի թվանշանների մի կեսի գումարը հավասար է մյուս կեսի գումարին: Օրինակ՝ 370541 թիվը 'երջանիկ' է, որովհետև 3+7+0=5+4+1, իսկ 272591 թիվը՝ ոչ, որովհետև 2+7+2≠5+9+1:

<u>Խնդիր 27:</u> Որոշել քառանիշ թվերից բոլոր 'երջանիկ' թվերը:

Միգուցե ձեզանից ոմանք, գնալով թարմ հետքերով, փորձեն այս խնդիրը լուծել, կիրառելով Ա23 ալգորիթմը: Այսինքն, N_4=[1000;9999] թվերից յուրաքանչյուրը վերլուծելով բաղադրիչ չորս թվանշանների և համեմատելով առաջին երկուսի գումարը հաջորդ զույգի գումարի հետ: Դե ինչ, որպես անցած թեմայի կրկնություն, կարող ենք միայն ողջունել նման ցանկությունը: Ձեր մտահաղացումը իրականացնելուց հետո համեմատեք ստացած արդյունքը ստորև առաջարկված լուծման հետ:

Փորձենք խնդրին մոտենալ այլ տեսանկյունից՝ ոչ թե վերլուծելով N_4 բազմության թվերը բաղադրիչ թվանշանների, այլ ընտրելով չորս անկախ թվանշաններից կազմած բոլոր հնարավոր կոմբինացիաները: Դիցուք, քառանիշ թվի թվանշանները նշանակված են՝ **a**, **b**, **c** **d** տառերով, որոնցից յուրաքանչյուրը կարող է ընդունել 0...9 արժեքներ: Գուցե և հնարավոր բոլոր դեպքերը ստանալու ամենակարճ ճանապարհը հետևյալն է.

1 0 0 0	1 0 1 0	1 0 2 0	. . .	1 1 0 0	1 1 1 0	. . .
1 0 0 1	1 0 1 1	1 0 2 1	. . .	1 1 0 1	1 1 1 1	. . .
1 0 0 2	1 0 1 2	1 0 2 2	. . .	1 1 0 2	1 1 1 2	. . .
. . . . .	. . . . .	. . . . .	. . .	. . . . .	. . . . .	. . .
1 0 0 9	1 0 1 9	1 0 2 9	. . .	1 1 0 9	1 1 1 9	. . .

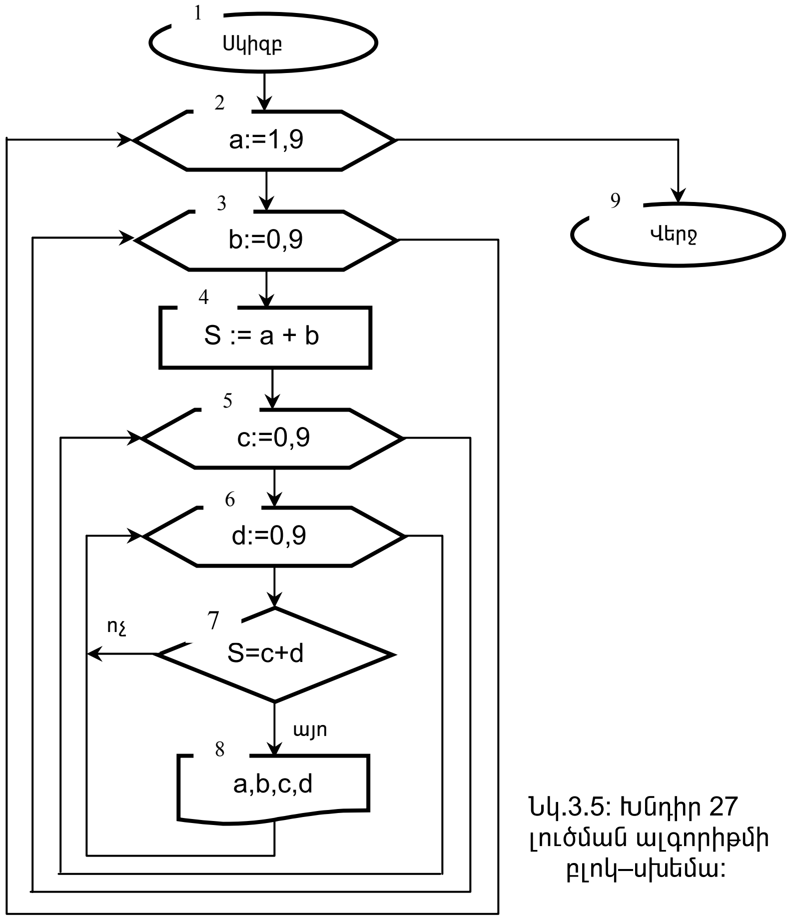
Նկ.3.4: Չորս թվանշաններից կազմած բոլոր տեղափոխությունների ստացման հերթականությունը:

Ուշադիր նայելով բերված աղյուսակին, կարելի է նկատել հետևյալ օրինաչափությունը, որ յուրաքանչյուր սյան մեջ փոփոխվում է միայն կրտսեր՝ **d** թվանշանը, ընդունելով հնարավոր բոլոր 10 արժեքները: Սյունից սյուն անցնելիս, մեկ միավորով աճում է **d** թվանշանին հարակից հաջորդ՝ **c** թվանշանը: Այդպես շարունակվում է մինչև 10–րդ սյունը, երբ **c** պարամետրը սպառում է հնարավոր բոլոր արժեքները: Այնուհետև մեկ միավորով աճում է **c**–ին կից՝ **b** պարամետրը, **c**–ն ընդունում է '0' արժեք և ամեն ինչ կրկնվում է: Այսինքն, առաջին 10 սյուներում իրականացվող գործողությունները նույնությամբ կրկնվում են հաջորդ 10×10 սյուներում՝ այն տարբերությամբ, որ յուրաքանչյուր տասնյակից հետո աճում է **b** կարգի արժեքը: Շարունակելով նման դատողությունները կարող ենք եզրակացնել, որ յուրաքանչյուր 100 սյունակից հետո աճում է **a** կարգի արժեքը: Եվ այսպես 9 անգամ՝ ավագ կարգի ընդունած արժեքների քանակով:

Անկախ այն բանից, որ վերը բերված գործընթացի բանավոր նկարագիրը լիովին պարզաբանում է երևույթը, այնուամենայնիվ, այն ամբողջական չէ և չի էլ կարող լինել այդպիսին, քանի որ ալգորիթմների բանավոր նկարագրումը անզոր է մանրամասն ներկայացնելու իրականացվող բոլոր գործողությունները: Ինչպես նկատելի է, միջանկյալ բոլոր գործողությունների պարզաբանումը թողնված է մեր երևակայությանը: Եթե քառանիշ թվերի դեպքում մեր համբերությունը կների բացատրագիրը լրացնելու, ապա ընդհանուր դեպքում՝ **n**–կարգանի թվերի համար ստիպված ենք ընդհանրապես հրաժարվել այդ մտքից: Մնում է դիմել արդեն փորձություն անցած և իրեն դրական կերպով դրսևորած բլոկ–սխեմայի օգնությանը:

Այս խնդրի լուծման ալգորիթմը լիովին համապատասխանում է բազմապարամետրիկ ֆունկցիայի հաշվման ընդհանուր սխեմային և, հետևաբար, կարելի է կիրառել Ա26 ալգորիթմը: Դրանում համոզվելու համար բավական է վերը բերված աղյուսակի բանավոր նկարագիրը վերաձևակերպել, շարադրելով հետևյալ կերպ. ***փոփոխելով a թվանշանի արժեքը [1;9] հատվածում, յուրաքանչյուրի համար փոփոխել b թվանշանի արժեքը [0;9] հատվածում, որոնցից յուրաքանչյուրի համար փոփոխել c թվանշանի արժեքը [0;9] հատվածում, որոնցից յուրաքանչյուրի համար փոփոխել d թվանշանի արժեքը [0;9] հատվածում և եթե թվանշանների յուրաքանչյուր կոմբինացիայի համար a+b=c+d, ապա արտածել abcd թիվը:*** Նման ձևակերպմամբ ալգորիթմի բլոկ–սխեման բերված է նկ.3.5–ում:

Ինչպես երևում է գծանկարից, արտածվում են միայն այն քառանիշ թվերը որոնց թվանշանները բավարարում են խնդրում դրված պայմանին: Մյուս կողմից, ակնհայտ է, որ կարելի է հրաժարվել թիվ 4 բլոկից, տեղադրելով (a+b) արտահայտությունը թիվ 7 բլոկում ստուգվող առնչության ձախ կողմում: Սակայն խորհուրդ չենք տա դիմել նման քայլի, քանի որ բերված ձևը տնտեսում է հաշվարկման ընդհանուր ժամանակը. չէ՞ որ թիվ 5 բլոկից սկսվող ցիկլում **a** և **b** թվանշանները փոփոխության չեն ենթարկվում: Դրանում համոզվելու համար հաշվենք տվյալ սխեմայով իրականացվող ցիկլերի քանակը, որը անուղղակի կերպով տալիս է տվյալ ալգորիթմի ժամանակային բնութագիրը:



Նկ.3.5: Խնդիր 27 լուծման ալգորիթմի բլոկ–սխեմա:

Դատելով վերը բերված աղյուսակի նկարագրից, ներքին՝ **d** պարամետրով ցիկլի կրկնությունների քանակը հավասար է 9×10^3: Ընդ որում թիվ 4 բլոկը կատարվում է 9×10 անգամ. այնքան, ինչքան որ փոփոխվում է **b** թվանշանը: Այս երկու թվերի տարբերությունը ցույց է տալիս (a+b) գումարման գործողության տնտեսված քանակը, որը կազմում է՝ 9000–90=8910: Հետագա մեկնաբանությունները, թվում է, ավելորդ են:

Պետք է նշել, որ տվյալ խնդրում կարևոր չէ ներդրված ցիկլերի պարամետրերի ընտրության կարգը. ինչ հերթականությամբ էլ նրանք փոփոխվեն միևնույն է բոլոր հնարավոր տեղափոխություններով մենք կանցնենք: Որոշ դեպքերում միայն կարող է վերանալ թիվ 4 բլոկը: Պարզապես ընտրված հերթականությունը ապահովում է երթը մեկ ուղղությամբ՝ աջից ձախ, ինչը հեշտացնում է ալգորիթմով ստացվող արդյունքների հսկումը:

Այսպիսով, դուք ծանոթացաք, տվյալ դեպքում չորս, իսկ ընդհանուր դեպքում **n** թվերից կազմած բոլոր տեղափոխությունների ընտրման ալգորիթմին, որը հաճախակի է կիրառվում գործնականում:

Հետևելով նախորդ բաժնում քննարկվող խնդիրների ընտրման տրամաբանությանը՝ հաջորդ քայլում առաջարկում ենք լուծել հետևյալ խնդիրը:

<u>Խնդիր 28:</u> Հաշվել և արտածել

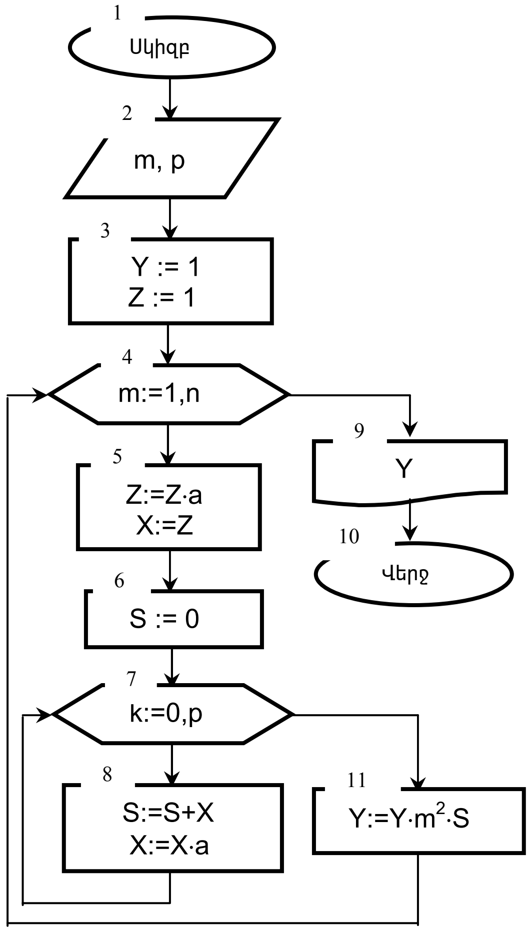
$$Y = \prod_{m=1}^{n}\left[m^2 \sum_{k=0}^{p} a^{m+k} \right]: \qquad (3.1)$$

Բերված արտահայտությունը այնպիսի գումարների արտադրյալ է, որտեղ յուրաքանչյուր գումար ընդգրկված է իր դիրքին համապատասխան գործակցով: Եթե հնարավոր լիներ m–րդ գումարը (S_m) ստանալ (m–1)-րդ գումարից (S_{m-1}), ապա տվյալ խնդրի լուծումը հնարավոր կլիներ ներկայացնել մեկ ցիկլի միջոցով՝ օգտագործելով ռեկուրսիայի գաղափարը: Համեմատելով այդ երկու գումարների արտահայտությունները.

$$S_{m-1} = a^{(m-1)+0} + a^{(m-1)+1} + \cdots + a^{(m-1)+p} \quad \text{և}$$

$$S_m = a^{m+0} + a^{m+1} + \cdots + a^{m+p},$$

նման նպատակից ստիպված ենք հրաժարվել: Սակայն եթե գումարից գումար ստանալը անհնար է, ինչպես տեսնում եք, կարելի է S_m գումարի առաջին գումարելին ստանալ նախորդ՝ S_{m-1} գումարի առաջին գումարելիից, բազմապատկելով **a** մեծության հետ:



Նկ.3.6: Խնդիր 28 լուծման ալգորիթմի բլոկ–սխեմա:

Այլ բան է՝ տրված **p** և **m** պարա–մետրերի համար S_m գումար հաշվելը: Ինչպես արդեն գի–տենք, մեկ գումարի սահման-ներում a^{m+k} արտահայ-տությունը ռեկուրսիվ բնույթի է, քանի որ հաստատուն **m**–ի դեպ-քում փոփոխվում է մի-այն աստիճանացույցի **k** բաղադրիչը:

Այսպիսով, արտա–դրյալ հաշվող ցիկ–լում անհրաժեշտ է նախատեսել գումար հաշվող ցիկլ, որը պետք է գործի m–ի յուրաքանչյուր արժեքի համար, այսինքն՝ եկանք ներդրված երկու ցիկլերի կիրառման գաղափարին: Այս խնդրի լուծման ալ-գորիթմի բլոկ–սխեման բերված է նկ. 3.6–ում:

Այստեղ Z տա–ռով նշանակված է յուրաքանչյուր S_m գու-մարի առաջին գումա-րելին: Նախապես վե-րագրելով Z–ին ’1’ ար-ժեք (բլ.3), **m** պարա-մետրով արտաքին ցիկլի յուրաքանչյուր կրկնության սկզբում այն աճում է **a** անգամ (բլ.5), ընդունելով a^m արժեք:

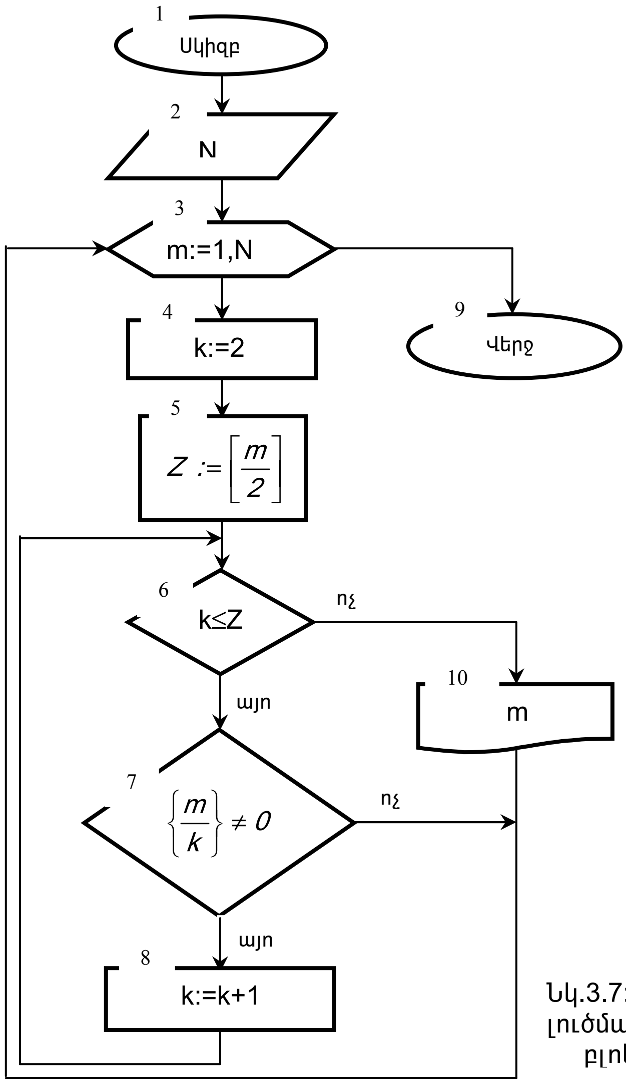
7 և 8 բլոկներից կազմված ներքին ցիկլում Z փոփո–խականը պետք է փոխի իր արժեքը a^m–ից մինչև a^p արժեքը, հաջորդաբար բազմապատկվելով **a**–ով: Սակայն մենք չենք կարող թույլ տալ, որ Z–ը փոխի իր արժեքը, այլապես հաջորդ գումարի համար չենք ունենա առաջին՝ a^{m+1}

գումարելին: Հետևաբար, մինչև ներքին ցիկլի առաջին կատարումը անհրաժեշտ է կրկնօրինակել Z փոփոխականի արժեքը մեկ այլ փոփոխականում, օրինակ, X–ում: Հետագայում փոփոխվում է X–ի արժեքը՝ թողնելով Z–ի արժեքը անփոփոխ: Միայն ավարտելով ներքին ցիկլի աշխատանքը՝ մենք կունենանք հերթական գումարի արժեքը, որը կբազմապատկենք համապատասխան գործակցով և կընդգրկենք Y արտադրյալի մեջ (բլ.11): Եվ այդ բոլորն արտաքին ցիկլի ներքո:

<u>Խնդիր 29</u>: Որոշել 1–ից մինչև N բնական թվերից բոլոր պարզ թվերը:

Առաջին հայացքից խնդրի լուծման ալգորիթմը կարելի կառուցել մեկ ցիկլից, որի պարամետրը պետք է փոփոխվի նշված սահմաններում և յուրաքանչյուր արժեքի համար ստուգվի պարզության պայմանը: Սակայն, հաշվի առնելով պարզ թվի սահմանումը, մենք չենք կարող պնդել, թե թվի պարզությունը որոշվում է մեկ պայմանով: Ուրեմն ո՞ր թիվն է կոչվում պարզ: Հիշեցման կարգով. այն բնական թիվը, որը անմնացորդ բաժանվում է միայն մեկի և իր վրա, կոչվում է պարզ թիվ: Իսկ Ինչպե՞ս կարելի է ստուգել թվի բաժանարարները, եթե ոչ՝ հաջորդաբար բաժանելով տվյալ թիվը իրենից փոքր բոլոր բնական թվերի վրա և ստուգելով բաժանման մնացորդը: Հետևաբար առաջանում է երկրորդ ցիկլի կազմակերպման անհրաժեշտությունը, որը, բնական է, պետք է ընդգրկված լինի առաջինի մեջ: Ընդ որում, եթե արտաքին ցիկլը պետք է կատարվի որոշակի քանակությամբ՝ N անգամ, ապա նույն բանը չի կարելի ասել ներքին ցիկլի մասին, որովհետև այն պետք է ավարտել, հանդիպելով ստուգվող թվից և մեկից տարբերվող բաժանարարի:

Օգտվելով մեր իսկ կողմից սահմանված միջոցներից՝ ձևակերպենք խնդրի լուծման ալգորիթմը ընդհանուր տեսքով: ***Փոփոխելով m փոփոխականի արժեքը 1–ից մինչև N մեկ քայլով, յուրաքանչյուր արժեքի համար, սկսած k=2 թվից, կատարել. քանի դեռ (k<m) և m թիվը չի բաժանվում k թվի վրա փոխել k–ն:*** Այսպես կառուցված ներքին ցիկլը, ինչպես հիշում եք, ունի երկու ելք. բնական և ստի–պողական: Ըստ շարադրված նախապայմանների ներքին ցիկլը ավարտում է աշխատանքը բնական ձևով, եթե **m** թիվը չի բաժանվում անմնացորդ իրենից փոքր և ոչ մի թվի վրա, որի հետևանքով ստանում ենք՝ **k>m** պայմանը: Ցիկլը ընդհատում է իր աշխատանքը, երբ հանդիպում է առաջին բաժանարարին:



Նկ.3.7: Խնդիր 29 լուծման ալգորիթմի բլոկ–սխեմա:

Ալգորիթմի բլոկ–սխեման բերված է նկ.3.7–ում, որտեղ պարզ երևում են երկու ելքերից հետևող շարունակությունները:

Գծանկարի և շարադրանքի միջև դուք կարող էիք նկատել մեկ տարբերություն, որը կապված է ալգորիթմի արագացման հետ: Եթե մինչև այժմ չեք նկատել, ապա ասենք, որ արագացման նպատակով ներքին ցիկլի կրկնությունների քանակը կարելի է կրճատել, փնտրելով m թվի բաժանարարները ոչ թե **[2;m–1]** հատվածին պատկանող բնական թվերի մեջ, այլ 2–ից մինչև **m** թվի կեսը, քանի որ կեսից բարձր թվերի վրա բաժանելն անիմաստ է և հավասարազոր է ժամանակը վատնելուն: Բերված բլոկ–սխեմայում **m** թվի կեսի փոխարեն վերցված է $z = \left[\frac{m}{2}\right]$ ար–

ժեքը, որը, ինչպես հիշում եք, նշանակում է՝ **m** թվի կեսի ամբողջ մասը, այսպես կլորացնելով **m**–ի կենտ արժեքից առաջացող կոտորակային թիվը:

Գոյություն ունի պարզ թվերի որոշման ավելի հնարամիտ, ավելի գեղեցիկ ալգորիթմ, որի արագագործությունը պայմանավորված է նրանով, որ 1–ից N բոլոր թվերը վերլուծելու փոխարեն հատուկ եղանակով հաջորդաբար կատարվում է անցում մեկ պարզ թվից իրենից մեծ հաջորդ պարզ թվին: Սակայն հիշեցնենք, որ մենք նպատակ չենք դրել այս ձեռնարկի սահմաններում ծանոթացնել ձեզ ներկայացված խնդիրների լուծման լավագույն տարբերակներին, ավելի հնարամիտ, ավելի արագագործ: Ամենևին էլ ոչ: Մեր նպատակները ավելի համեստ են, բայց պատվա–վոր, այն է՝ ծանոթացնել զանազան տեսակի ալգորիթմների կառուցման սկզբունքներին, այն մտածելակերպին, որը բնորոշ է ծրագրավորողին: Միայն տիրապետելով այս այբուբենին, կարելի է մատուցել ավելի բարդ կառուցվածքով, շատ անգամ հատուկ գիտելիքների կիրառում պահանջող ճշգրիտ ալգորիթմներ:

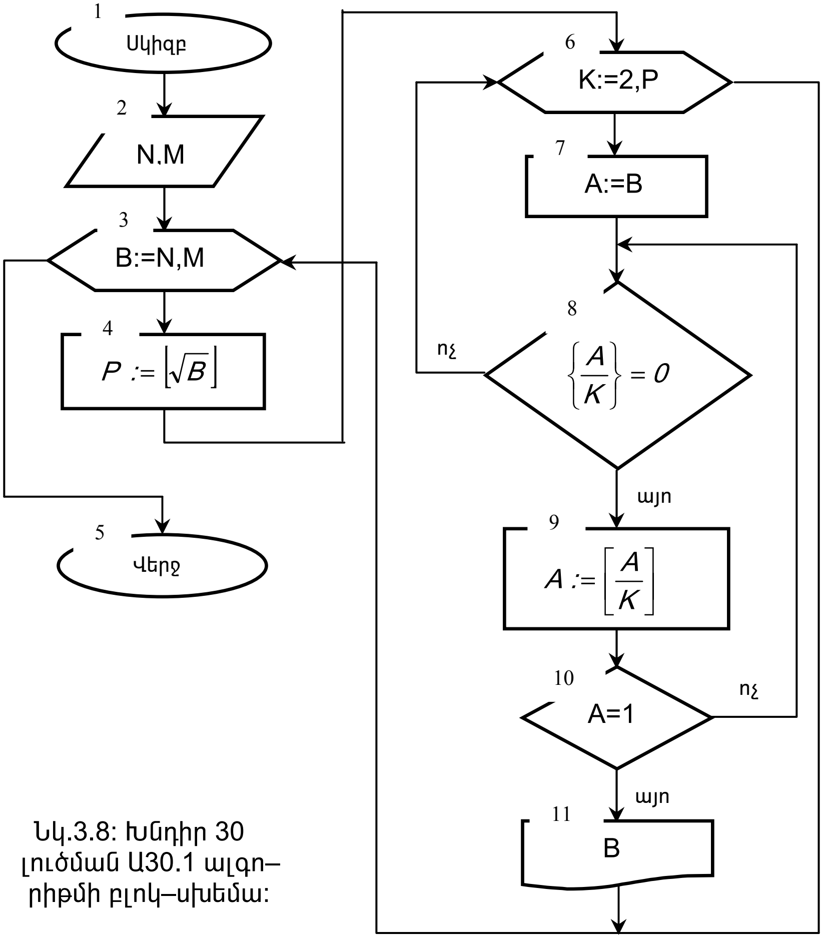
Յուրաքանչյուր խնդիր լուծելիս, որպես կանոն, հնա–րավոր չէ անմիջապես գտնել լավագույն լուծում՝ հատկապես սկսնակի համար: Լավագույնը ընտրվում է միևնույն խնդրի լուծման մի շարք ալգորիթմներից, ելնելով պահանջվող հատկանիշներից, երբեմն՝ նախասիրություններից: Միայն համեմատելով տարբերակները՝ կարելի որոշել նախընտրելին: Օրինակ, գտնվելով վերջին խնդրի տպավորության տակ, հաջորդ խնդիրը կարելի է լուծել միևնույն եղանակով, օգտագործելով երկու թվերի ամննացորդ բաժանման սկզբունքը: Սակայն, կառուցելով ալգորիթմի այլ տարբերակ և համեմատելով միմյանց հետ, կտեսնենք, թե առավելությունը որի կողմն է: Այսպես մենք հավատարիմ կմնանք արդեն ավանդույթ դարձրած սկզբունքին՝ յուրաքանչյուր բաժին ավարտել միևնույն խնդրի լուծման տարբեր ալգորիթմների կառուցմամբ և դրանց համեմատությամբ:

Խնդիր 30: Տրված են N և M բնական թվերը (N<M): Որոշել և արտածել [N;M] միջակայքի այն թվերը, որոնք հանդիսանում են որևէ թվի աստիճան:

Ելնելով խնդրի ձևակերպումից և նախորդ խնդրի թարմ փորձից՝ լուծման առաջին տարբերակը կարելի է ներկայացնել հետևյալ կերպ. ***N–ին գումարել մեկ մինչև M, ստուգելով՝ միջակայքի յուրաքանչյուր թիվ հանդիսանում է, արդյո՞ք, որևէ թվի աստիճան:*** Այսպիսի ձևակերպումը ենթադրում է ներդրված ցիկլեր: Դրանցից արտաքին ցիկլի պարամետրը որոշում է [N;M] միջակայքի թիվը, ներքին ցիկլը որոշում է այն K թիվը, որի աստիճանները պետք է հաշվել: Սիա այս փաստը կարելի է պարզել, ինչպես վերը ասացինք, օգտվելով նախորդ խնդրում կիրառված եղանակից՝ բաժանելով նշված միջակայքի թիվը K–ի վրա, քանի դեռ բաժանման մնացորդը հավասար է զրոյի մինչև քանորդի ամբողջ մասը հավասարվի մեկի: Այսպիսով, երկրորդ ցիկլում ուրվագծեցինք երրորդ՝ ամենաներքին ցիկլը, որի ելքը կանխորոշում է նրան ընդգրկող ցիկլի հետագա ընթացքը:

Առաջին տարբերակով կառուցված Ա30.1 ալգորիթմի բլոկ–սխեման բերված է նկ.3.8–ում: Եթե արտաքին ցիկլը պետք է պարտադիր կատարվի (M–N+1) անգամ ստուգելով միջակայքի բոլոր B թվերը, ապա K պարամետրով 6…10 բլոկներից կազմված ներքին ցիկլի կառուցվածքը այլ է՝ այն երկու ելքով է: Բնական ելքը պայմանավորված է նրանով, որ միջակայքի B թիվը [2;P] միջակայքից ոչ մի K թվի աստիճան չի հանդիսանում, իսկ եթե հանդիպում է այդպիսի թիվ, ցիկլը ստիպողաբար է ընդհատում իր աշխատանքը, անցնելով թիվ 10 բլոկից դեպի թիվ 11 և այնուհետև՝ արտաքին ցիկլ:

Այստեղ K պարամետրի (աստիճանի հիմքի) վերին սահմանային արժեքը բնական է ընդունել հավասար այն–պիսի P թվի, որի քառակուսի աստիճանը չի գերազանցում B արժեքը, այսինքն՝ $P = \lfloor \sqrt{B} \rfloor$: Դրանից մեծ թվերը որպես հիմք ընդունելը անիմաստ է, այլապես կնշանակի ժամանակի կորուստ:



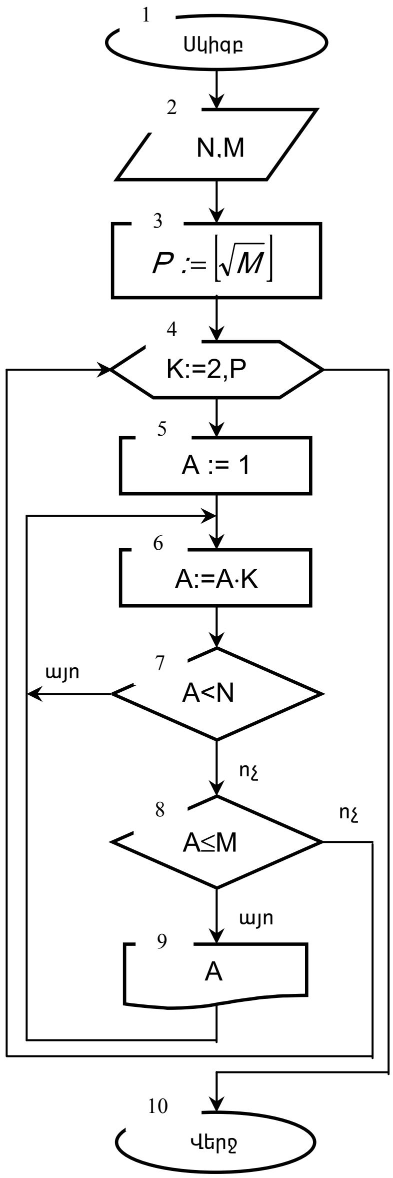
Նկ.3.8: Խնդիր 30 լուծման Ա30.1 ալգո–րիթմի բլոկ–սխեմա:

Գծանկարից երևում է, որ K–ի վրա բաժանումները իրականացնելու համար B–ի արժեքը կրկնօրինակվում է A փոփոխականում (բլ.7), որը և բաժանվում է ընտրած թվի վրա: Եթե բաժանման մնացորդը հավասար չէ զրոյի (բլ.8), ապա փորձում ենք բաժանել K–ի հաջորդ արժեքի վրա, ավելացնելով ընթացիկ արժեքին մեկ միավոր (անցում թիվ 6 բլոկին): Օրինակ, A=40 և K=2 դեպքում ամենաներքին ցիկլը կկատարվի երեք անգամ. յուրաքանչյուր քայլում թիվ 9 բլոկի միջոցով A–ի արժեքը նվազում է 2 անգամ և մենք ստանում ենք սկզբնական 40–ից 20, 10 և 5 արժեքները: Վերջին՝ 5 արժեքը անմնացորդ 2–ի չի բաժանվում՝ այդ պատճառով ամենաներքին ցիկլը ընդհատում է իր աշխատանքը և վերադառնում իրեն ընդգրկող ցիկլ՝ K–ին մեկ ավելացնելու նպատակով: Իսկ A=27 և K=3 դեպքում ամենաներքին ցիկլը նույնպես կկատարվի երեք անգամ, սակայն A–ի 27, 9 և 3 հաջորդական արժեքներից վերջինը 3–ի անմնացորդ բաժանվելու հետեվանքով նվազում է երեք անգամ, ստանալով A=1 արժեքը, որի պատճառով ամենաներքին ցիկլը ավարտում է իր աշխատանքը բնական ձևով, արտածվում 3–ի աստիճան հանդիսացող B արժեքը և ոչ թե վերադառնում է իրեն ընդգրկող ցիկլ, այլ անմիջապես արտաքին ցիկլ, քանի որ ցանկացած թիվ կարող է միայն մեկ թվի աստիճան լինել:

Այժմ կառուցենք նույն խնդրի լուծման այլ ալգորիթմ՝ հիմնվելով կիրառված եղանակի լրիվ հակադարձ սկըզ–բունքի վրա: Այն է. [2;P] հատվածից յուրաքանչյուր K թվի, որպես հիմքի համար պարզենք նրա որևէ աստիճանի կամ աստիճանների պատկանելությունը [N;M] միջակայքին: Չէ՞ որ միևնույն հիմքով տարբեր աստիճաններ կարող են հայտնվել նույն միջակայքում: Օրինակ, [100;300] հատ-վածում են գտնվում $2^7=128$ և $2^8=256$ թվերը: Իսկ ամե-նափոքր բնական թիվը, որի քառակուսին գերազանցում է վերին սահմանը, 18–ն է, քանի որ $18^2>300$, իսկ $17^2<300$: Հետևաբար, տվյալ միջակայքի համար հիմքերից մեծագույնը կարող է լինել $P=\lfloor\sqrt{300}\rfloor\approx\lfloor 17.32\rfloor=17$ թիվը:

Այսպիսով, Ա30.2 ալգորիթմը կարելի է ձևակերպել հետևյալ կերպ. ***փոփոխելով հիմքի արժեքը [2;P] տիրույթում մեկ քայլով, պարզել, թե որի աստիճաններն են պատկանում [N;M] միջակայքին, արտածելով նրանց արժեքները:*** Ինչպես տեսնում եք, տվյալ ձևակերպումը ենթադրում է երկու ներդրված ցիկլեր. արտաքին ցիկլը որոշում է K հիմքը, իսկ ներքինը հաշվում և ստուգում, թե ընտրած հիմքով որ աստիճաններն են պատկանում [N;M] միջակայքին: Ի տարբերություն նախորդ ալգորիթմի, այս տարբերակում հիմքերի մեծագույն արժեքը, հավանաբար անհրաժեշտ է ընդունել՝ $P=\lfloor\sqrt{M}\rfloor$:

Ա30.2 ալգորիթմի բլոկ–սխեման բերված է նկ.3.9–ում: Այստեղ K հիմքով A աստիճանները հաշվվում են ռեկուրենտ բանաձևով՝ 6...9 բլոկներից կազմված ներքին ցիկլում: Մինչև K–ի նվազագույն աստիճանի՝ [N;M] միջակայքում հայտնվելը գործում են ներքին ցիկլի 6 և 7 բլոկները: Հենց որ նվազա-



Նկ.3.9: Խնդիր 30 լուծման Ա30.2 ալգորիթմի բլոկ–սխեմա:

գույն աստիճանը հայտնվում է միջակայքում, այն արտածվում է (բլ.9) և շարունակում բազմապատկվել K–ով (բլ.6)՝ ավելի բարձր աստիճանները ստուգելու նպատակով: Եթե հաջորդ աստիճանները նույնպես գտնվում են միջակայքում, ապա գործում են ներքին ցիկլի բոլոր բլոկները: Երբ աստիճանի արժեքը գերազանցում է միջակայքի վերին սահմանը, ներքին ցիկլից վերադառնում ենք արտաքին ցիկլ՝ հիմքի արժեքը մեկով ավելացնելու համար:

Եթե այժմ համեմատենք խնդրի լուծման երկու տարբերակները, ապա, անկասկած, առավելությունը կլինի վերջինի կողմը: Նախ այն պատճառով, որ երկրորդ տարբերակը իրականացվում է ոչ թե երեք, այլ երկու ներդրված ցիկլերով: Հաջորդ գործոնը թվերի ընտրության հերթականությունն է, այն է. եթե նախորդ ալգորիթմում քննարկվում էին [N;M] հատվածից **բոլոր** թվերը, ապա վերջինում մենք ընտրում ենք միայն այն թվերը, որոնք որևէ թվի աստիճան են:

Ստորև առաջարկվող խնդիրները լուծելիս, կամա թե ակամա դուք ստիպված կլինեք վերանայել վերը բերված խնդիրների լուծման ալգորիթմները, կրկին վերլուծելով դրանց, գուցե և ավելի հետաքրքիր ալգորիթմներ առաջարկելով: Միայն այսպիսի ստեղծագործական մոտեցումը կտա ցանկալի արդյունքը:

ՎԱՐԺՈՒԹՅՈՒՆՆԵՐ

1. Երեքպարամետր $y=g(a,p,t)$ ֆունկցիան անընդհատ է պարամետրերի համապատասխան արժեքների համար ($a\in[a_1,a_2]$, $p\in[p_1,p_2]$, $t\in[t_1,t_2]$): Տրված B և Y մեծություն–ների համար որոշել և արտածել t պարամետրի այն արժեքները, որոնց դեպքում $a+p \le B$ և $y < Y$:

2...5 խնդիրներում տառային պարամետրերի կամա-յական թվային արժեքների համար հաշվել և արտածել հետևյալ արտահայտությունների արժեքները.

2. $Y = \sum_{i=n}^{m} i^i$:

3. $Y = \sum_{i=1}^{n}\left[(i+1)\prod_{j=1}^{i}\left(i+j^2\right)\right]$:

4. $S = \sum_{i=1}^{n}\frac{i^i x^i}{i!}$:

5. $S = \frac{1}{n!}\sum_{k=1}^{n}(-1)^k\left(2k^2+1\right)!$:

6. Որոշել տրված N բնական թվի բոլոր պարզ բաժանա–րարները:

7. Տրված են N և M բնական թվերը: Ստանալ N–ից փոքր բոլոր այն բնական թվերը, որոնց թվանշանների քառա-կուսիների գումարը հավասար է M–ի:

8. Բնական թիվը կոչվում է **կատարյալ**, եթե այն հավասար է իր բոլոր բաժանարարների, բացառությամբ իրեն, գու–մարին: Թիվ 6-ը կատարյալ է, որովհետև 6=1+2+3, իսկ թիվ 8–ը կատարյալ չէ, որովհետև $8\neq1+2+4$:

Որոշել տրված N բնական թվից փոքր բոլոր կատա–րյալ թվերը:

9. Տրված է N բնական թիվը: Որոշել և արտածել այդ թիվը չգերազանցող բնական թվերի բոլոր պյութագորյան եռ–յակները, այսինքն այնպիսի a, b, c եռյակներ, որոնց համար կատարվի $a^2+b^2=c^2$ ($a\le b\le c\le N$) պայմանը:

10. N թվանշաններից բաղկացած բնական թիվը հանդիսանում է Արմստրոնգի թիվ, եթե իր թվանշանների N–րդ աստիճանների գումարը հավասար է նույն թվին, ինչպես, օրինակ՝ $153=1^3+5^3+3^3$: Որոշել և արտածել երկու, երեք և չորս թվանշաններից կազմված Արմստրոնգի թվերը:

4. ՏՎՅԱԼՆԵՐԻ ԶԱՆԳՎԱԾՆԵՐ

Բազմաթիվ օրինակներով մենք ցուցադրեցինք գրեթե բոլոր տիպային ալգորիթմների կառուցման սկզբունքները: Ցանկացած նոր խնդիր լուծելիս դուք կարող եք օգտվել արդեն ներկայացված ալգորիթմներից, վերածնելով դրանք, կամ ’հավաքելով’ նրանցից, որպես պատրաստի մոդուլներից, նոր ալգորիթմներ: Մնում է միայն ճիշտ կողմնորոշվել ընտրության և ձևափոխման հարցերում:

Որպես կանոն, դիտարկված խնդիրներում մշակման ենթակա օբյեկտները պարզ փոփոխականներ էին, այսինքն՝ փոփոխականներ, որոնց անունները ինդեքսավորված չեն: Այնուամենայնիվ մեկ թե երկու անգամ մենք ստիպված էինք դիմել ինդեքսով փոփոխականներին՝ մի խումբ թվեր ժամանակավոր հիշելու միտումով (խնդիր 24): Տարրերի անորոշ քանակը մեզ ստիպեց հրաժարվել անունների տարբերակումից և կիրառել միևնույն անունը, բայց ինդեքսով, որն ամբողջ թիվ է և ցույց է տալիս համապատասխան տարրի տեղը տվյալների շարքում:

Մաթեմատիկայում տարբեր համարներով միևնույն անուն կրող նույնատիպ տվյալների խումբը կոչվում է **զանգված**: Այսպես, a, b, c, d անվանումներով իրական տիպի թվերը զանգված չեն կազմում, որովհետև դրանց անունները տարբեր են: Զանգվածում ընդգրկելու համար անհրաժեշտ է այդ անունները նույնացնել և համարա–կալել, օրինակ, (x_1,x_2,x_3,x_4), որտեղ a-ն զանգվածի առաջին տարրն է (x_1), b-ն՝ երկրորդ (x_2), c-ն՝ երրորդ (x_3) և d-ն՝ չորրորդ (x_4): Այստեղ x_i–ն ինդեքսով փոփոխական է, որի ինդեքսի արժեքը միանշանակ որոշում է տվյալի դիրքը հաջորդականությունում: Մնում է միայն ժամանակին և ճիշտ հասցեներով գրանցել տվյալները, որպեսզի անհրաժեշտության դեպքում կարողանանք դրանց օգտագործել: Եթե հիշենք Ա24 ալգորիթմը, ապա նրանում բնական թվի թվանշանները, սկսած միավորներից, հաջորդաբար ’արտաքըսվում՝ էին թվից և գրանցվում A զանգվածում, որպեսզի հաջորդ փուլում այդ թվանշանները օգտագործվեին իրենց ՚արտաքսմանը՚ հակառակ հերթականությամբ: Եթե նախկի-նում որևէ փոփոխականն էր հանդիսանում ցիկլի պարամետր, ապա զանգվածների մշակման խնդիրներում, ինչպես համոզվեցիք, ցիկլի պարամետրի դերում հանդես է գալիս ինդեքսը:

Վերը դիտարկված զանգվածները կոչվում են *միաչափ*, քանի որ զանգվածի տարրերի դիրքը որոշվում է մեկ ինդեքսի արժեքով: Մաթեմատիկայում միաչափ զանգված–ները այլ կերպ անվանում են **վեկտոր**, իսկ այն փաստը, որ տրված է **n** տարր պարունակող **X** վեկտոր, ներկայացվում է այսպես. $X=\left(x_i\right),\ \ i=\overline{1,n}$, այսինքն ինդեքսը կարող է ընդու-նել $1,2,\dots,n$ արժեքները:

Հաջորդ տեսակի զանգվածը, որը գործնական լայն կիրառում ունի, դա *երկչափ* զանգվածն է (**մատրից**), որտեղ յուրաքանչյուր տարր բնորոշվում է երկու ինդեքսներով, իսկ տարրերի փոխադարձ դիրքը հետևյալն է.

$$B=\begin{pmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1m} \\ b_{2,1} & b_{2,2} & \cdots & b_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ b_{n1} & b_{n2} & \cdots & b_{nm} \end{pmatrix} \quad \text{կամ} \quad B=\left\|b_{ij}\right\|,\ \ i=\overline{1,n};\ \ j=\overline{1,m}\ :$$

Ընդունված է որպես առաջին ինդեքս օգտագործել տողի համարը, իսկ որպես երկրորդ ինդեքս՝ սյան համարը: Այսպիսով $b_{3,5}$ կնշանակի երկչափ մատրիցի այն տարրը, որը գտնվում է 3-րդ տողի և 5-րդ սյան հատման կետում:

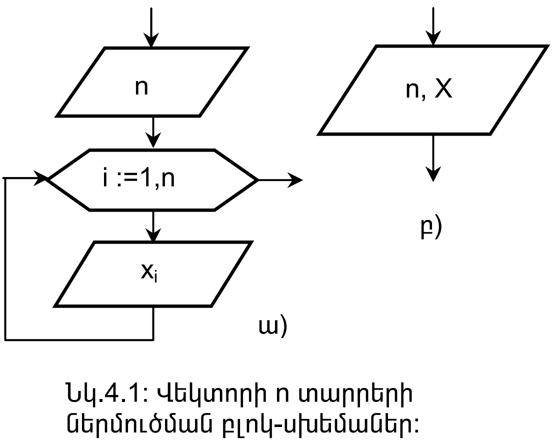
Առանձնահատուկ տեղ են զբաղեցնում քառակուսի մատրիցները, որոնց տողերի քանակը համընկնում է սյու–ների քանակի հետ (n=m): Նման մատրիցների յուրա–հատկությունն այն է, որ նրանք պարունակում են, այսպես կոչված, *գլխավոր* և *օժանդակ* անկյունագծեր: Ընդ որում, գլխավոր անկյունագիծը ընդգրկում է մատրիցի վերևի ձախ և ներքևի աջ անկյունները միացնող անկյունագծի վրա տեղադրված տարրերը, որոնց երկու ինդեքսները հավասար են միմյանց: Օժանդակ անկյունագիծը ընդգրկում է մատրիցի վերևի աջ և ներքևի ձախ անկյունները միացնող անկյունագծի վրա տեղադրված տարրերը, որոնց երկու ինդեքսները կապված են հետևյալ բանաձևով. $j=n-i+1$, որտեղ i-ն տողի համարն է, իսկ j-ն՝ սյան:

Սկզբունքորեն կարելի է ստեղծել եռա-, քառա- և այլ չափանի զանգվածներ, որոնք նույնպես կոչվում են մատրիցներ, ավելացնելով փոփոխականին կից երրորդ, չորրորդ և այլ ինդեքսներ: Եթե եռաչափ մատրիցը դեռ կարելի է պատկերացնել և գծագրել, որպես երկչափ մատրիցների հաջորդականություն, որտեղ երրորդ ինդեքսը որոշում է այդ մատրիցների տարածական դիրքը, ապա քառաչափ և ավելի բարձր չափանի մատրիցների պատկերումը մնում է մեր երևակայության խնդիր:

Այս բաժնում մենք կձևափոխենք նախկինում դիտարկված խնդիրների մեծամասնության լուծման ալգորիթմները, կիրառելով զանգվածներ՝ սկզբից միաչափ, հետո երկչափ: Կփորձենք նաև ընդհանրացնել դրանք և անհրաժեշտության դեպքում առաջարկել լուծման նոր եղանակներ:

4.1. ՄԻԱՉԱՓ ԶԱՆԳՎԱԾՆԵՐԻ ՄՇԱԿՈՒՄ

Մինչև խնդիրներին անցնելը պարզենք զանգվածի տարրերի ներմուծման հարցը: Չէ՞ որ ներմուծման են ենթակա բազմաթիվ տվյալներ, որոնք առանձին-առանձին ներկայացնել հնարավոր չէ: Սակայն, հաշվի առնելով այն հանգամանքը, որ տվյալները ներմուծվում են հաջորդաբար՝ մեկը մյուսից հետո, X զանգվածի ներմուծման գործընթացը կարելի է նկարագրել ցիկլով, ինչպես ներկայացված է Նկ.4.1ա-ում (հուսով ենք մեկնաբանության կարիք չկա):



Նկ.4.1: Վեկտորի n տարրերի ներմուծման բլոկ-սխեմաներ:

Հետագայում զանգվածների կիրարկմամբ գրեթե բոլոր խնդիրների ալգորիթմները կառուցելիս մենք առընչվելու ենք զանգվածների տարրերի ներմուծման հետ և ամեն անգամ բլոկ-սխեմաներում կիրառել ա) տարբերակը նպատակահարմար չէ: Հաշվի առնելով ալգորիթմների բարդանալու և դրա հետևանքով բլոկ-սխեմաների ավելի ընդարձակվելու միտումը, զանգվածների ներմուծման բ) տարբերակով ներկայացնելը չի նսեմացնի ալգորիթմի արժեքը: Առանձին դեպքերում, երբ ներմուծումը կհամատեղվի այլ գործընթացի հետ, մենք կրկին կդիմենք ա) տարբերակին՝ ցիկլերի քանակը կրճատելու նպատակով:

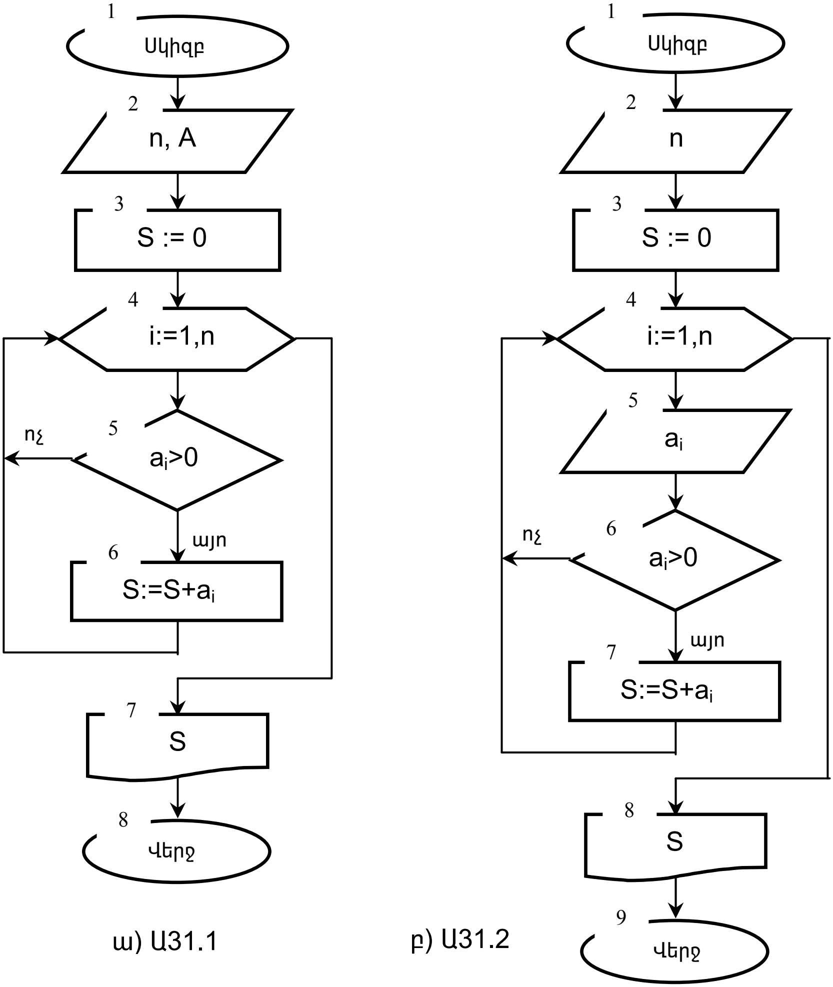
<u>Խնդիր 31:</u> Տրված է n իրական թվեր պարունակող A զանգված: Հաշվել զանգվածի դրական տարրերի գումարը:

Ինչպես վերը նշել ենք, A վեկտորի տարրերը նշանակվում են հետևյալ կերպ. $a_1, a_2, \ldots, a_n$: Ենթադրվում է, որ զանգվածի տարրերի արժեքները կամայական իրական թվեր են, որոնք նախապես ներմուծվելու են հաջորդաբար, զբաղեցնելով իրենց համար հատկացված 1-ից n տեղերը: Չիմանալով տարրերի արժեքները՝ մեզ մնում է դիմել նրանց համարով. a_i և փոփոխել ինդեքսի արժեքը 1-ից n ($i=\overline{1,n}$): Մնացած գործողությունները հայտնի են, փոխվել է միայն առարկայի ընտրության եղանակը: Նայելով նկ. 4.2ա-ին, որտեղ ներկայացված է ալգորիթմի բլոկ-սխեման, դժվար չէ պատկերացնել հետագա գործընթացը:

Վերլուծելով գծանկարում բերված ալգորիթմը, կարող ենք վկայել, որ այն բավարարում է ալգորիթմի կառուցման դասական կանոններին՝ սկզբում ներմուծում, ապա հաշվարկ և արտածում: Թվում է, թե այն լիովին պետք է մեզ բավարարի և հետագա վերամշակման ենթակա չէ: Սակայն հիշենք, որ ներմուծման թիվ 2 բլոկը ընդհանրացված է և իրականում n թվեր ներմուծող ցիկլ է: Ստացվում է, որ մենք կիրառում ենք նույն քայլով և կրկնությունների թվով երկու ցիկլեր. թիվ 2 բլոկում նախատեսված և 4...6 բլոկներից կազմված ցիկլերը: Հետևաբար կարելի է նրանց համատեղել, փոխարինելով երկու ցիկլերը մեկով, որտեղ ներմուծմանը զուգահեռ թիվը ստուգվում է և անմիջապես կատարվում են անհրաժեշտ գործողությունները: Ալգորիթմի երկրորդ՝ Ա31.2 տարբերակը բերված է նկ.4.2բ-ում:

Ինչպես տեսնում եք, ցիկլի ընդամենը մեկ բլոկով համալրումը հանգեցնում է ժամանակի զգալի տնտեսման, ինչը էական է դառնում n-ի մեծ արժեքների դեպքում:

Տեսնենք, թե ճնչին չափով փոխելով տվյալ խնդրի պայմանը, ինչպե՞ս կձևափոխվի ալգորիթմը: Դրա պատասխանը կստանաք, ծանոթանալով հաջորդ խնդրի լուծմանը:

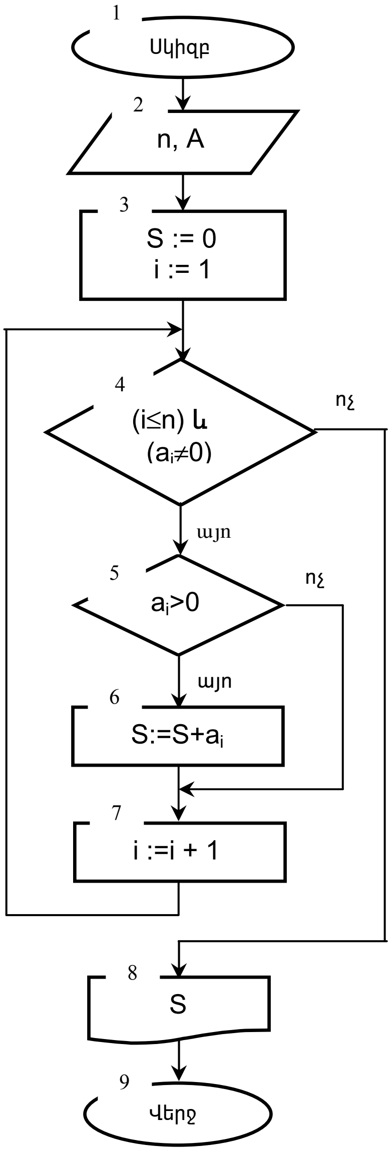


ա) Ա31.1 բ) Ա31.2

Նկ.4.2. Խնդիր 31 լուծման ալգորիթմների
երկու տարբերակների բլոկ-սխեմաներ

<u>Խնդիր 32:</u> Տրված է n իրական թվեր պարունակող A զանգված: Հաշվել զանգվածի այն դրական տարրերի գումարը, որոնք տեղադրված են մինչև առաջին զրոն: Եթե զանգվածը զրո արժեքով տարր չի պարունակում, հաշվել վեկտորի բոլոր դրական տարրերի գումարը:

Նկ.4.3-ում պատկերված այս խնդրի լուծման ալգորիթմի բլոկ-սխեմայից երևում է, թե ինչ փոփոխությունների է հանգեցնում *բոլոր* բառի փոխարինումը *մինչև* բառով: Չէ՞ որ առաջինը ենթադրում է 1-ից n բոլոր տարրերի քննարկումը, իսկ երկրորդը՝ ոչ բոլոր, միայն մինչև զրո հանդիպելը: Զրոյի բացակայության դեպքում ստիպված ենք դիտարկել բոլոր n տարրերը:



Նկ.4.3.Խնդիր 32 լուծման ալգորիթմի բլոկ-սխեմա:

Այսպիսով, գումարման գործընթացը կարելի է ներկայացնել հետևյալ կերպ. *առաջին տարրից սկսած քանի դեռ (i≤n) և (a_i≠0), գումարել միայն դրական թվերը:* Այսինքն ցիկլի կրկնության պայմանները երկուսն են, և որևէ մեկի խախտման դեպքում ցիկլի աշխատանքը պետք է ընդհատել:

Այստեղ մենք առաջին անգամ պայմանի ստուգման բլոկում (բլ.4) ներկայացրեցինք երկու առնչություններ, կցված 'և' շաղկապով: Նման բան թույլ է տրվում, եթե առնչությունների երկու հընարավոր շարունակությունները համընկնում են և ընդհանուր պայմանական արտահայտությունը տեղավորվում է շեղանկյան սահմաններում:

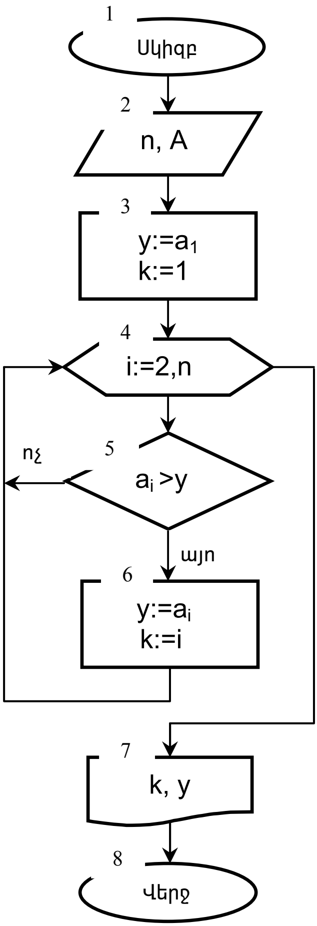
Այժմ պատկերացրեք, թե ինչպես կձևափոխվի ալգորիթմը, եթե խնդրի ձևակերպման մեջ փոխենք երկրորդ պայմանը, նշելով. *եթե զանգվածը զրո արժեքով տարր չի պարունակում, ոչինչ չհաշվել:* Այս դրվածքով խնդրի լուծումը, հավանաբար, կներկայացվի երկու փուլով կամ երկու անկախ ցիկլերով. առաջինը պետք է պարզի զրոյի առկայությունը վեկտորի կազմում և նրա համարը, եթե այն գտնվի, իսկ երկրորդը անհրաժեշտության դեպքում պետք է հաշվի մինչև որոշած համարը տեղադրված դրական թվերի գումարը: Այսպիսով, վերջին դեպքում զրոյի ստուգման և գումարի հաշվման համատեղումը դառնում է անիմաստ, քանի որ զրոյի բացակայության դեպքում հաշվված գումարը պետք չէ: Խնդրի վերջին տարբերակի լուծման ալգորիթմի կառուցումը թողնում ենք ձեզ, որի ընթացքում դուք մեկ անգամ ևս կհամոզվեք, որ յուրաքանչյուր խնդիր լուծվում է յուրովի, ինչի պատճառով դրանք պետք է լուծվեն ստեղծագործաբար:

Հաջորդ խնդիրը, որն առաջարկում ենք քննարկել և լուծել, միաչափ զանգվածի տարրերից մեծագույնի (նըվազագույնի) որոշման խնդիրն է: Մենք կառաջարկենք երկու տարբեր ալգորիթմներ, որոնցից առաջինը կարելի է համարել նախկինում դիտարկված երեք թվերից մեծագույնի (նվազագույնի) որոշման խնդրի ընդհանրացումը, իսկ երկրորդը՝ ավելի բնական եղանակի իրականացում: Այսպիսով, առաջարկվող խնդիրն է.

<u>Խնդիր 33:</u> Տրված է n թվեր պարունակող A զանգվածը: Որոշել ամենամեծ արժեք ունեցող տարրի համարը:

Անդրադառնանք խնդիր 5-ի լուծման ալգորիթմներին: Դիմելով Ա5.1 ալգորիթմին, կարող ենք անմիջապես նըկա–տել, որ այն պիտանի չէ ընդհանրացմանը, քանի որ հստա-կեցված չեն այն գործողությունները, որոնք պետք է կրկնվեն զանգվածի՝ տարրից տարր անցնելիս: Նույնը չենք կարող ասել Ա5.2 ալգորիթմի մասին: Այստեղ հստակ շարադրված է, որ յուրաքանչյուր փոփոխական համեմատվում է նախորդ թվերից մեծագույնի հետ, ինչի արդյունքում մեծագույն արժեքը կարող է փոխվել՝ ստանալով ավելի մեծ արժեք: Այժմ եթե պատկերացնեք, որ a, b, c անկախ անվանումների փոխարեն կիրառել ենք համարով տարբերվող՝ $a_1,a_2,a_3,\ldots,a_n$ անունները, ապա համոզված ենք, որ ձեր երևակայության մեջ առաջացել է նույն ցիկլի պատկերը, որը ներկայացված է նկ.4.4-ում:

Այստեղ նույնպես մինչև ցիկլի առաջին կատարումը y փոփոխականին վերագրվում է ցուցակով առաջին տարրի արժեքը (բլ.3), իսկ k փոփոխականին՝ ընտրած տարրի համարը, այսինքն մեկ: Որից հետո, սկսած երկրորդ տարրից,



Նկ.4.4. Խնդիր 33 լուծման Ա33.1 ալգոռիթմի բլոկ-սխեմա:

յուրաքանչյուր a_i տարր $(i=\overline{2,n})$ համեմատվում է ընտրած արժեքի հետ (բլ.5)։ Որ տարրի արժեքը կգերազանցի նախորդ թվերից ընտրված մեծագույնին, նրա արժեքը կընդունվի որպես ընթացիկ մեծագույն արժեք, իսկ նրա համարը կվերագրվի k փոփոխականին (բլ.6)։ Այսպիսով, 4...6 բլոկներից կազմված ցիկլի (n-1) կրկնությունների արդյունքում կստանանք զանգվածի բոլոր տարրերից մեծագույն արժեքը y-ում, իսկ նրա համարը՝ k-ում:

Սակայն կառուցած ալգորիթմը ավելի արհեստական է, քան բընական: Դրանում դուք կարող եք համոզվել, եթե փորձեք ինքներդ լուծել նման խնդիր՝ մեկնաբանելով կատարած յուրաքանչյուր քայլ:

Վերցնենք թվեր (n=5) և որոշենք դրանցից ամենամեծը: Ենթադրենք, որ այդ թվերն են.

$a_1=5$; $a_2=3$; $a_3=8$; $a_4=11$; $a_5=2$:

1) Առաջին քայլում, բնական է, համեմատվելու են առաջին երկու թվերը $a_2>a_1$ պայմանով: Համեմատության արդյունքն է՝ ՛**ոչ**՛, երեքը փոքր է հինգից:

2) Հետևելով բանականությանը, այս քայլում կհամեմատենք առաջին և երրորդ թվերը $a_3>a_1$ պայմանով, ինչի պատասխանը կլինի ՛**այո**՛, ութը մեծ է հինգից:

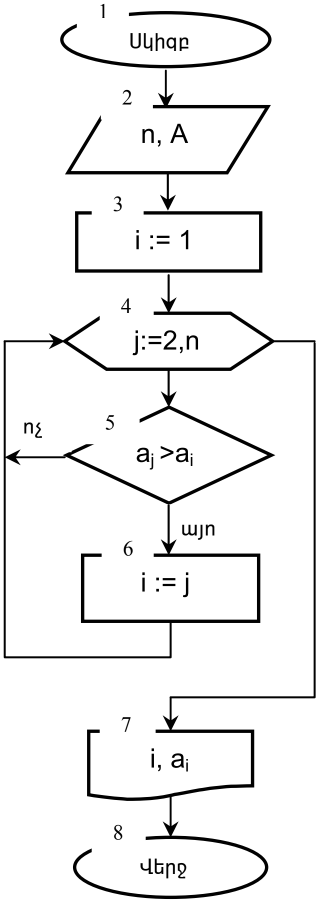
3) Այս քայլում չորրորդ թվի հետ համեմատվում է a_3-ը, որպես նախորդ թվերից մեծագույնը՝ $a_4>a_3$ պայմանով, որը կրկին առկա է և պատասխանն է՝ ՛**այո**՛:

4) Վերջապես, այս քայլում համեմատում ենք չորրորդ և հինգերորդ թվերը՝ $a_5>a_4$ պայմանով: Տեղադրելով փոփոխականների արժեքները տվյալ առնչությունում՝ համոզվում ենք, որ համեմատության արդյունքն է՝ ՛**ոչ**՛ պատասխանը և

a_4=11 արժեքը մնում է մեծագույն, ինչը համա– պատասխանում է իրականությանը:

Այստեղ որոշ ընթերցողներ կարող են տարակուսել, չնկատելով այն վճռորոշ օրինաչափությունները, որոնց հիման վրա հնարավոր է կառուցել ցիկլ: Սակայն ովքեր ընթերցում էին ուշադիր և վերլուծաբար, հավանաբար նկատեցին: Այն ավելի տեսանելի դարձնելու համար արտագրենք յուրաքանչյուր քայլում ստացած արդյունքները միասին, առանց մեկնաբանությունների: Կստանանք.

$a_2 > a_1$ → ’ոչ’
$a_3 > a_1$ → ’այո’
$a_4 > a_3$ → ’այո’
$a_5 > a_4$ → ’ոչ’



Նկ.4.5. Խնդիր 33 լուծման Ա33.2 ալ-գորիթմի բլոկ-սխեմա:

Հուսով ենք, նկատեցիք ին–դեքսների փոփոխման հետ կապված օրինաչափությունը: Եթե ոչ, ապա հուշենք, որ աղյուսակում բերված առնչությունների ձախ կողմում ինդեքսը, որը նշանակենք j-ով, փոփոխվում է 2-ից 5 (n), մնալով միշտ ավելի մեծ, քան աջակողմյան՝ i ինդեքսը, որը ընդունում է ձախակողմյան ին–դեքսի արժեքը, երբ առընչու–թյունը ճշմարիտ է: Հակառակ դեպքում նրա արժեքը մնում է անփոփոխ:

Իր իմաստով դա նշանակում է, որ *քանի դեռ a_1 (i=1) մնում է ավելի մեծ, քան իրեն հաջորդող թվերը, մեծագույն տարրի ինդեքսի արժեքը չի փոփոխվում: Հենց որ հանդիպում ենք իրենից մեծ՝ a_j թիվ, մեծագույնի ինդեքսը հավասարեցնում ենք j–ին* *(i:=j), ինչը կնշանակի, որ a_j թիվը իրեն նախորդող թվերից մեծագույնն է: Տվյալ պահից սկսած՝ իրեն հաջորդող տարրերը կհամեմատվեն արդեն a_j թվի հետ, և ոչ թե a_1-ի:* Այսպես շարունակելով՝ i ինդեքսի վերջնական արժեքը ցույց կտա զանգվածի մեծագույն տարրի համարը: Տվյալ պարբերությունում բերված շարադրությունը ընդունենք որպես նկ.4.5-ում ներկայացված խնդիր 33-ի Ա33.2 ալգորիթմի նկարագրություն:

Ալգորիթմի երկրորդ տարբերակի առավելությունը առաջինի նկատմամբ ակնհայտ է, թեկուզ այն պատճառով, որ մեծագույն տարրի համարը որոշելու նպատակով լրա–ցուցիչ փոփոխականի կիրառման և նրա հետ կապված ավելորդ գործողությունների կատարման հարկ չկա: Այս դեպքում, որոշելով մեծագույն տարրի համարը, մենք միաժամանակ անուղղակի կերպով որոշում ենք և նրա արժեքը:

Վերջին օրինակով տեսանք, որ բնական վարվելաձևի ալգորիթմի վերածելը գերադասելի է: Սակայն միշտ չէ, որ դա այդպես է: Յուրաքանչյուր անգամ պետք է ընդունել իրավիճակին համապատասխանող լավագույն որոշում, ինչպես հաջորդ օրինակում:

<u>Խնդիր 34:</u> Տրված է n թվերից բաղկացած X զանգվա–ծը: Պահանջվում է զանգվածից հեռացնել զրո արժեք ունեցող բոլոր տարրերը:

Եթե դիմենք բնական վարվելաձևի, ապա տրված թվերը նույն հերթականությամբ, բայց առանց զրոների, պետք է հաջորդաբար արտագրենք նոր տողում, ինչպես ցուցա–դրրված է նկ.4.6-ում: Ալգորիթմների լեզվով դա համարժեք է՝ X վեկտորի տարրերի առանց զրոների արտագրմանը ոմն Y զանգվածի մեջ: Բերված օրինակում նկատելի է, որ X-ի տարրերի ինդեքսների արժեքները համընկնում են նույն տարրի Y-ում զբաղեցրած դիրքի հետ միայն մինչև առաջին զրոն: Որից հետո նախկին և նոր զբաղեցրած դիրքերի միջև առաջանում է տարբերություն, ինչը բացատրվում է զրոյին հավասար տարրերի հեռացումով:

Եթե X վեկտորի տարրերի ինդեքսը նշանակենք i-ով, իսկ Y վեկտորինը՝ j-ով, ապա այն դեպքում, երբ i-ն ընդունում է 1-ից n հաջորդական արժեքներ, j-ն աճում է միայն այն դեպքում, երբ առաջանում է զանգվածից զանգված տարրի արտագրման անհրաժեշտություն: Գործընթացն ավարտելիս j ինդեքսի արժեքը ցույց կտա Y զանգվածում գրանցված

$X -\ x_1,\ x_2,\ 0,\ x_4,\ 0,\ 0,\ x_7,\ 0$

$Y -\ y_1,\ y_2,\ y_3,\ y_4$

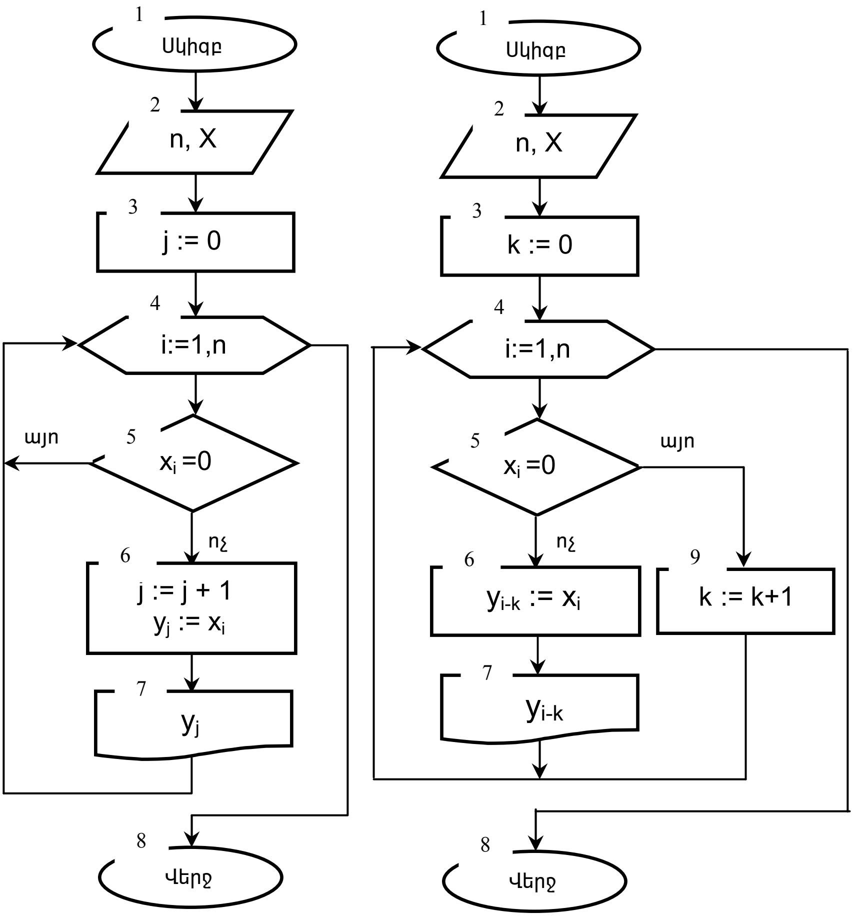
Նկ.4.6. Լրացուցիչ զանգվածի օգտագործմամբ զրոների հեռացման գործընթացի սխեմա:

տարրերի քանակը: Տվյալ սխեմանն իրականացնող ալգորիթմի բլոկ-սխեման բերված է նկ. 4.7ա-ում:

Միգուցե մենք շտապեցինք, Y վեկտորի տարրերի համար կիրառելով առանձին j ինդեքս, այն դեպքում, երբ 4.6 սխեմայի ավելի ուշադիր զննման արդյունքում կարող էինք նկատել, որ x_i տարրի նոր դիրքը նվազ է i դիրքից մինչև այդ տարրը հեռացված զրոների քանակով: Իրոք, x_4 արտագրվում է 3-րդ դիրք, x_7՝ 4-րդ և այլն: Այս տարբերակը իրականացնելու համար անհրաժեշտ է կիրառել հեռացված զրոները հաշվող հաշվիչ: Նկ. 4.7բ-ում նման հաշվիչը նշանակված է k-ով, որի նախնական արժեքն է զրոն: Իսկ տվյալի նոր դիրքը նախորդի համեմատ որոշվում է (i-k) արտահայտությամբ: Եթե, օրինակ, X զանգվածը զրո արժեքով տարր չպարունակի, ապա հաշվիչի արժեքը կմնա հավասար զրոյի և X-ի պարունակությունը ամբողջությամբ կարտագրվի Y զանգվածի մեջ՝ զբաղեցնելով նույն տեղերը:

Պետք է խոստովանենք, որ երկու տարբերակներն էլ բերված են միտումնավոր՝ առանց շտապելու կամ ուշացնելու, քանի որ նրանցում իրականացված երկու սկզբունքները գործնականում հաճախակի են կիրառվում:

Այժմ պատկերացրեք, որ դուք զինված եք մատիտով ու ռետինով, իսկ խնդիրը պահանջում է նույն զրոները X զանգվածից հեռացնել առանց նոր տող զբաղեցնելու: Բնական է, որ դուք կվարվեք երկրորդ ալգորիթմում իրա–կանացված եղանակով, միայն թե x_i-ն արտագրելով ոչ թե Y զանգվածում, այլ նույն X-ում՝ (i-k) տեղում, ջնջելով այնտեղ գրված թիվը և գրանցելով նորը: Մնում է վերջում ավելացնել n:=n-k գործողությունը, հաշվի առնելով տար–րերի քանակի պակասեցման փաստը: Տվյալ եղանակի նկարագրությունը թողնելով ձեզ, ավելացնենք, որ զանգ–վածից նման ձևով թվերի հեռացումը կոչում են զանգվածի ’**սեղմում**’:



ա) Ա34.1 բ) Ա34.2

Նկ.4.7. Խնդիր 34 լուծման ալգորիթմների երկու տարբերակներ:

Զանգվածների կիրարկմամբ թվերի վերաբաշխումով ալգորիթմներ գործնականում հաճախ են հանդիպում, չնայած զանգվածից զանգված տարրերի արտագրելը ավելի հեշտ է ու արագ: Վերաբաշխման անհրաժեշտությունը առաջանում է ազատ տեղի սղության կամ բացակայության պատճառով: Այս պրոբլեմը հիմնականում կապված է համակարգիչների հիշողության սահմանափակ ծավալների հետ, երբ ստիպված ենք հաշվառել յուրաքանչյուր տվյալին հատկացվող բջիջների քանակը: Մյուս կողմից վերաբաշխման ալգորիթմները առանձին հետաքրքրություն են ներկայացնում իրենց յուրահատուկ տրամաբանությամբ:

Այսուհետև ենթադրենք, որ զանգվածներին հատկացվում են անհրաժեշտ քանակությամբ համարակալված վանդակներ, որտեղ յուրաքանչյուր վանդակում կարելի գրանցել մեկ տարր (թիվ): Տարրի համարը համընկնում է իր կողմից զբաղեցրած վանդակի համարի հետ: Այսպես, խոսելով x_i տարրի մասին, նկատի ունենք, որ այն զբաղեցնում է հաջորդական վանդակներից i-րդը: Թիվը վանդակից վանդակ արտագրելիս կրկնօրինակվում է, իսկ վերջին վանդակի նախկին պարունակությունը ջնջվում: Հետևաբար, եթե որևէ վանդակի պարունակություն ձեզ հետագայում պետք է, ապա մինչև այնտեղ նոր թիվ գրանցելը արտագրեք նրա պարունակությունը այլ վանդակի մեջ: Ինչպես հաջորդ օրինակում:

<u>Խնդիր 35:</u> Տրված են a և b փոփոխականների արժեքները: Կատարել նշված փոփոխականների արժեքների տեղափոխություն:

Առաջին հայացքից շատ պարզ թվացող այս խնդրի լուծումը այնուամենայնիվ պահանջում է որոշ մտավոր աշխատանք: Առաջին անգամ ծրագրավորման հետ առնչվող լսարանում անմիջապես առաջարկում են կատարել հետևյալ երկու գործողությունները. a:=b և b:=a: Սակայն բավական է հուշել, որ առաջին գործողության արդյունքում արտագրելով թիվը b-ին հատկացված վանդակից a-ի վանդակի մեջ մենք կորցնում ենք վերջինիս նախնական արժեքը, բոլորը միաբերան առաջ կքաշեն օժանդակ՝ z փոփոխականի (վանդակի) վարկածը, արդյունքում, ճիշտ լուծումը կարելի է ներկայացնել երեք գործողությունների միջոցով. z:=a; a:=b և b:=z:

Երկու թվերի տեղափոխման սկզբունքը մեզ պետք կգա հաջորդ խնդիրը լուծելիս:

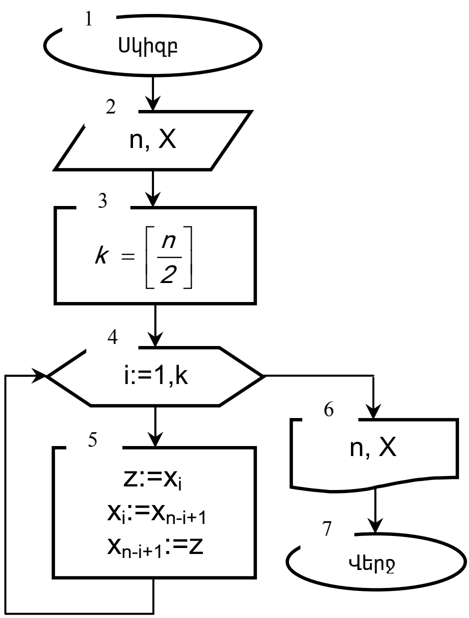
<u>Խնդիր 36:</u> Տրված է n տարր պարունակող X զանգվա–ծը: Վերդասավորել զանգվածի տարրերը հակադարձ հերթականությամբ:

Փորձենք տվյալ խնդրի լուծումը նկարագրել միայն թվարկելով այն տարրերի զույգերը, որոնց արժեքները պետք է տեղերով փոխանակել, առանց ավելորդ մեկնաբանությունների:

Բերված սխեմայից ակնհայտ է դառնում իրականացվող գործընթացի օրինաչափությունը, ինչը որոշում է համապատասխան ցիկլի i պարամետրի փոփոխման օրենքը: Մնում է ճշտել նրա վերին սահմանը: Հավանաբար այն պետք է ընթանա մինչև զանգվածի կենտրոնի թիվը, որի համարն է՝ $k = \left[\frac{n}{2}\right]$:

$x_1 \leftrightarrow x_n$
$x_2 \leftrightarrow x_{n-1}$
$x_3 \leftrightarrow x_{n-2}$
. . . .
$x_i \leftrightarrow x_{n-i+1}$

Նկ.4.8-ում ներկայացված է բերված գործընթացը իրականացնող ալգորիթմի բլոկ-սխեման, որը այլևս, հուսով ենք, մեկնաբանության կարիք չունի: Միգուցե որոշ տարակուսանք առաջացնի արտածման՝ թիվ 6 բլոկը: Սակայն ոչ մի նորույթ այն չի պարունակում, բացի այն, որ ներմուծման պես ընդհանրացված ձևով արտահայտում է X զանգվածի n քանակությամբ տարրերի արժեքների արտածում: Ձեզ նկատելի տեղի սղությունը լրիվ արդարացնում է կիրառված ընդհանրացումը:



Նկ.4.8. Խնդիր 36 լուծման ալգորիթմի բլոկ-սխեմա:

Զանգվածի մեծագույն (փոքրագույն) տարրի որոշման և տարրերի արժեքների տեղափոխման դիտարկված սկզբունքները ստորև կկիրառենք հաջորդ խնդիրը լուծելիս: Զանգվածի տարրերի՝ ըստ որևէ բնութագրի կարգավորումը գործնականում կիրառվող կարևորագույն խնդիրներից է: Այս խնդիրը իրականացնող ալգորիթմներն ունիվերսալ բնույթի են, քանի որ թույլ են տալիս կարգավորել ոչ միայն թվեր, այլ նաև բառեր և նախադասություններ: Բավական է նշել այբբենական կարգով անձնակազմի ցուցակի կարգավորման խնդիրը, կամ ցուցակի կարգավորումը ըստ անձերի տարիքի, աշխատած տարիների և այլն: Թվային մեծ զանգվածների կարգավորվածությունը թույլ է տալիս համապատասխան արագագործ ալգորիթմների շնորհիվ շատ արագ գտնել անհրաժեշտ թվերը: Ինչևէ, ստորև մենք կզբաղվենք տվյալների կարգավորման ալգորիթմների կառուցումով:

<u>Խնդիր 37:</u> Տրված է n տարրեր պարունակող թվային A զանգվածը: Կարգավորել զանգվածի տարրերն ըստ նրանց արժեքների նվազման:

Տվյալ խնդիրը նպատակահարմար է լուծել առանց լրացուցիչ զանգված օգտագործելու, քանի որ գործնա–կանում այդպիսի հնարավորություն կարող է չընձեռվել: Ավելին, դուք շուտով կհամոզվեք, որ միևնույն զանգվա–ծում տվյալների կարգավորումը ավելի արդյունավետ է:

Այս դեպքում էլ մենք կղեկավարվենք արդեն ավան–դույթ դարձած սկզբունքով, խնդրի լուծման համար առա–ջարկելով մեկից ավել ալգորիթմներ, որոնց հետագա կի–րառումը կախված է կոնկրետ իրավիճակից: Այս անգամ սկզբից կդիտարկենք կարգավորման գործընթացի 'վատ' տարբերակի ալգորիթմացումը, որից հետո՝ բնական և արհեստական: 'Վատ' ալգորիթմի ներկայացման ցանկու–թյունը առաջացել է նրա չհիմնավորված հաճախակի կի–րառման հետևանքով, ինչը անթույլատրելի է: Յակառակ դրան վերջին երկուսը կիրառելի են տարբեր պայման–ներում, ինչում դուք առիթ կունենաք համոզվելու տվյալ ձեռնարկի սահմաններում:

<u>Ալգորիթմ Ա37.1</u>

Տվյալ ալգորիթմի գաղափարը, հավանաբար, առաջա–ցել է առանց խորը դատելու և նրա իրագործողի հնարա–վորությունները հաշվի առնելու: Յամենայն դեպս, ծանո–թանալով այս ալգորիթմի հետ, համոզված ենք, ձեզանից ոչ ոք չի ստանձնի նման առաքելություն: Իսկ մտահղացումը հետևյալն է:

Առաջին քայլում a_1 տարրը հաջորդաբար համեմատ–վում է նրանից հետո տեղադրված բոլոր տարրերի հետ: Յանդիպելով ավելի մեծ արժեքով a_j տարրի, նրանց տե–ղերով փոխանակում ենք և շարունակում ենք երթը դեպի վերջին տարրը: Այս ընթացքում a_j տարրից հետո տեղա–

դրված տարրերը կհամեմատվեն նույն a_j տարրի հետ, որը հայտնվեց առաջին դիրքում: Նրանից մեծ արժեքով a_k-ն և առաջին տարրը կփոխանակենք տեղերով, որի հետևանքով a_j–ն կհայտնվի k–րդ դիրքում: Այսպես շարունակելով՝ առաջին տեղում կհայտնվի զանգվածի մեծագույն արժեքով տարրը:

Երկրորդ քայլում նույնը կատարում ենք երկրորդ դիր–քում հայտնված տարրի հետ: Այսինքն այն հաջորդաբար համեմատում ենք իրենից հետո տեղադրված բոլոր տար–րերի հետ: Յանդիպելով ավելի մեծ արժեքով a_j տարրը, նրանց փոխանակում ենք տեղերով և շարունակում ենք երթը դեպի վերջին տարրը: Այս ընթացքում a_j տարրից հետո տեղադրված տարրերը կհամեմատվեն նույն a_j տարրի հետ, որը հայտնվեց երկրորդ դիրքում: Նրանից մեծ արժեքով a_k–ն և երկրորդ տարրը կփոխանակենք տեղերով, որի հետևանքով a_j-ն կհայտնվի k–րդ դիրքում: Այսպես շարունակելով՝ երկրորդ տեղում կհայտնվի զանգվածի արժեքով երկրորդ մեծագույն տարրը:

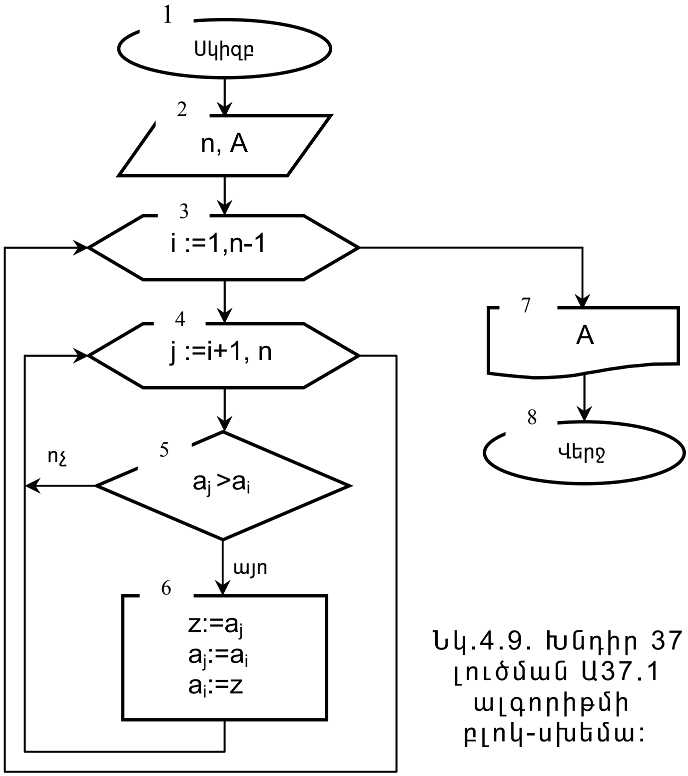
Նույնը կրկնելով հաջորդ տարրերի հետ՝ մենք քայլ առ քայլ կդասավորենք զանգվածի տարրերը ըստ նրանց արժեքների նվազման կարգի: Դրանում դուք կարող եք հա–մոզվել դիմելով նկ.4.9-ում բերված Ա37.1 ալգորիթմի բլոկ-սխեմային:

Պատկերացնու՞մ եք, թե յուրաքանչյուր i–րդ ցիկլում որքան տարրեր պետք է տեղերով փոխանակել, որպեսզի i համարից մինչև վերջին դիրքը տեղադրված տարրերից մեծագույնը հայտնվի i–րդ դիրքում: Գերադասում ենք այդ քանակը չհաշվել և առաջարկել հետևյալ մոտեցումը:

<u>Ալգորիթմ Ա37.2</u>

Ալգորիթմի աշխատանքի սկզբունքը հետևյալն է:

Առաջին քայլում որոշում ենք a_1-ից a_n թվերից մեծա–գույնը, որը փոխանակում ենք տեղերով a_1-ի հետ: Այսպի–սով մեծագույն արժեքով տարրը հայտնվում է առաջին տեղում: Իսկ ինչպե՞ս վարվել, եթե a_1-ն է մեծագույնը: Պարզապես ոչինչ չձեռնարկել, թողնելով այն նույն տե–ղում:



Նկ.4.9. Խնդիր 37 լուծման Ա37.1 ալգորիթմի բլոկ-սխեմա:

Երկրորդ քայլում որոշում ենք a_2-ից a_n թվերից մեծագույնը, որը փոխանակում ենք տեղերով a_2-ի հետ: Այսպիսով, արժեքով երկրորդ մեծագույն տարրը հայտնը–վում է երկրորդ տեղում: Եթե այս քայլում մեծագույնը լինի a_2-ը, ապա այն պետք է թողնել տեղում:

Այսպես շարունակելով՝ քայլ առ քայլ մենք դասավո–րում ենք զանգվածի թվերը մեծից փոքր: Վերջին՝ (n-1)-րդ քայլում կմնան երկու թվեր, որոնցից մեկը կզբաղեցնի նախավերջին՝ (n-1)-րդ տեղը, իսկ մյուսը՝ վերջին:

Ինչպես տեսանք, քայլերից յուրաքանչյուրում մենք պետք է որոշենք մեծագույնը՝ այդ քայլի համարով տարրից մինչև վերջինը: Քանի որ մեզ ավելի շատ հետաքրքրում է մեծագույն տարրի դիրքը, ապա կիրառենք վերը քննարկված Ա33.2 ալգորիթմը չնչին փոփոխություններով (տես նկ.4.5): Եթե քայլը հաշվող փոփոխականը անվանենք q-ով, ապա որպես ենթադրվող առաջին մեծագույն թիվ պետք է ընտրել ոչ թե a_1-ը, այլ a_q-ն (i:=q): Դրա հետ կապված՝ կփոխվի նաև մեծագույնի հետ համեմատվող առաջին տարրի համարը, որը հավասար կլինի (q+1)-ի: Այսպիսով, խնդիրը լուծող ալգորիթմը կներկայացվի ներդրված երկու ցիկլերով, որոնցից արտաքին ցիկլը պետք է հաշվի քայլերը, իսկ ներքինը որոշի ընթացիկ քայլին համապատասխանող մեծագույն տարրի համարը: Ա37.2 ալգորիթմի բլոկ-սխեման բերված է նկ.4.10-ում:

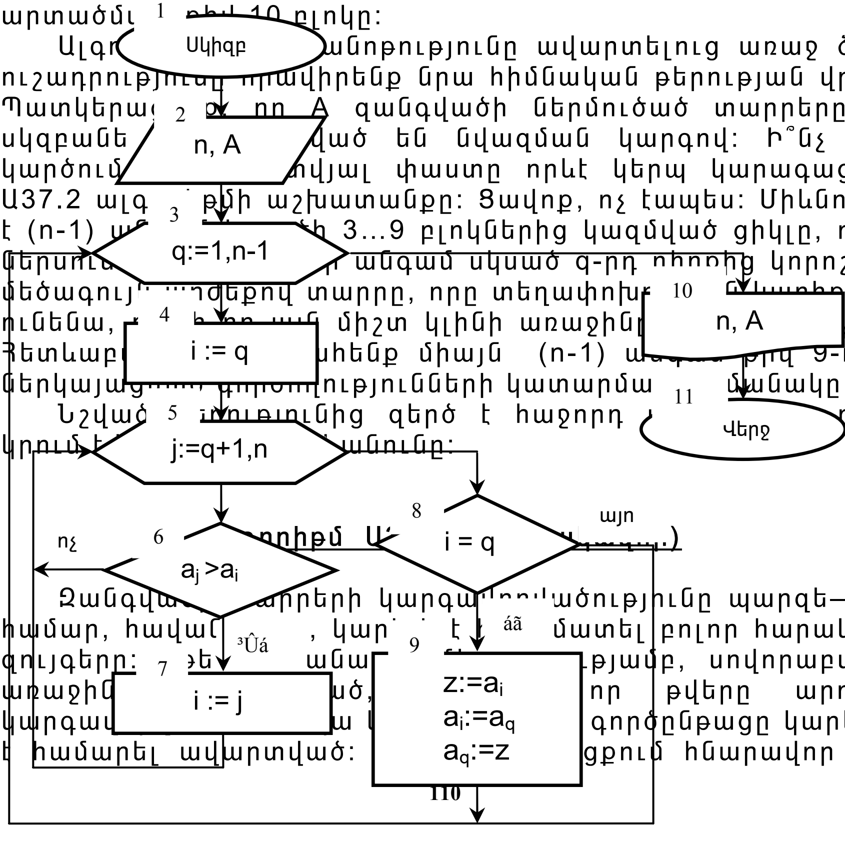
Այն մանրամասն նկարագրելու կարիք չկա, քանի որ մեծագույն տարրի արժեքը որոշող գծանկարի 4...7 բլոկներից կազմված ցիկլի աշխատանքը արդեն քննարկվել է թիվ 33 խնդիրը լուծելիս: Իսկ թիվ 8 և 9 բլոկների առկայությունը մեզ ապահովագրում է յուրաքանչյուր քայլում որոշվող մեծագույն տարրի ավելորդ տեղափոխությունից: Նկարագրված գործընթացը (n-1) անգամ կրկնելուց հետո A զանգվածի տարրերը կվերադասավորվեն արժեքների նվազման կարգով: Դրանում համոզվելու համար կիրառված է արտածման թիվ 10 բլոկը:

Ալգոր[illegible]անոթությունը ավարտելուց առաջ ձեր ուշադրությունը հրավիրենք նրա հիմնական թերության վրա: Պատկերաց[illegible] որ A զանգվածի ներմուծված տարրերը ի սկզբանե [illegible]ված են նվազման կարգով: Ի՞նչ եք կարծում[illegible] նշված փաստը որևէ կերպ կարագացնի Ա37.2 ալգ[illegible]թմի աշխատանքը: Ցավոք, ոչ էապես: Միևնույն է (n-1) ա[illegible] 3...9 բլոկներից կազմված ցիկլը, որի ներսու[illegible] անգամ սկսած q-րդ դիրքից կորոշվի մեծագույ[illegible]րժեքով տարրը, որը տեղափոխ[illegible] չի ունենա, [illegible] այն միշտ կլինի առաջինը[illegible]-ն: Հետևաբ[illegible]հենք միայն (n-1) [illegible] 9-ում ներկայաց[illegible] գործողությունների կատարմա[illegible]մանակը:

Նշված [illegible]ությունից զերծ է հաջորդ [illegible] որը կրում է [illegible]անունը:

Ալգորիթմ Ա[illegible])

Զանգված[illegible]արրերի կարգա[illegible]ածությունը պարզելու համար, հավա[illegible], կար[illegible] է [illegible] մատել բոլոր հարակից զույգերը: [illegible]ե[illegible] անա[illegible]թյամբ, սովորաբար, առաջին[illegible]ած, [illegible] որ թվերը արդեն կարգավ[illegible] գործընթացը կարելի է համարել ավարտված: [illegible]ցքում հնարավոր են



Նկ.4.10. Խնդիր 37 լուծման Ա37.2 ալգորիթմի բլոկ-սխեմա:

շեղումներ երբ որևէ զույգում նախորդ թիվը ավելի մեծ է հաջորդից: Այս դեպքում կարելի է հարակից թվերը տեղերով փոխանակել և շարժվել առաջ: Այսպես ընթանալով դեպի զանգվածի վերջը, մենք քայլ առ քայլ փոքրագույն տարրին դուրս կմղենք վերջին տեղը:

Մեկ անգամ անցնելով սկզբից մինչև վերջ՝ պետք է պարզել՝ եղե՞լ է, արդյոք, գոնե մեկ զույգում թվերի տե–ղափոխություն: Եթե չի եղել, ապա կարելի է գործընթացը ավարտել, հակառակ դեպքում՝ վերսկսել ստուգումները՝ սկսած առաջին թվից մինչև նախավերջին տարրը:

Երկրորդ երթի արդյունքում արժեքով երկրորդ փոքր թիվը կհայտնվի վերջից երկրորդ տեղում: Ցանկացած եր–կու թվերի տեղափոխության պարագայում զանգվածի տարրերի ստուգումները պետք է վերսկսվեն:

Այսպես շարունակում ենք, մինչև որևէ երթի ընթացքում թվերի տեղափոխություն չկատարվի: Ուրեմն, զանգվածի տարրերի վրայով մենք կանցնենք այնքան անգամ, որքան թվեր խախտում են ընդհանուր կարգավորվածությունը:

Մնում է որոշել, թե զանգվածի վրայով անցնելուց հետո ինչպե՞ս պարզել երկու թվերի տեղափոխության փաստը: Նման դեպքերում, երբ գործողությունները ավարտելուց հետո հարկ է լինում պարզել որևէ երևույթի իրականացման փաստը, կիրառում են, այսպես կոչված, ՛**փոխանջատիչներ**՛: Փոխանջատիչը տրամաբանական տիպի փոփոխական է, որը կարող է ընդունել երկու արժեքներից մեկը. ՛**այո**՛ կամ ՛**ոչ**՛: Իսկ կիրարկման եղանակը հետևյալն է. մինչև գործընթացի սկիզբը փոխանջատիչի արժեքը ընդունում են հավասար ՛այո՛ կամ ՛ոչ՛ արժեքին: Եթե ընթացքում հետաքրքրող երևույթը տեղի է ունենում, ապա փոխանջատիչին վերագրում են սկզբնականին հակադարձ արժեք: Գործընթացը ավարտելուց հետո փոխանջատիչի արժեքը կարող է միանշանակ վկայել երևույթի իրականացումը:

Որոշ դեպքերում, որպես փոխանջատիչ, կիրառում են թվային փոփոխական, որին գործընթացի սկզբում վերա–գրում են որևէ թվային արժեք, իսկ հետաքրքրող երևույթի իրականացման պահին՝ այլ արժեք: Գործընթացը ավարտե-լուց փոխանջատիչի արժեքի հիման վրա կարելի է երևույթի մասին անել որոշակի եզրակացություններ:

Եթե որպես թվային փոխանջատիչի սկզբնական արժեք վերցվի զրոն, և երևույթի յուրաքանչյուր կատարման պահին փոխանջատիչի ընթացիկ արժեքին գումարվի ՛1՛ (մեկ), ապա վերջում փոխանջատիչի զրո արժեքը կվկայի երևույթի բացակայության մասին, իսկ որևէ այլ արժեք ցույց կտա այդ երևույթի իրականացման քանակը:

Այսպիսով դուք ծանոթացաք փոխանջատիչների երկու տարբերակներին: Թե որը ընտրել այս կամ այն դեպքում՝ կախված է կոնկրետ իրավիճակից: Անդրադառնալով մեր խնդրին՝ նպատակահարմար ենք գտնում կիրառել տրամաբանական տիպի փոխանջատիչ, քանի որ մեզ հետաքրքրքրում է միայն երկու թվերի տեղափոխման փաստը, և ոչ թե քանակը:

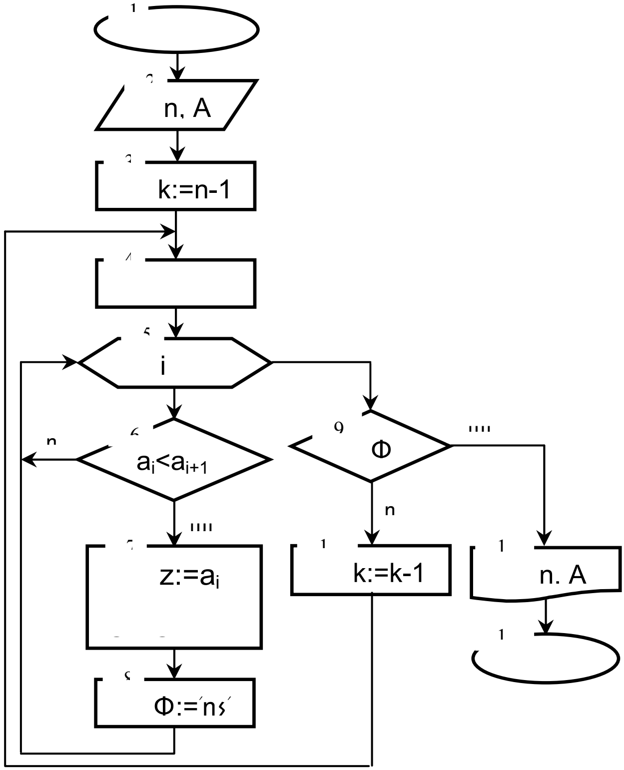
Տրամաբանական փոխանջատիչի կիրարկմամբ կարգավորման պղպջակավոր ալգորիթմի բլոկ-սխեման ներկայացված է նկ.4.11-ում: Այստեղ փոխանջատիչի դերում հանդես է գալիս Φ փոփոխականը, որը 4…10 բլոկներից կազմված վերջնապայմանով ցիկլի յուրաքանչյուր կրկընության սկզբում ընդունում է ՛այո՛ արժեք, որը կարող է փոխարինվել հակադարձ արժեքով միայն, եթե զանգվածի վրայով անցնելիս, որևէ երկու տարր տեղերով փոխանակվեն (բլ.7,8): Իսկ k փոփոխականի դերը՝ ցիկլի կրկնությունից կրկնություն կարգավորվող տարրերի քանակի մեկով նվազեցումն է: Չէ՞ որ վերը նշվեց, որ զանգվածի վրայով յուրաքանչյուր անգամ անցնելիս, վերջից տեղադըրվում են՝ նվազագույն արժեքով տարրը, հետո՝ նրան գերազանցող արժեքով երկրորդ տարրը և այլն:

Դուք արդեն նկատեցիք, որ թիվ 10 բլոկում ստուգվող պայմանը ներկայացված է անսովոր ձևով: Եթե պայմանը գրանցեինք առնչության տեսքով, ապա այն կներկայացվեր այսպես. Φ=՛այո՛: Հաշվի առնելով այն հանգամանքը, որ առնչության ստուգման արդյունքը կարող է համընկնել ՛այո՛ արժեքի հետ, նման դեպքերում ընդունված է առնչությունից հրաժարվել՝ գրելով միայն տրամաբանական փոփոխականի արժեքը:

Համեմատելով պղպջակավոր ալգորիթմը նախորդ՝ Ա37.2 ալգորիթմի հետ կարող ենք հավաստել, որ այս դեպքում փոխանջատիչի շնորհիվ զանգվածի վրայով ավելորդ անցումներ չեն կատարվում: Իսկ եթե զանգվածի տարրերը ներմուծվում են արդեն կարգավորված հերթականությամբ, ապա մեկ անգամ անցնելով առաջին տարրից մինչև վերջին տարրը փոխանջատիչը իր արժեքը չի փոխի, հետևաբար, ալգորիթմի աշխատանքը կավարտվի:

Եթե կարծում եք, որ սրանով ավարտեցինք ծանոթությունը զանգվածի տարրերի կարգավորման գործընթացի տարբերակներին, ապա շտապեցիք: Քիչ էլ չարաշահենք ձեր համբերությունը և մատուցենք ևս մեկ տարբերակ: Ավելի ճիշտ՝ պղպջակավոր մեթոդի կատարելագործված տարբե-րակ: Թե որը դուք կնախընտրեք, էական չէ: Պարզապես ցանկություն կա մեկ անգամ ևս ցուցադրելու մարդկային մտքի՝ անվերջ որոնելու անգնահատելի հատկությունը:

Եվ այսպես, պատկերացրեք, որ հանդիպելով կարգա–վորվածությունը խախտող երկու հարակից՝ a_i և a_{i+1} տար–րերը, դուք սկսում եք հաջորդաբար համեմատել a_{i+1} տարրը a_i-ին նախորդող տարրերի հետ հակառակ ուղղությամբ, այսինքն, դեպի զանվածի սկիզբը: Յանդիպելով առաջին իսկ a_k տարրին, որի արժեքը գերազանցում է a_{i+1}-ի արժեքը, բնական է տեղափոխել a_{i+1} տարրը (k+1) դիրք: Այսպիսով կստանանք մասամբ կարգավորված տվյալների հաջորդա-

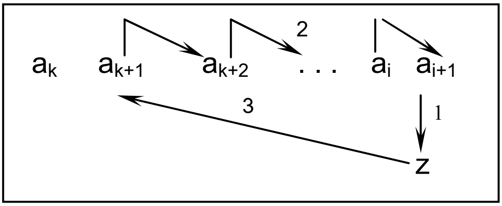


Նկ.4.11. Զանգվածի տարրերի ըստ արժեքների նվազման կարգավորման պղպջակավոր՝ Ա37.3 ալգորիթմի բլոկ-սխեմա:

կանություն: Շարունակելով առաջընթացը նույնը կարելի է իրագործել հաջորդ ՛անհաջող՛ զույգերի նկատմամբ:

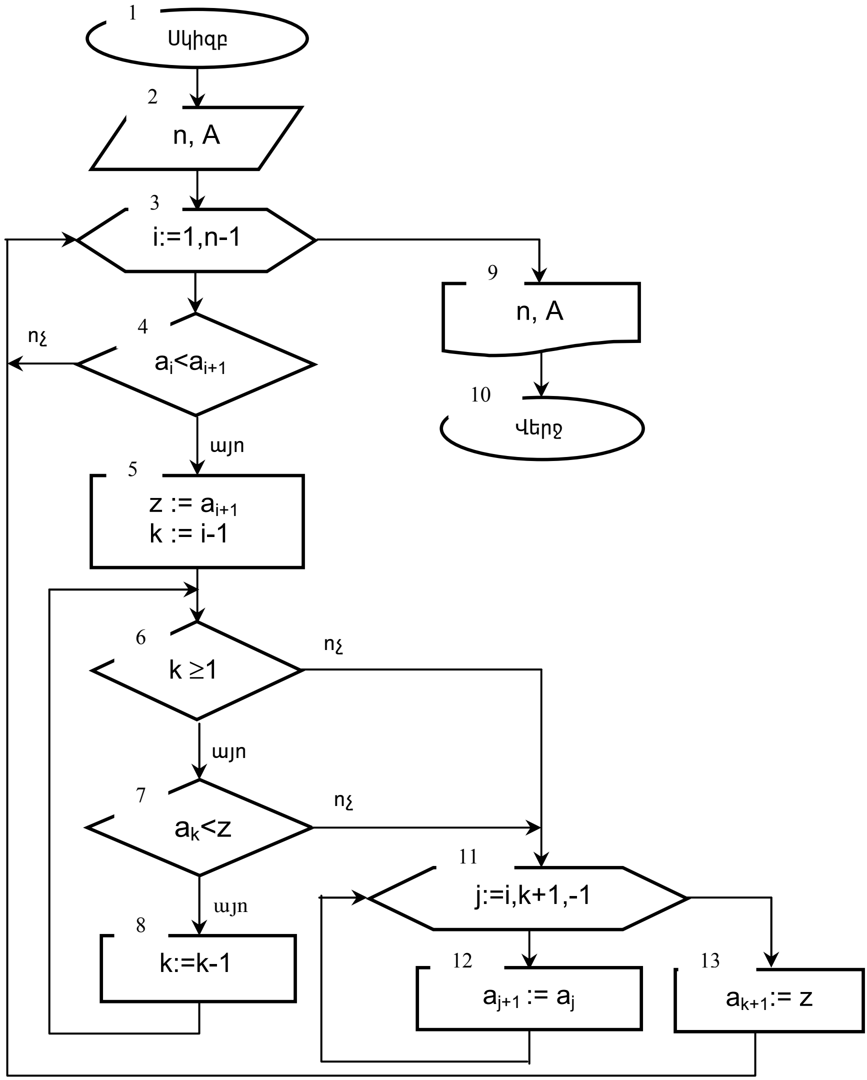
Սակայն հաջորդականությունում տարրի տեղափոխու–թյունը միանշանակ չէ, քանի որ այն տեղը, որը պատ–րաստվում ենք գրավել, զբաղված է: Մեր դեպքում (k+1)–րդ դիրքում, ինչպես և այլ համարների տակ, գրանցված են տարբեր թվեր: Հետևաբար, միակ ելքն է՝ a_{i+1} թիվը կրկնօ-րինակել և a_{k+1}, a_{k+2}, ... ,a_i տարրերը տեղաշարժել դեպի աջ մեկ դիրքով: Այժմ կարելի է a_{i+1}-ի կրկնօրինակված արժեքը արտագրել (k+1) դիրքում առանց վնասելու այդտեղ նախկի-նում գրանցված թիվը: Նկարագրածը պատկերացնելու համար առաջարկում ենք դիմել 4.12 գծանկարին:

Նկ.4.12. a_{i+1} տարրի (k+1)-րդ դիրք տեղա-փոխման գործընթացի պարզաբանում:



Գծանկարում թվերով նշանակված է համապատաս–խան գործողության համարը տեղափոխման գործողությունների հերթականությունում, որն արդեն վերը շարա–դրվել է: Կարգավորման չորրորդ տարբերակի իրացման ալգորիթմում, որի բլոկ-սխեման բերված է նկ.4.13-ում, թվի տեղափոխությունը ներկայացված է թիվ 5, 11-12 և 13 բլոկներով: Ինչպես երևում է 11-12 բլոկներից կազմված ցիկլից, a_{k+1}, a_{k+2}, ... , a_i տարրերի տեղաշարժը դեպի աջ կատարվում է սկսած վերջին՝ a_i տարրից, տեղ ազատելով ձախից հարակից տարրի համար:

Ասածին մնում է ավելացնել, որ թիվ 6-8 բլոկներից կազմված ցիկլը իրականացնում է a_{i+1} թվի նոր տեղի որո–նումը զանգվածի սկզբնամասում, որը մինչև a_i տարրը արդեն կարգավորված է: Որոնելով այնպիսի a_k տարր, որը արժեքով գերազանցում է a_{i+1} տարրը, մենք հատկացնում ենք նրան հաջորդող (k+1) տեղը a_{i+1} տարրին: Եթե որոն–ման ընթացքում այդպիսի տարր չի հանդիպում, ապա ցիկլի k պարամետրը ընդունում է զրո արժեք և a_{i+1} տարրի նոր տեղի համարն է դառնում առաջին դիրքը (k+1=1):



Նկ.4.13. Խնդիր 37 լուծման Ա37.4 ալգորիթմի բլոկ-սխեմա:

Ինչպես երևում է վերջին գծանկարից, զանգվածի կարգավորումը իրագործվում է, հիմնականում, մեկ ցիկլով, որի պարամետրն է i փոփոխականը: Ներքին երկու ցիկլերը կատարվում են միայն անհրաժեշտության դեպքում, երբ հանդիպում ենք ՛անհաջող՛ զույգի, որը խախտում է կարգավորվածությունը. այնպես որ ալգորիթմի ժամանակային բնութագիրը անմիջականորեն կախված է այդպիսի զույգերի քանակից: Իսկ դրանց բացակայության դեպքում, ունենալով տվյալների ի սկզբանե կարգավորված հաջորդականություն, կաշխատի միայն արտաքին ցիկլը, բաղկացած 3-4 բլոկներից:

Այսպիսով կարելի է սահմանափակել միաչափ զանգվածներ մշակող ալգորիթմների ուսումնասիրությունը, մեկ անգամ ևս թվարկելով այն հիմնական նպատակները, որոնց հետապնդում էինք: Օգտագործելով զանգվածների տարրերի հաջորդական համարակալումը, հնարավորություն է ընձեռվում, փոփոխելով ինդեքսի արժեքը հաջորդաբար ընտրել բոլոր տարրերը, վերլուծել նրանց արժեքները և մշակել ըստ խնդրի պահանջների: Իրականացվող գործողությունները կարող են լինել տարբեր բնույթի, օրինակ, գումարի (արտադրյալի) կամ քանակի հաշվում, մեծագույն (փոքրագույն) արժեքի որոշում, զանգվածի տարրերի վերադասավորում և ըստ որևէ հայտանիշի կարգավորում: Բոլոր դիտարկված ալգորիթմները հիմնարար դեր են խաղում ծրագրավորման ոլորտում: Թվարկված խնդիրներից յուրաքանչյուրն այս կամ այն տարբերակներով գործնականում հաճախակի են հանդիպում և բոլոր դեպքերում պետք է ստեղծագործաբար կիրառել լուծման համապատասխան ալգորիթմը: Երբեմն փակուղուց դուրս գալու համար անհրաժեշտ է լինում փնտրել և կառուցել նոր ալգորիթմներ կամ ընդհանրացնել եղածը: Որպես ասածի հիմնավորում կարելի է հիշատակել երկուսից ավելի թվերից մեծագույնի որոշման խնդիրը, երբ օգտվեցինք 5.2 ալգորիթմում իրականացված սկզբունքից, նկատելով այնտեղ գործողությունների կրկնողություն և մերժելով 5.1-ը:

Ինչպես դժվար չէր նկատել, զանգվածների կիրարկ–մամբ խնդիրներում հիմնական բարդությունը ներկայաց–նում է տվյալների մշակման հերթականության ընտրու–թյունը, որը կարելի է իրագործել տարբեր ուղղություն–ներով: Այսպես, խնդրի պայմաններից կախված զանգվածի տարրերը կարելի է ընտրել սկզբից, շարժվելով դեպի վերջը, կամ վերջից՝ ընթանալով դեպի սկիզբը: Տվյալ դրույթը վճռորոշ դեր է խաղում հատկապես երկչափ զանգվածների մշակման ժամանակ, որոնց ուսումնասիր–մանը մենք պատրաստվում ենք անցնել անհապաղ:

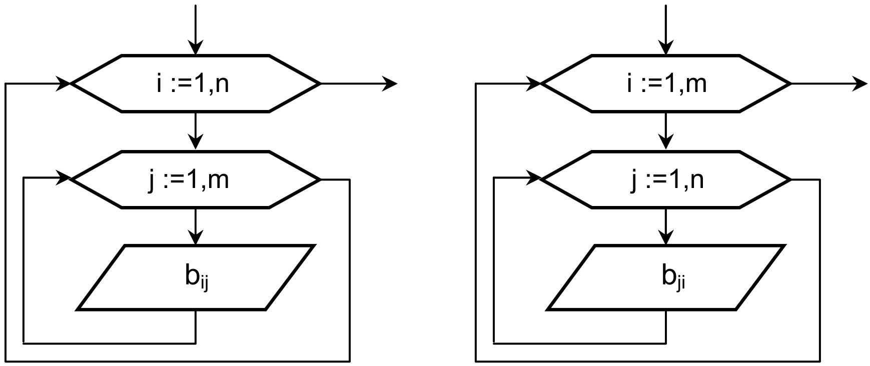
4.2. ԵՐԿՉԱՓ ԶԱՆԳՎԱԾՆԵՐԻ ՄՇԱԿՈՒՄ

Երկչափ զանգվածը կամ մատրիցը տվյալների դասավորման այլ տարբերակ է, որտեղ յուրաքանչյուր տարրի դիրքը որոշվում է իր երկու ինդեքսների կամ կոորդինատների արժեքներով: Ինչպես վերը նշվել է, առաջին ինդեքսը ցույց է տալիս այն տողի, իսկ երկրորդ ինդեքսը՝ այն սյան վրա, որոնց հատման կետում գտնվում է տվյալ տարրը: Մատրիցի տարրերը մշակելու համար նրանց արժեքները անհրաժեշտ է նախապես ներմուծել և տեղադրել իրենց տեղերում: Դրանից հետո մեր դեմ ծառանում է հաջորդ՝ տվյալների ընտրության խնդիրը, թե ինչպիսի հերթականությամբ ընտրել տվյալները նպատակին հասնելու համար: Ընտրությունը պետք է այնպես կազմակերպել, որպեսզի ոչ մի տարր դուրս չմնա մեր տեսադաշտից, իսկ ընթացքը լինի հնարավորին չափ արագ:

Դիցուք պահանջվում է ներմուծել n տողերից և m սյուներից բաղկացած հետևյալ մատրիցի տարրերը.

$$B = \begin{pmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1m} \\ b_{2,1} & b_{2,2} & \cdots & b_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ b_{n1} & b_{n2} & \cdots & b_{nm} \end{pmatrix}:$$

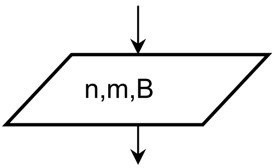
Սովորաբար մատրիցի տարրերը ներմուծում և մշակում են հետևյալ երկու հերթականություններով. ՛տող առ տող՛ կամ ՛սյուն առ սյուն՛, ինչը կապված է աղյուսակների ըն–թերցման կարգի հետ: Երկու դեպքում էլ սկզբում ներմուծվում է առաջին տողը՝ ձախից աջ, կամ սյունը՝ վերևից ներքև, որից հետո ներմուծվում է, համապատասխանաբար, երկրորդ տողը կամ սյունը և այլն: Վերլուծելով առաջին տարբերակը՝ կարող եք նկատել, որ կամայական i–րդ տողը ներմուծելիս, առաջին ինդեքսի արժեքը մնում է անփոփոխ և հավասար i, իսկ երկրորդ ինդեքսը ընդունում է 1-ից m հաջորդական ամբողջ արժեքները: Ինչպես ներկայացված է նկ.4.14ա-ում, ընդգրկելով նշված ցիկլը i պարամետրով ցիկլի մեջ և կրկնելով այն առաջին ինդեքսի 1-ից n բոլոր արժեքների համար, կիրականացնենք ամբողջ մատրիցի տարրերի ներմուծում:



ա) բ)

Նկ.4.14. ՛Տող առ տող՛ (ա) և ՛սյուն առ սյուն (բ) եղանակներով մատրիցի տարրերի ներմուծման ալգորիթմի բլոկ-սխեմաներ:

Նման դատողությունների հիման վրա կառուցված է ՛սյուն առ սյուն՛ ներմուծման ալգորիթմի բլոկ-սխեման, որը բերված է նկ.4.14բ-ում: Ի տարբերություն նախորդ տարբերակի այստեղ i–րդ սյան տարրերի ներմուծման ընթացքում անփոփոխ է մնում տարրի երկրորդ ինդեքսը, իսկ առաջինը փոփոխվում է 1-ից մինչև n: Այսուհետև մատրիցների հետ առնչվող բոլոր խնդիրներում, եթե մատրիցի տարրերի ներմուծմանը չի համատեղվում որևէ այլ գործողություն, մենք կկիրառենք ներմուծման ընդհանրացված բլոկ, ինչպիսին ներկայացված է նկ.4.15-ում: Այսպիսի ներկայացման ժամանակ մատրիցի տարրերի ներմուծման հերթականությունը էական չէ, իսկ մատրիցի չափսերը նշելիս առաջին հերթին նշում են տողերի քանակը (n), հետո՝ սյուների (m):



Նկ.4.15.

Նույն ձևով կարելի է վարվել մատրիցի տարրերի արտածման դեպքում, փոխարինելով ներմուծման բլոկը արտածման բլոկով և տեղադրելով, բնական է, մատրիցի տարրերի արժեքների որոշման հատվածից հետո:

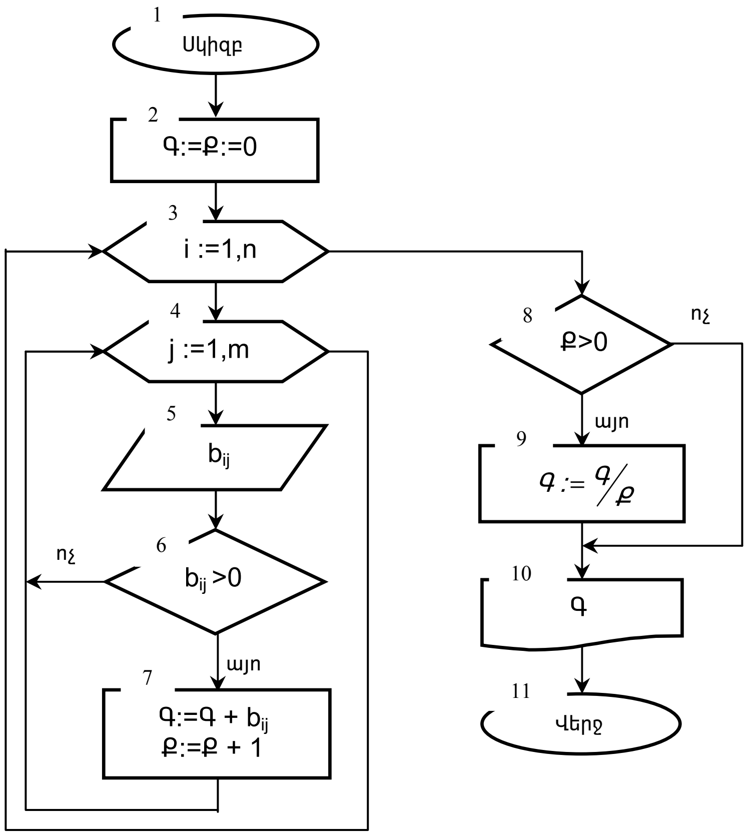
Այսպիսով բոլոր նախապատրաստական աշխատանքները ավարտված են, կարող ենք անցնել մեր հիմնական գործին:

<u>Խնդիր 38:</u> Տրված է՝ $B=\left\|b_{ij}\right\|,\ i=\overline{1,n};\ j=\overline{1,m}$ մատրիցը: Պահանջվում է հաշվել և արտածել մատրիցի բոլոր դրական տարրերի միջին թվաբանականը:

Վերիիշելով, որ թվերի միջին թվաբանականը այդ թվերի գումարի և քանակի քանորդն է, կարելի է հետևյալ կերպ ուրվագծել տվյալ խնդրի լուծումը. մատրիցի յուրաքանչյուր տարրի արժեքը ներմուծելուց անմիջապես հետո անհրաժեշտ է այն համեմատել զրոյի հետ: Եթե թիվը դրական է, ապա այն պետք է ընդգրկել գումարի մեջ և հաշվել: Այսպիսով խնդրի լուծումը հանգեց մեզ քաջ հայտնի գումարի և քանակի հաշվմանը, որոնք համատեղվում են տվյալների ներմուծման հետ: Միակ նորույթը՝ մատրիցի վրայով երթևեկության կազմակերպումն է: Ընդունենք տարրերի ներմուծման ՛տող առ տող՛ տարբերակը և դիմենք նկ. 4.16-ին որոշ գործողություններ պարզաբանելու միտումով:

Գծանկարից պարզաբանման կարիք ունեն՝ ցիկլերի աշխատանքը ավարտելուց հետո կատարվելիք գործողու– թյունները, իսկ ավելի ստույգ՝ թիվ 8 բլոկում ստուգվող առնչությունը: Այն պայմանավորված է նրանով, որ մատրիցի մեջ հնարավոր է չգտնվի ոչ մի հատ դրական թիվ, ինչի հետևանքով և գումարը (Գ) և քանակը (Ք) կմնան հավասար զրոյի: Պատկերացնու՞մ եք, թե ինչ տեղի կունենա եթե առանց պայմանը ստուգելու կատարենք թիվ 9 բլոկը: Թվում է՝ հետևանքները բացատրելու կարիք չկա:

Այսպիսով, դրական թվերի բացակայության դեպքում կստանանք Գ=0 պատասխանը, իսկ գոնե մեկ թվի առկա– յության դեպքում՝ զրոյից տարբերվող թիվ:



Նկ.4.16. Խնդիր 38 լուծման ալգորիթմի բլոկ-սխեմա:

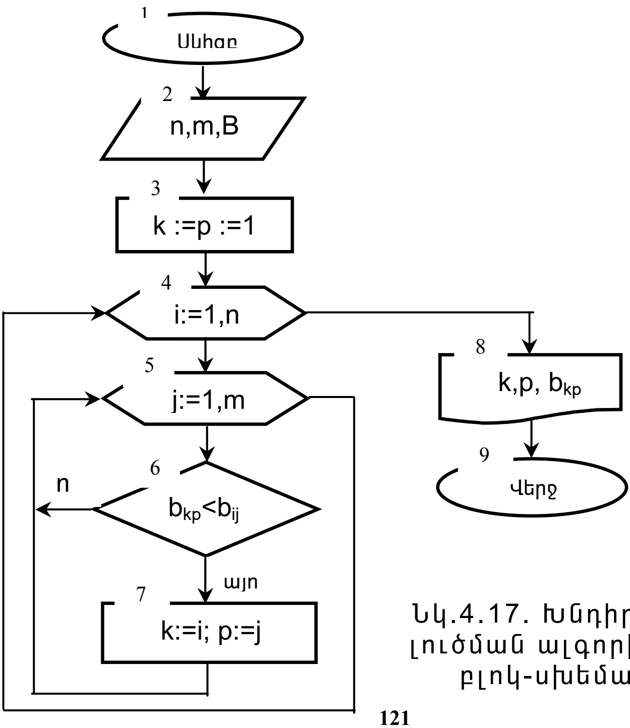
<u>Խնդիր 39:</u> Տրված է` $B = \|b_{ij}\|,\ i = \overline{1,n};\ j = \overline{1,m}$ մատրիցը: Պահանջվում է որոշել և արտածել մատրիցի ամենափոքր արժեք ունեցող տարրը:

Տվյալ խնդիրը լուծելու համար կիրառենք միաչափ զանգվածներում նմանատիպ խնդիր լուծող Ա33.2 ալգորիթմի կառուցման սկզբունքը (տես նկ.4.5): Քանի որ հիշատակված ալգորիթմը ժամանակին մանրամասն վերլուծվել է և, հուսով ենք` ձեր կողմից յուրացվել, ապա սահմանափակվենք նույն սկզբունքը իրականացնող մատրիցի տարրերից փոքրագույնը որոշող ալգորիթմի բլոկ-սխեմայի ներկայացմամբ, որը բերված է նկ.4.17-ում:

Խնդիր 40: Տրված է` $B = \|b_{ij}\|$, $i = \overline{1,n}$; $j = \overline{1,m}$ մատրիցը: Պահանջվում է մատրիցի տողերը կարգավորել ըստ k-րդ սյան տարրերի արժեքների աճման` վերևից ներքև:

Չնարավոր չէ վերադասավորելով մատրիցի տողերը միաժամանակ կարգավորել բոլոր սյուների տարրերը: Սակայն շատ հաճախ անհրաժեշտ է լինում աղյուսակները կարգավորել ըստ որևէ սյան տվյալների արժեքների աճման կամ նվազման: Օրինակ, կարելի է պատկերացնել, որ մատրիցի k-րդ սյունը պարունակում է n անձանց աշխատանքային ստաժը և տվյալ պահին հարկավոր է մատրիցի տողերը վերադասավորել անձերի ստաժի աճման կամ նվազման կարգով:

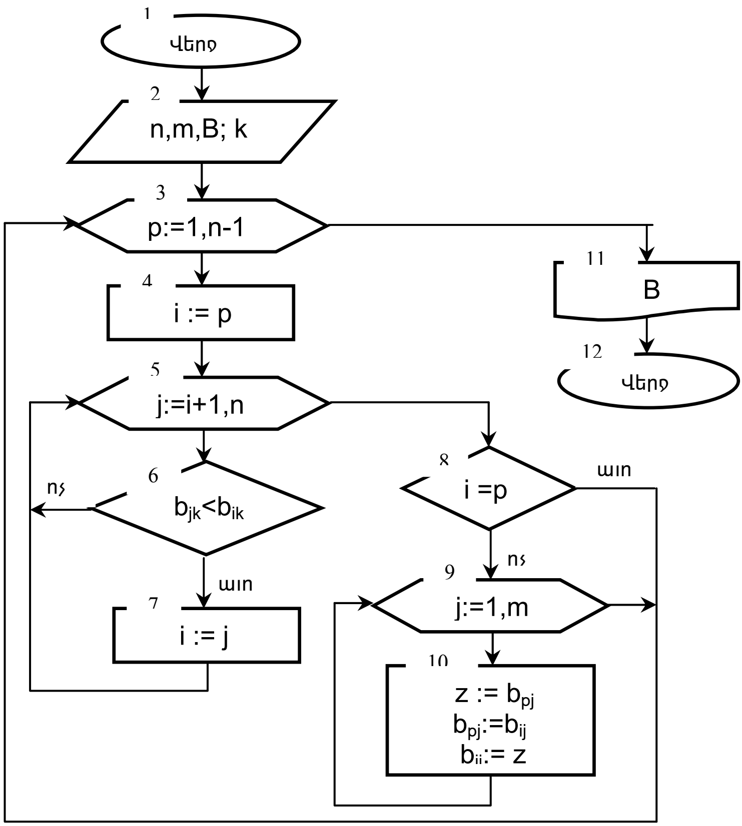
Առաջադրված խնդիրը լուծելու համար կարող ենք օգտվել վեկտորների նկատմամբ կիրառված կարգավորման ալգորիթմներից: Եթե հիշում եք, դրանք երկուսն էին, և



Նկ.4.17. Խնդիր 39 լուծման ալգորիթմի բլոկ-սխեմա:

համեմատության արդյունքում պարզվեց, որ պղպջակավոր եղանակը նախընտրելի է: Սակայն մատրիցների կիրարկմամբ տվյալ մեթոդը իրեն չի արդարացնում, քանի որ կարգավորման ընթացքում տարրերի տեղափոխությունների քանակը չափազանց շատ է: Եթե միաչափ զանգվածում տեղափոխման ենթակա է մեկ տարր, ապա երկչափ զանգվածում՝ մեկ տող, որը բաղկացած է m տարրերից: Ուրեմն երկու տողերի տեղափոխությունը հանգում է **2m** տարրերի տեղափոխության, որն ընդունելի չէ: Հետևաբար, մնում է կիրառել վեկտորների կարգավորման առաջին եղանակը, որը մատրիցների կիրարկմամբ ձևափոխվում է այն տեսքի, որը բերված է նկ.4.18-ում:

Համեմատելով Ա37.2 ալգորիթմը՝ մատրիցների կի–



Նկ.4.18. Խնդիր 40 լուծման ալգորիթմի բլոկ-սխեմա:

րարկմամբ Ա40 ալգորիթմի հետ՝ կարող ենք նշել նրանց բացարձակ նույնությունը, տարբերությամբ թիվ 9-10 բլոկ–ների: Չէ՞ որ մատրիցի յուրաքանչյուր սյուն կամ տող կա–րելի է դիտարկել որպես վեկտոր, որի տարրերը բնութա–գրվում են երկու ինդեքսներով: Ընդ որում համընկնում են միևնույն սյան տարրերի երկրորդ ինդեքսները, իսկ միև–նույն տողի տարրերինը՝ առաջին: Եթե վեկտորների դեպ–քում երկու տարրերի արժեքները տեղափոխելու համար պահանջվում էր երեք գործողություն (նկ.4.10-բլ.9), ապա մատրիցներում այդ երեք գործողությունները իրականաց-վելու են p և i տողերի համապատասխան տարրերի միջև, որտեղ i-ն այն տողի համարն է, որտեղ գտնվում է k-րդ սյան փոքրագույն տարրը՝ սկսած p-րդ տարրից (b_{pk}) մինչև վեր-ջին՝ b_{nk} տարրը: Իսկ մեկից ավելի, m քանակով զույգ տար-րերի տեղափոխությունը կարելի է իրագործել միայն ցիկլի միջոցով, ինչը և ներկայացված է 9-10 բլոկներից կազմված ցիկլով:

Այսպիսով, մատրիցի առանձին սյուների կամ տողերի վերաբերյալ կարելի է կիրառել բոլոր այն ալգորիթմները, որոնք փորձարկել ենք վեկտորների նկատմամբ: Իսկ սյունից սյուն կամ տողից տող անցնելը ապահովվում է արտաքին ցիկլում: Մնում է միայն ճիշտ ընտրել երթևեկության ուղղությունը և ժամանակին կատարել որոշ փոփոխական-ների սկզբնական արժեքների վերագրում, ինչպես հաջորդ օրինակում:

<u>Խնդիր 41:</u> Տրված է՝ $B=\left\|b_{ij}\right\|,\ i=\overline{1,n};\ j=\overline{1,m}$ մատրի-ցը: Որոշել և արտածել յուրաքանչյուր սյան փոքրագույն արժեք ունեցող տարրը:

Առանձին սյան փոքրագույն արժեքով տարրը որոշելու համար կարող ենք օգտվել նախորդ խնդրում k-րդ սյան փոքրագույն տարրը որոշող ցիկլից: Փոփոխելով k-ի արժեքը 1-ից m, հաջորդաբար կստանանք բոլոր սյուների փոքրագույն արժեքները: Եթե այդ արժեքները հարկավոր չէ պահպանել հետագա օգտագործման համար, ապա բավական է ստանալուն պես դրանց արտածել և անցնել հաջորդ սյանը: Իսկ եթե լրացուցիչ պահանջվում է հիշել սյուների մինիմալ տարրերը, ապա արտածելուց բացի յուրաքանչյուր անգամ մինիմալ արժեքը հարկավոր է գրանցել առանձին վեկտորի համապատասխան դիրքում: Խնդրի լուծման վերջին տարբերակին կարող եք ծանոթանալ դիմելով նկ. 4.19-ին:

Նկ.4.20-ում բերված է հաջորդ խնդրի լուծման ալգո–րիթմի բլոկ-սխեման, որի պայմանը հետևյալն է.

<u>Խնդիր42:</u> Տրված է՝ $B=\left\|b_{ij}\right\|,\ i=\overline{1,n};\ j=\overline{1,m}$ մատրիցը: Ստեղծել n երկարություն ունեցող A վեկտոր, որի տարրերի արժեքները հավասար լինեն մատրիցի համապա–տասխան տողի դրական տարրերի քանակին:

Թվում է, թե վերջին երկու խնդիրների միջև ընդհա–նուրը միայն երկչափ կառույցն է, ուր գրանցված են տվյալները: Սակայն, ինչպես երևում է 4.19 և 4.20 նկար–ներում բերված այդ երկու խնդիրների լուծման ալգո–րիթմների թեկուզ մակերեսային համեմատությունից, տվյալ երկու խնդիրների լուծման սկզբունքը նույնն է, տարբեր են միայն տարրերի ընտրման հերթականությունը և որոշվող ֆունկցիաները:

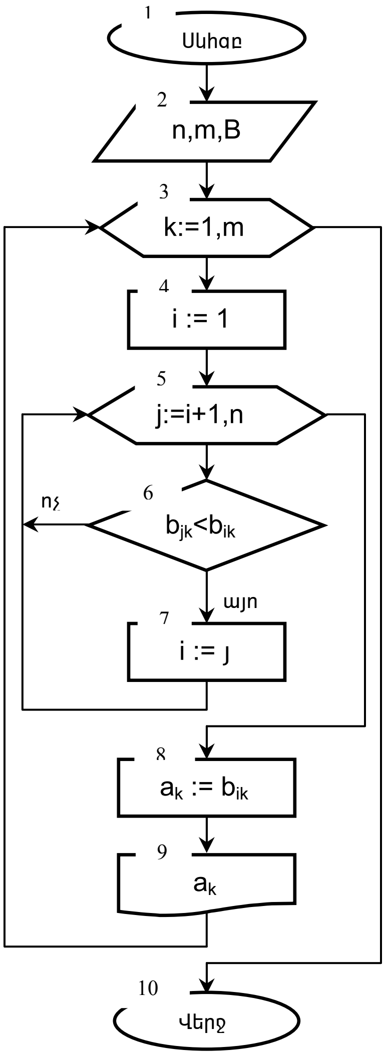
4.20 գծանկարի ներդրված երկու ցիկլերից ներքինը (բլ.5-7) կատարում է i-րդ տողի դրական տարրերի հաշ–վառում՝ տողի վրա ամեն անգամ դրական թիվ հանդիպելիս քանակին մեկ գումարելով: Քանի որ այդ տողին համա-պատասխանող քանակը պետք է գրանցվի A վեկտորի i–րդ դիրքում, այն նշանակված է a_i: Իսկ մենք արդեն գիտենք, որ քանակը, հանդիսանալով մեկերի գումար, պետք է նախապես ընդունի զրո արժեք, որը նախատեսված է տեղադրել տողի համարը որոշելուց հետո (բլ.3) և անմիջապես տողի վրայով երթևեկելուց առաջ:

Դուք հավանաբար նկատել եք, որ վերջին խնդիրների լուծումները ուղեկցվում են բավականին սուղ մեկնա–բանություններով՝ շատ ավելի հակիրճ, քան նախկինում: Դրա պատճառն այն է, որ մենք նոր ալգորիթմներ չենք առա-ջադրում, այլ կիրառում ենք արդեն կառուցած և ժամանակին մանրամասն ուսումնասիրված ալգորիթմները՝ ընտրելով լավագույնը և հարմարացնելով այն նոր պայմաններին: Նույնը վերաբերում է սկզբունքներին. պարտադիր չէ ալգորիթմների բացարձակ նման կրկնությունը, կարևորը՝ այդ ալգորիթմի տրամաբանության կամ նրա հիմքում ընկած սկզբունքի ճիշտ ընկալումն է, որը ձեզ հնարավորություն կընձեռի ցանկացած պայմաններում օգտվել նրանցից: Ասածը ցուցադրենք հաջորդ օրինակների վրա:

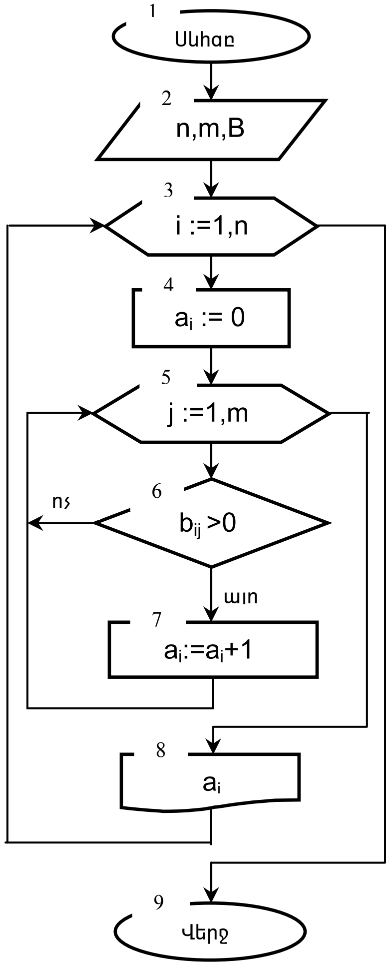
<u>Խնդիր 43:</u> Տրված է՝ $B=\left\|b_{ij}\right\|,\ i=\overline{1,n};\ j=\overline{1,m}$ մատրի-ցը: Պահանջվում է հաշվել և արտածել միայն այն տողերի

տարրերի գումարը, որոնց տարրերը կարգավորված են ըստ արժեքների նվազման:

Այսպիսով, հիմնական խնդիրը՝ գումար հաշվելն է, սակայն ոչ բոլոր թվերի, այլ ընտրովի, ինչը պետք է պարզվի



Նկ.4.19. Խնդիր 41 լուծման ալգորիթմի բլոկ-սխեմա:



Նկ.4.20. Խնդիր 42 լուծման ալգորիթմի բլոկ-սխեմա:

համապատասխան պայման ստուգելուց հետո: Քանի որ պայմանը կապված է տողերի հետ, մատրիցի տարրերը հարկավոր է ընտրել 'տող առ տող' սկզբունքով:

Յուրաքանչյուր տողի վրա անցնելիս, բնական է, նախ պետք է ստուգել պահանջվող պայմանը և հետո, եթե այն առկա է, գումարել այդ տողի բոլոր տարրերի արժեքները, հակառակ դեպքում՝ անցնել հաջորդ տողին: Սակայն տողին վերաբերող պայմանը չի որոշվում մեկ առնչություն ստուգելով, այդպիսի համեմատություն պետք է կատարվի հաջորդաբար հարակից զույգ թվերի նկատմամբ, այսինքն ցիկլով: Ընդ որում տվյալ ցիկլը կունենա երկու ելք. մեկը բնական, երբ տողը կարգավորված է, մյուսը հարկադրված, երբ կհանդիպենք կարգավորվածությունը խախտող առաջին զույգին: Հետևաբար պայման ստուգող ցիկլը չի կարող նկարագրվել 'մոդիֆիկացիա' բլոկի կիրառմամբ՝ այն կարող է ներկայացվել սկզբնապայմանով ցիկլով:

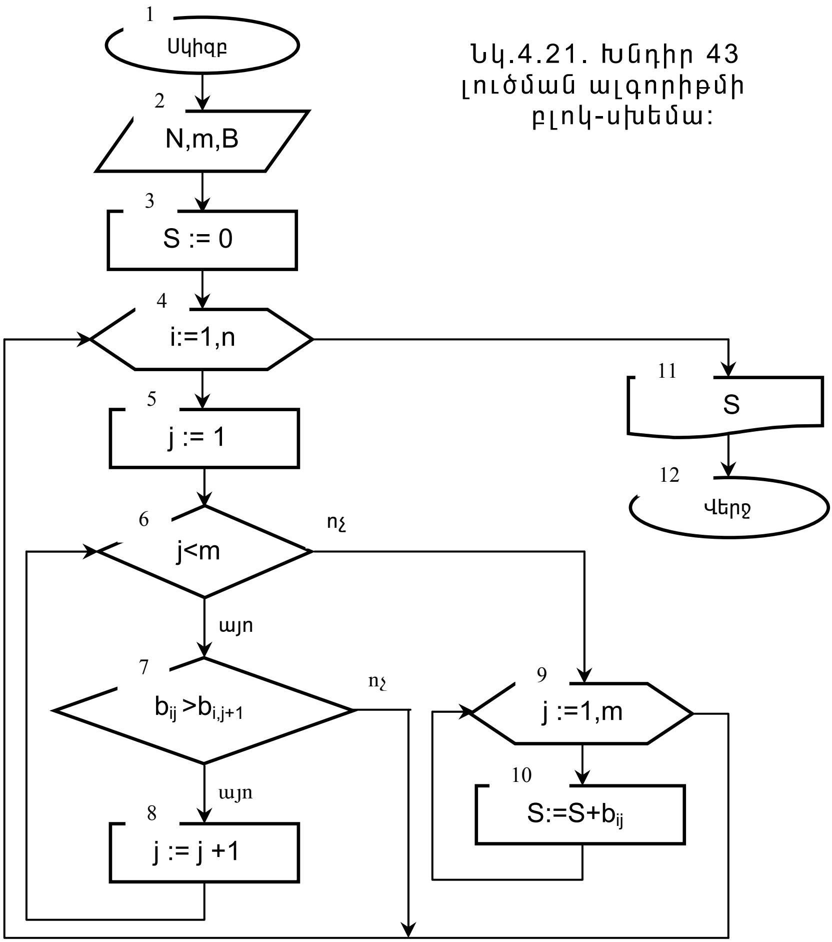
Այսպես ուրվագծելով խնդրի լուծումը և ընտրելով ալգորիթմում կիրառվող սկզբունքները, կարող եք ինքնուրույն կառուցել տվյալ խնդրի լուծման ալգորիթմի բլոկ–սխեման և համեմատել այն նկ.4.21-ում բերված գծանկարի հետ:

Շարունակելով մատրիցների և վեկտորների փոխհարաբերության թեման՝ դիտարկենք ևս մի շարք օրինակներ:

<u>Խնդիր 44:</u> Տրված են n, m բնական թվերը և n տարր պարունակող $X(x_1,x_2,\ldots,x_n)$ վեկտորը: Պահանջվում է ստանալ և արտածել $m\times n$ չափի A մատրիցը, որի տեսքն է.

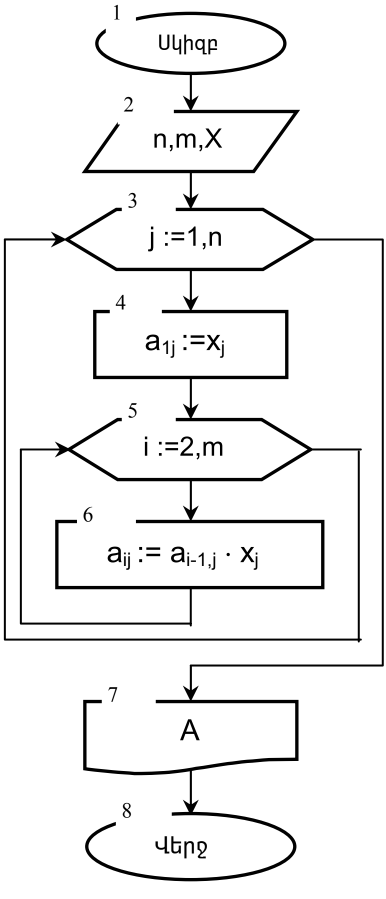
$$A=\begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \cdots & \cdots & \cdots & \cdots \\ x_1^m & x_2^m & \cdots & x_n^m \end{pmatrix}:$$

Ինչպես երևում է մատրիցի տեսքից, նրա առաջին տողում ամբողջությամբ գրանցվելու են X վեկտորի տարրերը: Վերևից ներքև յուրաքանչյուր j-րդ սյուն ($j=\overline{1,n}$) պարունակելու է իր առաջին տողում գրանցված x_j տարրի հաջորդական աստիճանները՝ մինչև x_j^m: Մտաբերելով նախկինում քննարկված՝ աստիճանի հաշվման ռեկուրսիվ գործընթացը, կարող ենք ընտրել մատրիցի ձևավորման հետևյալ

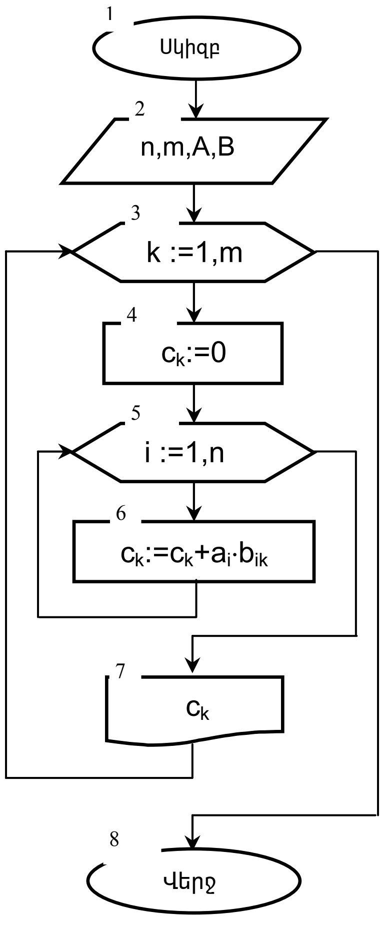


Նկ.4.21. Խնդիր 43 լուծման ալգորիթմի բլոկ-սխեմա:

սկզբունքը. շարժվելով առաջին սյունից դեպի վերջինը (արտաքին ցիկլ $j=\overline{1,n}$ պարամետրով), յուրաքանչյուր սյան առաջին դիրքում տեղադրում ենք x_j տարրը ($a_{1j} = x_j$), որից հետո ներքին ցիկլում ($i=\overline{2,m}$) տվյալ սյան յուրաքանչյուր a_{ij} տարրի արժեք կհաշվարկվի անմիջապես իր վերևի տարրի և վեկտորի j–րդ տարրի արժեքների արտադրյալով: Համոզված ենք, որ դուք ի վիճակի եք ինքնուրույն կառուցել այսպիսի ընթերցմամբ ալգորիթմի բլոկ-սխեման և համեմատել այն նկ.4.22-ում բերված գծանկարի հետ:



Նկ.4.22. Խնդիր 44 լուծման ալգորիթմի բլոկ-սխեմա:



Նկ.4.23. Խնդիր 45 լուծման ալգորիթմի բլոկ-սխեմա:

Տվյալ խնդրի լուծմանն այլ մոտեցում ցուցաբերելը, այսինքն մատրիցի 'տող առ տող' ձևակերպումը, կհան–գեցնի ավելորդ ցիկլի կիրառմանը, կապված առաջին տողի լրացման հետ: Դրանում համոզվելու համար առաջարկում ենք ձեզ որպես վարժություն իրագործել տվյալ սկզբունքը և համեմատել նախորդ ալգորիթմի հետ:

Գործնականում հաճախակի է առաջանում վեկտորի և մատրիցի կամ երկու մատրիցների արտադրյալներ հաշվելու անհրաժեշտությունը, ինչպես նաև մատրիցի տրանսպոնացման խնդիրը: Հաջորդ օրինակները նվիրված են հիշատակված խնդիրների լուծման ալգորիթմների կառուցմանը:

Խնդիր 45: Տրված են` n տարրեր պարունակող A վեկտորը և $B=\left\|b_{ij}\right\|,\ i=\overline{1,n};\ \ j=\overline{1,m}$ մատրիցը: Պահանջվում է ստանալ C=AB արտադրյալը:

Ինչպես հայտնի է մաթեմատիկայից, վեկտորի և մատրիցի արտադրյալը վեկտոր է, որի տարրերը տրված վեկտորի և մատրիցի համապատասխան սյան սկալյար արտադրյալն է: Հետևաբար, արդյունքում ստացվող վեկտորի երկարությունը հավասար է մատրիցի սյուների քանակին: Այսպիսով, A վեկտորի և B մատրիցի բազմապատկման արդյունքում ստացվող C վեկտորի տարրերն են` $c_1, c_2, \ldots, c_m$, որտեղ յուրաքանչյուր c_k ($k=\overline{1,m}$) տարրը A վեկտորի և B մատրիցի k-րդ սյան սկալյար արտադրյալն է: Սկալյար արտադրյալի բանաձևն է.

$$c_k=\sum_{i=1}^{n} a_i b_{ik}=a_1 b_{1k}+a_2 b_{2k}+\cdots+a_n b_{nk}\ : \qquad (4.1)$$

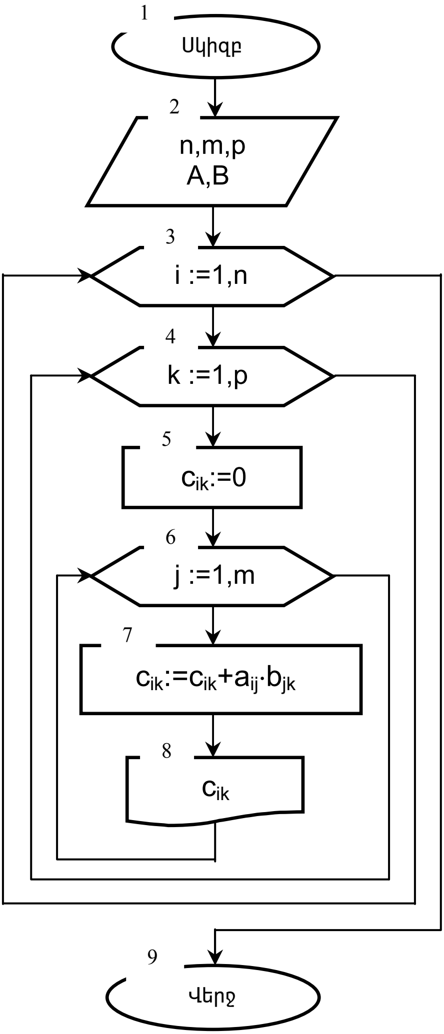
Նման գումարի հաշվման կազմակերպումը, ինչպես նաև մի շարք այսպիսի գումարների հաշվումը, մեզ համար սովորական երևույթ է: Դրա համար բավական է c_k մեծության բանաձև (4.1)-ով հաշվման ցիկլը ամբողջությամբ ընդգրկել k պարամետրով ցիկլի մեջ, որը պետք է փոփոխի k–ի արժեքը 1-ից m: Թե ինչպիսի տեսք կստանա ամբողջ ալգորիթմը, դուք կարող եք տեսնել` դիմելով նկ.4.23-ին:

Խնդիր 46: Տրված են A և B մատրիցները, որտեղ` $A=\left\|a_{ij}\right\|,\ i=\overline{1,n};\ j=\overline{1,m}$, իսկ $B=\left\|b_{ij}\right\|,\ i=\overline{1,m};\ j=\overline{1,p}$: Պահանջվում է հաշվել տրված մատրիցների C արտադրյալը:

Հայտնի է, որ երկու մատրիցներ բազմապատկելու համար անհրաժեշտ է, որ առաջին մատրիցի երկարու–թյունը (սյուների քանակը) հավասար լինի երկրորդ մատրիցի բարձրությանը (տողերի քանակին): Մատրիցների արտադրյալը նույնպես մատրից է, որի չափողականությունն է` n×p, այսինքն n տողերով և p սյուներով: C մատրիցի յուրաքանչյուր i–րդ տող ձևավորվում է նախորդ օրինակի նման՝ բազմապատկելով A մատրիցի i–րդ տողը, որպես վեկտոր, B մատրիցով.

$$c_{ik} = \sum_{j=1}^{m} a_{ij} b_{jk} = a_{i1} b_{1k} + a_{i2} b_{2k} + \cdots + a_{im} b_{mk}, \quad k = \overline{1, p:} \quad 4.2)$$

Մնում է ավելացնել, որ C մատրիցը ամբողջությամբ կձևավորվի, եթե բանաձև (4.2)-ը կիրառենք բոլոր տողերի համար, այսինքն, փոփոխենք տողի համարը 1–ից մինչև n: Իսկ դրա համար բավական է բանաձև (4.2)-ը իրականացնող Ա45 ալգորիթմում կիրառված ցիկլը ընդգրկել այլ ցիկլի մեջ, որի i պարամետրը կփոփոխի նշված սահմաններում (Նկ.4.24):



Նկ.4.24. Խնդիր 46 լուծման ալգորիթմի բլոկ-սխեմա:

Վերջին օրինակների հիման վրա դուք պետք է համոզվեիք, թե որքան ժամանակ է տնտեսվում հայտնի ալգորիթմներ կիրառելիս, մեկ խնդրում կիրառված սկզբունքը մյուսում կիրառելու դեպքում:

Անցնենք վերը նշված մատրիցների տրանսպոնացման խնդրին, որը կիրառելի է միայն քառակուսային մատրիցների նկատմամբ: Այդպիսի մատրիցների անկյունագծերի հետ կապված հիմնական յուրահատկության մասին արդեն նշել ենք: Դրան ավելացնենք այն հանգամանքը, որ քառակուսի մատրիցները սիմետրիկ են կամայական անկյունագծի նկատմամբ: Մատրիցի տրանսպոնացումը նրա այնպիսի ձևափոխումն է, երբ a_{ij} տարրը հայտնվում է a_{ji} տարրի տեղում, իսկ a_{ji} տարրը՝ a_{ij}–ի տեղում, այսինքն տեղի է ունենում քառակուսային մատրիցի պտույտ 180°-ով գլխավոր անկյունագծի շուրջ:

Տվյալ խնդրի լուծման ալգորիթմի պարզության պատճառով առաջարկում ենք նրա բլոկ-սխեմայի կառուցումը կատարել ինքնուրույն, հիշեցնելով միայն, որ երկու տարրերի տեղափոխությունը իրականացվում է օժանդակ փոփոխականի միջոցով:

Քանի որ անդրադարձանք քառակուսի մատրիցներին, դիտարկենք նրանց հետ կապված մի շարք այլ խնդիրներ, որտեղ մշակման են ենթակա անկյունագծերից վերև կամ ներքև տեղադրված տարրերը: Այդ նպատակով պարզենք նշված տիրույթների սահմանները:

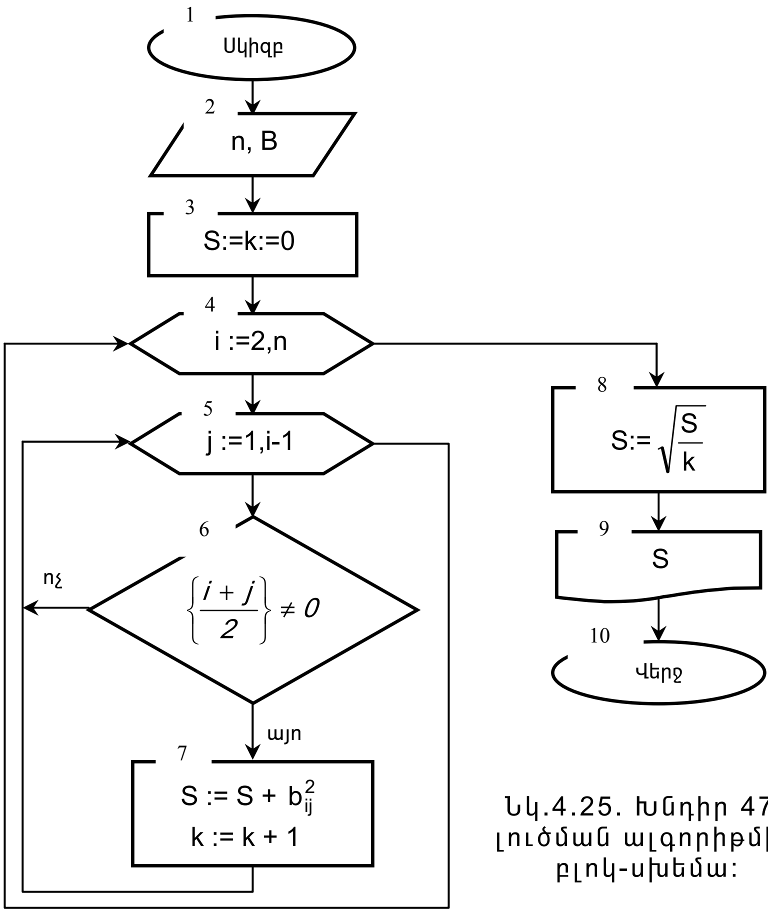
Գլխավոր անկյունագիծ: Հիշելով այն հանգամանքը, որ այդ անկյունագծի վրա տեղադրված տարրերի ինդեքսների արժեքները հավասար են միմյանց, կարող ենք հավաստել, որ կամայական տողի վրա մինչև գլխավոր անկյունագիծը տեղադրված տարրերի երկրորդ ինդեքսի արժեքը փոքր է առաջինից, իսկ անկյունագծից հետո տեղադրված տարրերի երկրորդ ինդեքսի արժեքը՝ ավելի մեծ: Այս դիտողությունից հետևում է, որ *գլխավոր անկյունագծից **ներքև** գտնվող տարրերի երկրորդ ինդեքսի արժեքը փոքր է առաջինից, իսկ գլխավոր անկյունագծից **վերև**՝ ավելի մեծ:*

Խնդիր 47: Տրված է n×n չափողականությամբ $B = \|b_{ij}\|$ քառակուսի մատրից ($i,j = \overline{1,n}$): Պահանջվում է հաշվել մատրիցի գլխավոր անկյունագծից ներքև գտնվող այն տարրերի միջին քառակուսայինը,որոնց ինդեքսների գումարը կենտ է:

Նախ հիշենք k քանակությամբ՝ $x_1,x_2,\ldots,x_k$ տարրերի միջին քառակուսայինի հաշվման բանաձևը, որը հետևյալ տեսքի է. $\sqrt{\frac{1}{k}\sum_{i=1}^{k} x_i^2}$: Մեր դեպքում գումարելիների դերում հանդես են գալիս խնդրի պայմաններին բավարարող B մատրիցի տարրերը, որոնց քանակը նախապես հայտնի չէ: Այն կորոշվի միայն մատրիցի գլխավոր անկյունագծից ներքև գտվող տարրերի ինդեքսների գումարները ստուգելուց և կենտ գումարների քանակը հաշվելուց հետո:

Այսպիսով, խնդրի լուծումը հանգում է հետևյալ գոր–ծողությունների կատարմանը. երբ գլխավոր անկյունագծից ներքև տեղադրված տարրերի վրայով, ինդեքսների կենտ գումարով տարրերի քառակուսիների գումարի և քանակի հաշվում: Ցիկլի ավարտից հետո, ունենալով ընտրված տարրերի քառակուսիների գումարը և քանակը, կարող ենք հաշվել նրանց միջին քառակուսայինը:

Նկարագրած ալգորիթմի բլոկ-սխեման բերված է նկ. 4.25-ում: Քանի որ գլխավոր անկյունագծի վրա գտնվող տարրերը ընդգրկված չեն որոշման տիրույթում, մատրիցի առաջին տողը մնում է այդ տիրույթից դուրս, և տողի համարը որոշող i պարամետրի արժեքը պետք է փոփոխվի սկսած 2-



Նկ.4.25. Խնդիր 47 լուծման ալգորիթմի բլոկ-սխեմա:

ից, այսինքն՝ երկրորդ տողից: Այան համարը որոշող երկրորդ՝ j ինդեքսի արժեքը պետք է փոփոխվի սկսած 1-ից մինչև (i-1), միշտ մնալով համապատասխան տողի համարից փոքր:

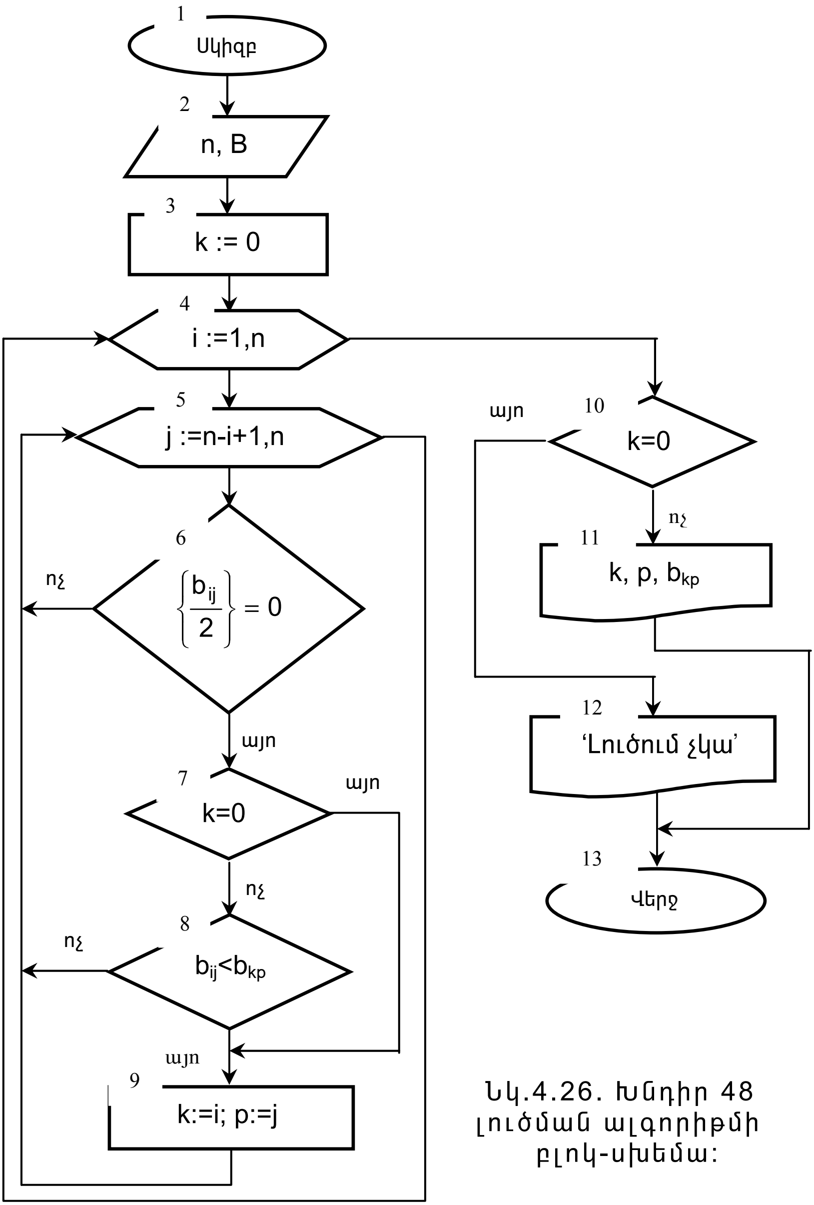
<u>Օժանդակ անկյունագիծ</u>: Այս անկյունագծի համեմատ վերև կամ ներքև տեղադրված տարրերի ինդեքսների հա– րաբերությունը կարելի է դուրս բերել, համեմատելով ան– կյունագծի վրա գտնվող տարրերի ինդեքսների արժեքների հետ: Եթե հիշում եք, այդպիսի տարրերի երկրորդ՝ j ինդեքսը կապված է առաջին՝ i ինդեքսի հետ հետևյալ բանաձևով. j=n–i+1, որտեղ n-ը մատրիցի տողերի և սյուների քանակն է: Յետևաբար, կարելի է պնդել, որ կամայական i–րդ տողի համար մինչև օժանդակ անկյունագիծը տեղադրված տարրերի երկրորդ ինդեքսի արժեքը փոքր է (n–i+1) մեծու-թյունից, իսկ անկյունագծից հետո տեղադրված տարրերի ինդեքսի արժեքը՝ դրանից մեծ: Այս դիտողությունից հետևում է, որ *օժանդակ անկյունագծից* ***վերև*** *գտնվող տարրերի երկրորդ ինդեքսի արժեքը փոքր է (n–i+1)-ից, իսկ օժանդակ անկյունագծից* ***ներքևինը***՝ *ավելի մեծ:*

<u>Խնդիր 48:</u> Տրված է n×n չափողականությամբ $B = \|b_{ij}\|$ քառակուսի մատրից ($i,j = \overline{1,n}$): Պահանջվում է հաշվել մատրիցի օժանդակ անկյունագծից ներքև կամ նրա վրա գտնվող զույգ արժեք ունեցող տարրերից փոքրագույնը:

Առաջին հայացքից թվում է, թե այս խնդրի լուծումը պետք է ընթանա նույնությամբ, ինչպես Ա39 ալգորիթմում. բավական է փոփոխել ինդեքսների սահմանները խնդրի պայմաններին համապատասխան՝ ավելացնելով թվի զույ– գությունը ստուգող պայմանը: Սակայն դա ճիշտ կլինի, եթե առաջին ընտրվող թվի արժեքը՝ b_{1n}-ը լինի զույգ: Իսկ նման ենթադրություն անելը անհիմն է, հետևաբար, կարելի է առաջարկել հետևյալ երկու մոտեցումները:

1) Մինչև մատրիցի տարրերի ստուգումը կարելի է որոշել հերթով առաջին զույգ թիվը, որի հետ պետք է սկսենք համեմատել հաջորդ թվերը: Տվյալ գործընթացը իրականացվում է առանձին ներդրված ցիկլերով՝ *օժանդակ անկյունագծի վրա կամ նրանից ներքև՝ մինչև զույգ թվի հանդիպելը:* Այդպիսի տարր հանդիպելուց հետո նրա արժեքը ընդունվում է որպես փոքրագույն, և հաջորդ ցիկլերում այն համեմատվում է մնացած տարրերի հետ: Սույն ալգորիթմի

նկարագրությունը թողնենք ձեզ` ինքնավարժանքի միտումով, իսկ մենք անցնենք երկրորդ տարբերակի նկարագրմանը, որի



Նկ.4.26. Խնդիր 48 լուծման ալգորիթմի բլոկ-սխեմա:

բլոկ-սխեման բերված է նկ.4.26-ում:

2) Եթե ձեզ հաջողվի բարեհաջող ավարտել խնդրի լուծման առաջին տարբերակի նկարագրությունը, ապա արժանավույն կգնահատեք նաև ստորև ներկայացվող երկրորդ տարբերակը: Որոշելով պահպանել Ա39-ի պատկերը՝ անհրաժեշտ է հոգալ առաջին զույգ թվի ընտրության մասին, որի հարցը լուծվում է հետևյալ ձևով: 'Տող առ տող' երթևեկելու պայմաններում չկարողանալով առաջին՝ b_{1n} տարրի արժեքը ընդունել որպես առաջին փոքրագույն արժեք, մեզ մնում է ընթացքում զույգ թիվ հանդիպելիս (բլ.6) պարզել՝ այն առաջի՞նն է, թե ոչ: Եթե 'այո', ապա k և p փոփոխականներին համապատասխանաբար վերագրվում են այդ տարրի առաջին և երկրորդ ինդեքսների արժեքները (բլ.9), հակառակ պատասխանի դեպքում թիվը համեմատվում է իրեն նախորդող թվերից փոքրագույնի՝ b_{kp}-ի հետ (բլ.8):

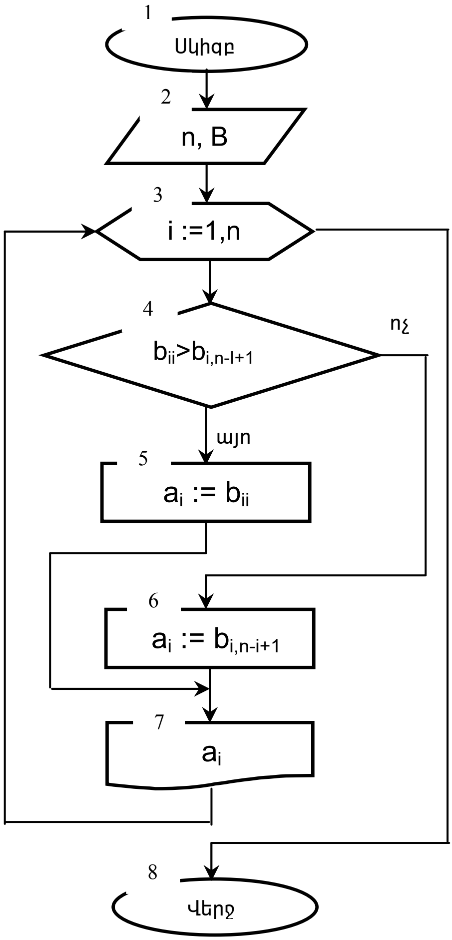
Ինչպես երևում է գծանկարից, զույգ թվի առաջնությունը պարզելու նպատակով ալգորիթմի սկզբնամասում, որպես փոքրագույն արժեքով տարրի, տողի համար ընդունված է k=0 արժեքը (բլ.3): Այն կարող է հետագայում ուղղվել՝ առաջին զույգ թվին կամ նրանից ավելի փոքրին հանդիպելիս (բլ.9): Եթե օժանդակ անկյունագծի վրա կամ նրանից ներքև զույգ թիվ չհանդիպենք, կարելի է պատասխանել, որ խնդիրը լուծում չունի (բլ.12), այլապես արտածվում է ինդեքսների k և p արժեքներով մատրիցի b_{kp} տարրը (բլ.11):

Այսպիսով, դուք ծանոթացաք մեկ խնդրի լուծման երկու տարբերակներին. մեկը բանավոր ընթերցմամբ, մյուսը՝ ձևակերպված բլոկ-սխեմայի տեսքով: Թե ո՞րն է նախընտրելի, որոշեք ինքներդ, իսկ մենք անցնենք հաջորդ խնդրին, որտեղ գործողությունները զարգանում են անկյունագծերի վրա:

<u>Խնդիր 49:</u> Տրված է n×n չափողականությամբ $B = \left\| b_{ij} \right\|$ քառակուսի մատրից ($i,j = \overline{1,n}$): Պահանջվում է ստանալ նոր՝ A վեկտոր, որի a_i տարրը պետք է հավասար լինի մատրիցի i–րդ տողի վրա գտնվող գլխավոր և օժանդակ անկյունագծերի տարրերից մեծագույնին:

Տվյալ խնդրի առաջարկը պայմանավորված է ոչ այնքան նրա ալգորիթմական յուրահատկություններով, որքան ինդեքսների առանձնահատուկ գործածմամբ: Չէ՞ որ գլխավոր անկյունագծի վրա գտնվող տարրերի երկու ինդեքսների արժեքները միմյանց հավասար են և սովորական b_{ij}

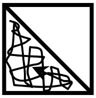
գրառման փոխարեն կարելի է գրել b_{ii}, իսկ օժանդակ անկյունագծինը՝ $b_{i,n-i+1}$: Այսինքն ինդեքսները ներկայացնելու համար կարելի է օգտվել մեկ փոփոխականի ծառայությունից:



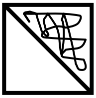
Նկ.4.27. Խնդիր 49 լուծման ալգորիթմի բլոկ-սխեմա:

Նկ.4.27-ում ներկայացված է վերը նշված դիտողություններով հանդերձ տվյալ խնդրի լուծման ալգորիթմի բլոկ–սխեման, որի լրացուցիչ պարզաբանումը, թվում է, ավելորդ է: Նշենք միայն, որ ցիկլի առաջին կատարման ընթացքում համեմատվում են առաջին տողի ծայրային տարրերը: Դրանից հետո տողի i համարը ’մոդիֆիկացիա’ բլոկի շնորհիվ աճում է մեկով և կատարվում է անցում մատրիցի հաջորդ տողը: Ցիկլում կատարվելիք մյուս գործողությունները, համոզված ենք, պարզաբանման կարիք չունեն, որովհետև պարզ են և դյուրին: Միայն ավելացնենք, որ բերված սխեմայում ստեղծվող վեկտորի տարրերի արտածումը համատեղված է նրանց որոշման հետ: Նույնը կարելի է կատարել առանձին ցիկլով հաշվարկման ցիկլի աշխատանքն ավարտելուց հետո, սակայն այդ կհանգեցնի ժամանակի կորստի:

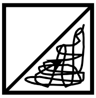
Ստորև ամփոփված ձևով ներկայացնենք քառակուսի մատրիցների՝ անկյունագծերով տրոհման արդյունքում առաջացած եռանկյունաձև տիրույթներում ինդեքսների փոփոխման սահմանները, որտեղ i–ով նշանակված է տողի համարը, իսկ j–ով՝ սյան:



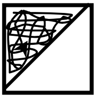
Գլխավոր անկյունագծից ներքև կամ նրա վրա գտնվող տարրերի ինդեքսների սահմաններն են. i=$\overline{1,n}$, իսկ j=$\overline{1,i}$: Միայն անկյունագծից ներքև գտնվող տարրերի համար՝ i=$\overline{2,n}$, իսկ j=$\overline{1,i-1}$:



Գլխավոր անկյունագծից վերև կամ նրա վրա գտնվող տարրերի ինդեքսների սահմաններն են. i=$\overline{1,n}$, իսկ j=$\overline{i,n}$: Միայն անկյունագծից վերև գտնվող տարրերի համար՝ i=$\overline{1,n-1}$, իսկ j=$\overline{i+1,n}$:



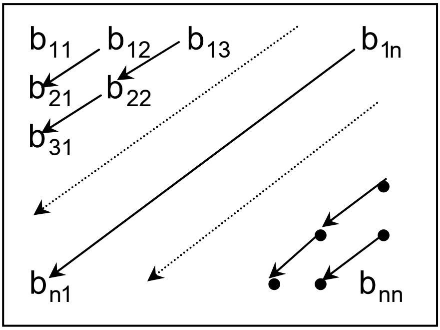
Օժանդակ անկյունագծից ներքև կամ նրա վրա գտնվող տարրերի ինդեքսների սահմաններն են. i=$\overline{1,n}$, իսկ j=$\overline{n-i+1,n}$: Միայն անկյունագծից ներքև գտնվող տարրերի համար՝ i=$\overline{2,n}$, իսկ j=$\overline{n-i+2,n}$:



Օժանդակ անկյունագծից վերև կամ նրա վրա գտնվող տարրերի ինդեքսների սահմաններն են.i=$\overline{1,n}$, իսկ j=$\overline{1,n-i+1}$: Միայն անկյունագծից վերև գտնվող տարրերի համար՝ i=$\overline{1,n-1}$, իսկ j=$\overline{1,n-i}$:

Մատրիցների թեման ավարտելուց առաջ կրկին անդրադառնանք նրանց վրայով անցման երթուղիների ընտրման հարցին: Ժամանակին մենք ասացինք, որ ամենատարածված և հիմնականում կիրառվող երթուղիներն են. ’տող առ տող’ և ’սյուն առ սյուն’ հերթականությունները: Սակայն երբեմն անհրաժեշտություն է առաջանում ընտրել այլ երթուղիներ, ինչը ալգորիթմներում առաջացնում է երթևեկության կազմակերպման հետ կապված զգալի բարդություններ, որոնք գերազանցում են հաշվարկող հատվածի բարդույթները: Դիտարկենք նման մեկ օրինակ:

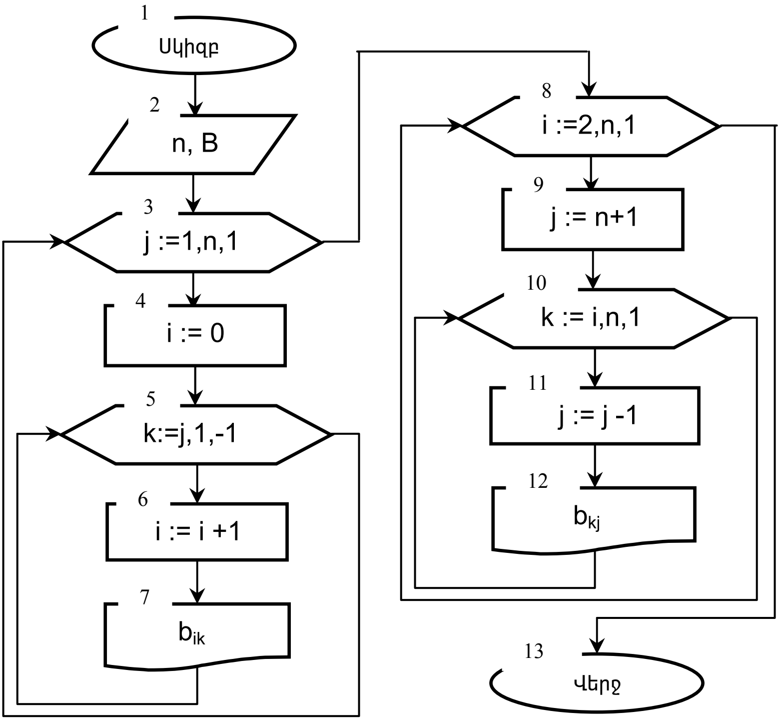
<u>Խնդիր 50:</u> Տրված է n×n չափողականությամբ $B=\|b_{ij}\|$ քառակուսի մատրից ($i,j=\overline{1,n}$): Պահանջվում է մատրիցի տարրերը արտածելէ սկսած վերևի ձախ անկյունից օժանդակ անկյունագծին զուգահեռ գծերով՝ մինչև ներքևի աջ անկյունը, ինչպես ներկայացված է նկարում.



Ըստ նկարի, մատրիցի տարրերի արտածման հերթականությունը հետևյալն է. b_{11}, b_{12}, b_{21}, b_{13}, b_{22}, b_{31}, ..., b_{1n}, ..., b_{n1}, ..., b_{nn}: Տվյալ գործընթացը իրականացնելու համար կարելի է մատրիցի տարրերը արտածել երկու մասով. սկզբում այն տարրերը, որոնք տեղադրված են օժանդակ անկյունագծից վերև, ներառյալ անկյունագիծը, իսկ հետո այդ անկյունագծից ներքև տեղադրված տարրերը:

Մինչև ալգորիթմի նկարագրության հետ ծանոթանալը առաջարկում ենք ինքնուրույն վերլուծել նրա բլոկ–սխեման, որը բերված է նկ.4.28-ում: Համաձայն արտածման սխեմայի, վերևի տիրույթի բոլոր ուղիղ հատվածները սկսվում են առաջին տողից, տեղափոխվելով մեկ սյուն առաջ: Հետևվաբար, որպես ցիկլի պարամետր կարելի է ընտրել սյան j համարը 1-ից n սահմաններով (բլ.3): Յուրաքանչյուր հատվածի վրայով անցնելիս նկատում ենք, որ այդ հատվածի վրա տեղադրված տարրերի տողի համարը աճում է, իսկ սյան համարը նվազում: Ընդ որում տողի համար փոփոխման սահմանն է 1-ից j, իսկ սյան համար. j-ից 1: Քանի որ այս երկու պարամետրերը փոփոխվելու են միաժամանակ, այդ գործընթացը պետք է կազմակերպել մեկ ցիկլով: Գծանկարում բերված տարբերակում այդպիսի ցիկլի պարամետր է ընտրած կրկին սյան k համարը (բլ.5), իսկ տողի i համարը փոփոխվում է նույն ցիկլում մեկ գումարելով (բլ.6):

Նման դատողությունների հիման վրա է կառուցված նաև օժանդակ անկյունագծից ներքև տեղադրված տարրե–րի արտածման ցիկլը, որը կազմված է 8...12 բլոկներից: Ինչպես երևում է խնդրում բերված երթևեկության սխեմա–յից, մատրիցի նշված տիրույթում բոլոր հատվածները սկսվում են մատրիցի վերջին սյունից և ավարտվում վերջին տողում: Այս հանգամանքը մեզ հուշում է, որ տողի համարը պետք է



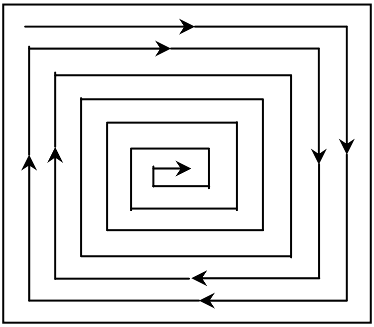
Նկ.4.28. Խնդիր 50 լուծման ալգորիթմի բլոկ-սխեմա:

հանդիսանա ցիկլի պարամետր (բլ.8): Մեկ հատվածով շարժվելիս նկատում ենք, որ այդ հատվածի վրա տեղադրված տարրերի տողի համարը աճում է, իսկ սյան համարը նվազում: Ընդ որում տողի համար փոփոխման սահմանն է i-ից n, իսկ սյան համար. n-ից i: Քանի որ այս երկու պարամետրերը փոփոխվելու են միաժամանակ, այդ գործընթացը կազմակերպված է մեկ ցիկլով: Գծանկարում բերված տարբերակում այդպիսի ցիկլի պարամետր է ընտրած կրկին տողի k համարը (բլ.10), իսկ սյան j համարը փոփոխվում է նույն ցիկլում` մեկ հանելով (բլ.11):

Տեսնու՞մ եք, թե ինչ ծավալուն ալգորիթմ ստեղծվեց պահանջվող, ոչ ավանդական, երթուղով տվյալները ար– տածելու համար: Դիտարկվածից շատ ավելի բարդ երթու– ղիներ ձեզ կառաջարկենք անցնել վարժությունները լու–

ծելիս և համոզված ենք, որ դուք կհաղթահարեք բոլոր դժվարությունները: Իսկ առայժմ առաջարկում ենք ծանոթանալ հաջորդ խնդրին:

<u>Խնդիր 51:</u> Տրված է n×n չափողականությամբ $B = \|b_{ij}\|$ քառակուսի մատրից ($i,j = \overline{1,n}$): Պահանջվում է մատրիցի տարրերը արտածել գալարաձև երթուղով, սկսած վերևի ձախ անկյունից, շարժվելով դեպի աջ և ավարտելով մատրիցի կենտրոնում, ինչպես ներկայացված է նկարում.



Փորձենք դատողությունների հիման վրա պարզել այն օրինաչափությունները, որոնք ընդհանուր են նշված ուղիների համար: Այն, որ երթուղու չորս ուղիները ընթանում են միևնույն օրենքով, և, հետևաբար, պետք է նկարագրվեն նույն տիպի ցիկլերով, ակնհայտ է: Հատվածները բնորոշվում են իրենց սահմանային կետերի կոորդինատներով: Նկատելի է, որ յուրաքանչյուր հատվածից հետո ճանապարհը թեքվում է դեպի աջ: Ընդ որում այլևս տվյալ հատվածին վերադարձ չկա: Դա նշանակում է, որ այդ հատվածով որոշվող սահմանը կարելի է տեղաշարժել մեկ միավորով:

Կարելի է ընդունել, որ երթևեկությունը ընթանում է վերևի (‘վ’) և ներքևի (‘ն’) սահմանային տողերի կամ աջ (‘ա’) և ձախ (‘ձ’) սահմանային սյուների վրայով: Երթը սկսվում է առաջին տողից, որը հանդիսանում է վերևի առաջին սահմանը (վ=1): Հասնելով տողի վերջ և թեքվելով աջ՝ դեպի ներքևի սահմանը, կարող ենք վերևի սահմանը իջեցնել երկրորդ տող (վ:=վ+1): Իջնելով աջ սահմանային սյունով ներքև (ա=n) և թեքվելով աջ՝ դեպի ձախ սահմանը, վստահորեն կարող ենք աջ սահմանը տեղաշարժել մեկ սյունով դեպի ձախ (ա:=ա-1): Շարժվելով ներքևի՝ ն:=n սահմանային տողով, հասնելով ձախ եզրին (ձ=1) և թեքվելով աջ՝ դեպի վերևի սահմանը, ներքևի սահմանը կարող ենք բարձրացնել մեկ տողով վերև (ն:=ն–1): Երթևեկության այս հատվածը կընթանա մինչև երկրորդ տող, քանի որ վերևի սահմանը արդեն իջել է մեկ տողով ներքև:

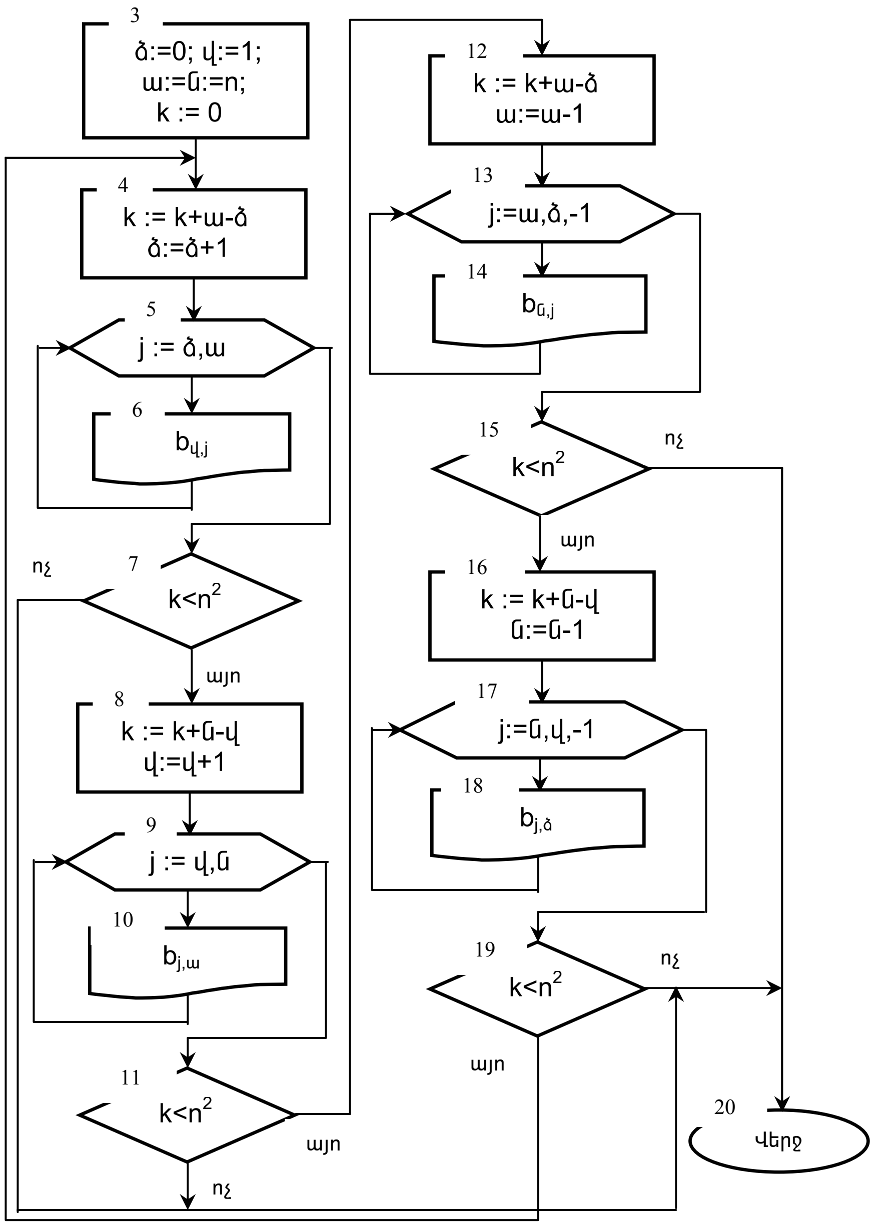
Այսպիսով, մեկ պտույտ կատարելուց հետո կհայտնը–վենք ոչ թե b_{11}, այլ b_{21} տարրի վրա: Կրկնելով վերը նըկա-րագրված երթուղին՝ հաջորդ պտույտը կկատարվի փոփոխ-ված սահմանների ներքո: Որի ընթացքում սահմանները կրկին կսեղմվեն:

Որպեսզի չզբաղվենք անպտուղ գուշակումներով, թե քանի՞ պտույտից հետո կավարտվի երթը, կամ արդյո՞ք երթևեկության ընթացքում իրականացվում են ամբողջ քա-նակությամբ պտույտներ, երթևեկելու ընթացքում հաշվենք արտածվող տարրերի k քանակը և յուրաքանչյուր հատված անցնելուց հետո համեմատենք այն մատրիցի տարրերի քանակի հետ, որը կազմում է՝ n^2: Այսպես կազմակերպված ալգորիթմը հնարավոր է կիրառել նաև ուղղանկյուն մատ-րիցների նկատմամբ:

Նկարագրված ալգորիթմի բլոկ-սխեման բերված է նկ. 4.29-ում: Տեղի սղության պատճառով գծանկարում բացա–կայում են առաջին երկու բլոկները, որտեղ նշված են՝ սկիզբը և մատրիցի ներմուծման գործողությունը: Դրանց բացակայությունը, հուսով ենք, չի խանգարի ալգորիթմի աշխատանքի ընկալմանը:

Եթե ցանկություն կունենաք երթևեկությունը սկսել մատրիցի որևէ այլ անկյունից, ապա բավական է համա–պատասխան եզրագծով երթևեկելու նկարագրությունը տե-ղափոխել առաջին տեղ, զբաղեցնելով 4-ից 6 բլոկները և չխախտելով բոլոր ցիկլերի հարաբերական դիրքը: Տվյալ դեպքում, բնական է, կփոխվի միայն ձախ եզրի սկըզբ-նական արժեքը և այն սահմանի, որից մտադրվել եք սկսել ձեր երթը:

Այսպես կարելի է անվերջ քննարկել զանազան խնդիրների օրինակներ, լուծել նրանց տարբեր եղանակ–ներով, համեմատել միմյանց հետ և ընտրել լավագույնը: Սակայն ինչքան խնդիրը բարդ է, այնքան լուծման տար-բերակները զանազան և այս պայմաններում անհնար է դառնում բոլոր խնդիրների վերլուծումը մեկ ձեռնարկի սահմաններում: Տվյալ աշխատությունը ուսումնասիրելիս, դուք պետք է նկատեիք, որ բովանդակությամբ խնդիրների ցանկը այնքան էլ ընդարձակ չէր: Պարզապես, միևնույն



Նկ.4.29. Խնդիր 51 լուծման ալգորիթմի բլոկ-սխեմա:

խնդիրները առաջարկվել է լուծել տվյալների տարբեր կառույցների վրա՝ սկսած պարզ փոփոխականներից և վերջացրած միաչափ ու երկչափ զանգվածներով: Որքան տվյալների կառույցը բարդանում էր, այնքան դժվար էր լինում ընտրել երթևեկելու ուղին, որը և ներկայացնում էր ալգորիթմի հիմնական բարդությունը: Սակայն մենք միասին հաղթահարեցինք այդ ճանապարհներին հանդիպած արգելքները: Եթե մեզ հաջողվեց ինչ որ չափով ձեզ հետաքրքրել և ներգրավել ծրագրավորման հիասքանչ աշխարհ, ապա կհամարենք, որ հասանք մեր առաջ դրված նպատակին:

ՎԱՐԺՈՒԹՅՈՒՆՆԵՐ

1. Տրված է $X(x_1,x_2,\ldots,x_n)$ միաչափ թվային զանգվածը: Պահանջվում է հաշվել և արտածել մինչև զանգվածի վերջին զրոն տեղադրված դրական թվերի արտադրյալը: Եթե զանգվածը զրո արժեքով տարր չի պարունակում, ոչինչ չհաշվել:

2. Տրված է $X(x_1,x_2,\ldots,x_n)$ միաչափ թվային զանգվածը: Պահանջվում է հաշվել և արտածել զանգվածի առաջին և վերջին զրոների միջև տեղադրված դրական թվերի գումարը: Եթե զանգվածը պարունակում է ընդամենը մեկ զրո, հաշվել այդ զրոյին հաջորդող բոլոր տարրերի գումարը: Եթե զանգվածը զրո արժեքով տարր չի պարունակում, ոչինչ չհաշվել:

3. Տրված է $X(x_1,x_2,\ldots,x_n)$ միաչափ թվային զանգվածը: Պահանջվում է հաշվել և արտածել այն տարրերի գումարը, որոնք զանգվածում չեն կրկնվում:

4. Տրված է N բնական թիվը: Պահանջվում է պարզել, թե որքա՞ն տարբեր թվանշաններից է կազմված այդ թիվը:

5. Տրված է $X(x_1,x_2,\ldots,x_n)$ միաչափ թվային զանգվածը: Պահանջվում է հաշվել և արտածել զանգվածում պարունակվող պարզ թվերի գումարը:

6. Տրված է $X(x_1,x_2,\ldots,x_n)$ միաչափ թվային զանգվածը: Պահանջվում է հաշվել և արտածել զանգվածում պարունակվող պարզ թվերից մեծագույնը:

7. Տրված է $X(x_1,x_2,\ldots,x_n)$ միաչափ թվային զանգվածը: Պահանջվում է այնպես վերադասավորել զանգվածի տարրերը, որ սկզբից տեղադրվեն բացասական, հետո զրոյական և

վերջում` դրական արժեքներով տարրերը: Միևնույն նշան ունեցող տարրերի հարաբերական դիրքը պետք է պահպանվի:

8. Տրված է $X(x_1,x_2,\ldots,x_n)$ միաչափ թվային զանգվածը: Պահանջվում է որոշել զանգվածի այն տարրերի քանակը, որոնք բավարարում են $2^k<x_k<2^{k+1}$ պայմանին, որտեղ $k=1,2,\ldots,n$:

9. Տրված են $X(x_1,x_2,\ldots,x_n)$ և $Y(y_1,y_2,\ldots,y_m)$ միաչափ թվային զանգվածները: Y զանգվածի այն տարրերը, որոնք չեն հանդիպում X-ում, ավելացնել X զանգվածի վերջում:

10. Տրված են $X(x_1,x_2,\ldots,x_n)$ և $Y(y_1,y_2,\ldots,y_n)$ միաչափ թվային զանգվածներ: Հայտնի է, որ X զանգվածի տարրերը կարգավորված են ըստ նվազման: Պահանջվում է տեղադրել Y զանգվածի տարրերը X-ի մեջ այնպես, որ X զանգվածի ընդհանուր կարգավորվածությունը չխախտվի:

11. Տրված է $X(x_1,x_2,\ldots,x_n)$ միաչափ թվային զանգվածը, որտեղ $n=m^2$: Պահանջվում է վեկտորը վերածել երկչափ քառակուսային մատրիցի, որի առաջին տողում անհրա–ժեշտ է գրանցել վեկտորի առաջին m տարրերը $(x_1,\ldots,x_m)$, երկրորդ տողում` $(x_{m+1},\ldots,x_{2m})$ տարրերը և այլն: Վերջին տողում, բնական է, կգրանցվեն $(x_{n-m+1},\ldots,x_n)$ տարրերը:

12. Տրված է $A=\|a_{ij}\|$ $(i=\overline{1,n};\ j=\overline{1,m})$ մատրիցը: Պահանջ–վում է հաշվել մատրիցի այն տարրերի միջին քառակուսայինը, որոնց արժեքը բաժանելիս 3-ի կմնա 2 մնացորդ:

13. Տրված է $A=\|a_{ij}\|$ $(i=\overline{1,n};\ j=\overline{1,m})$ մատրիցը: Պահանջվում է կարգավորել մատրիցը ըստ k-րդ տողի արժեքների աճման:

14. Տրված է $A=\|a_{ij}\|$ $(i=\overline{1,n};\ j=\overline{1,m})$ մատրիցը: Պահանջվում է ստանալ և արտածել նոր B վեկտոր, որի b_i տարրը հավասար լինի մատրիցի i–րդ տողում վերջին մեծագույն տարրի զբաղեցրած դիրքին:

15. Տրված է $A=\|a_{ij}\|$ $(i=\overline{1,n};\ j=\overline{1,m})$ մատրիցը: Պահանջ–վում է ստանալ և արտածել նոր B վեկտոր, որի b_i տարրը հավասար լինի մատրիցի i–րդ տողի պարզ թվերից փոքրագույնին:

16. Տրված է A=$\|a_{ij}\|$ (i=$\overline{1,n}$; j=$\overline{1,m}$) մատրիցը: Պահանջվում է հեռացնել մատրիցից այն տողերը, որոնք կարգավորված չեն ըստ արժեքների՝ ոչ աճման, ոչ նվազման:

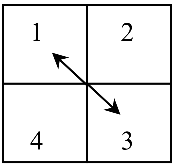
17. Տրված է A=$\|a_{ij}\|$ (i,j=$\overline{1,n}$) քառակուսային մատրիցը: Պահանջվում է՝ գլխավոր անկյունագծից ներքև կամ նրա վրա գտնվող տարրերից կազմել նոր վեկտոր, ուր նշված տարրերը պետք է արտագրվեն տող առ տող:

18. Տրված է A=$\|a_{ij}\|$ (i,j=$\overline{1,n}$) քառակուսային մատրիցը: Պահանջվում է կազմել նոր վեկտոր, որը կպարունակի մատրիցի օժանդակ անկյունագծից վերև գտնվող այն տարրերը, որոնց արժեքը չի կրկնվում օժանդակ անկյու–նագծից ներքև:

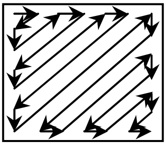
19. Տրված է A=$\|a_{ij}\|$ (i,j=$\overline{1,n}$) քառակուսային մատրիցը: Պահանջվում է՝ գլխավոր անկյունագծից վերև կամ նրա վրա գտնվող տարրերից կազմել նոր վեկտոր, որտեղ նշված տարրերը դասավորված լինեն ըստ նրանց ինդեքսների գումարի աճման:

20. Տրված է A=$\|a_{ij}\|$ (i,j=$\overline{1,n}$) քառակուսային մատրիցը: Պահանջվում է տեղերով փոխանակել մատրիցի ձախ և աջ կեսերը: Եթե n-ի արժեքը կենտ է, կենտրոնական սյունը թողնել անփոփոխ:

21. Տրված է A=$\|a_{ij}\|$ (i,j=$\overline{1,n}$) քառակուսային մատրիցը:



Պահանջվում է տեղերով փոխանակել մատրիցի կենտ համարի ենթամատ–րիցները, ինչպես նշված է նկարում: Եթե n-ի արժեքը կենտ է, ապա կենտրոնական սյան և տողի վրա գտնվող տարրերը թողնել տեղում:

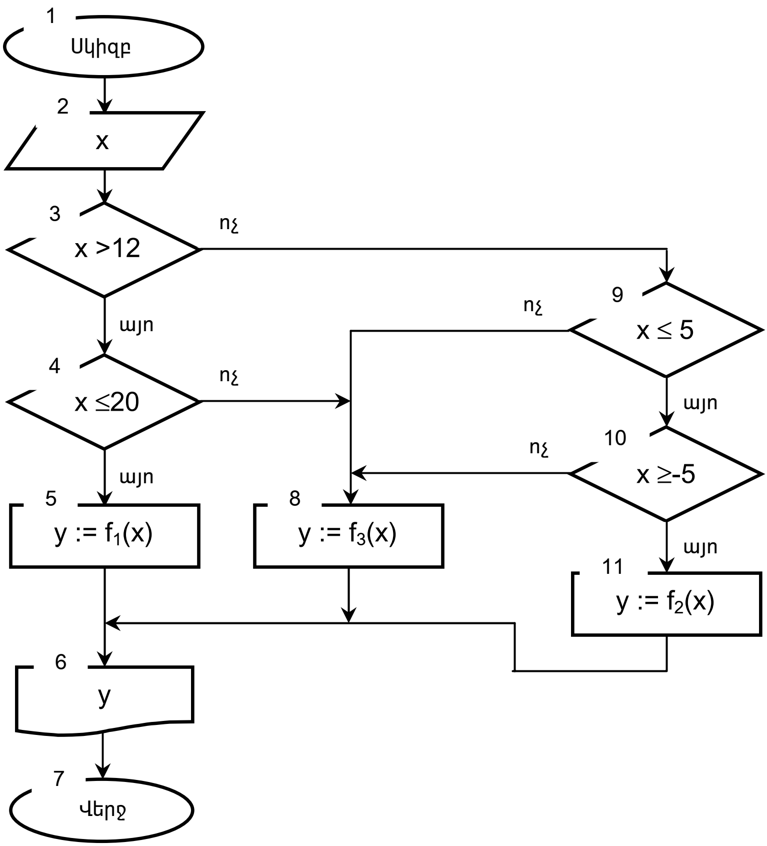


22.Տրված է A=$\|a_{ij}\|$ (i,j=$\overline{1,n}$) քառակուսային մատրիցը: Պահանջվում է մատրիցի տարրերը արտածել այն հերթականությամբ, ինչպես նշված է նկարում, այսինքն, սկսել վերևի ձախ անկյունից և ավարտել ներքևի աջ անկյունում (a_{11}, a_{12}, a_{21}, a_{31}, a_{22}, a_{13}, ..., $a_{n-1,n}$, $a_{n,n-1}$, a_{nn}):

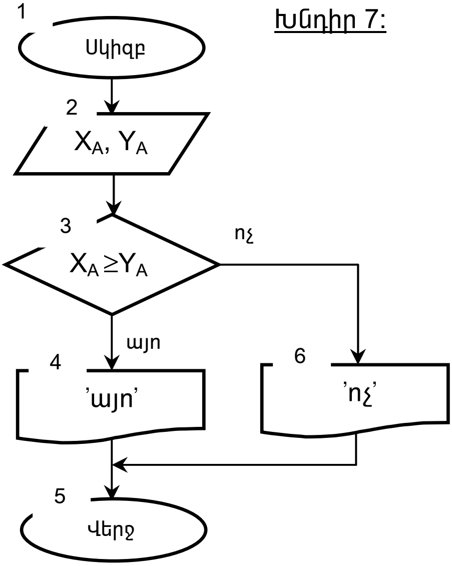
ՀԱՎԵԼՎԱԾ

Այս բաժնում դուք կգտնեք այն ալգորիթմների բլոկ–սխեմային նկարագրությունները, որոնք ժամանակին ներկայացվել են բանավոր ձևով և առաջարկվել ձեզ ինքնավարժանքի նպատակով:

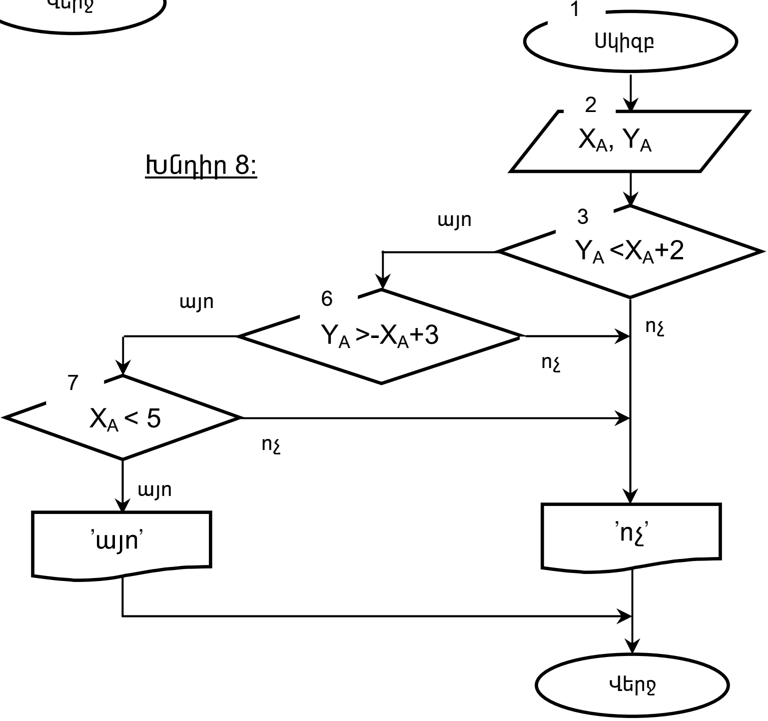
<u>Խնդիր 4:</u>

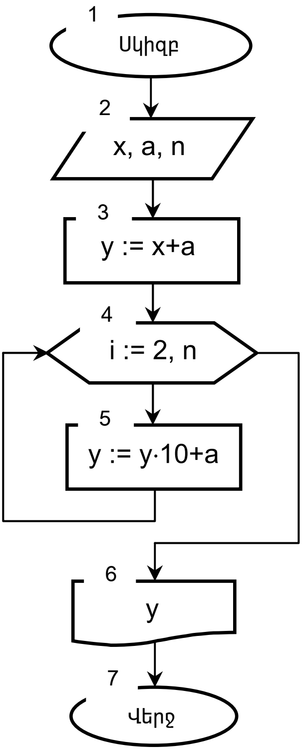


Խնդիր 7:

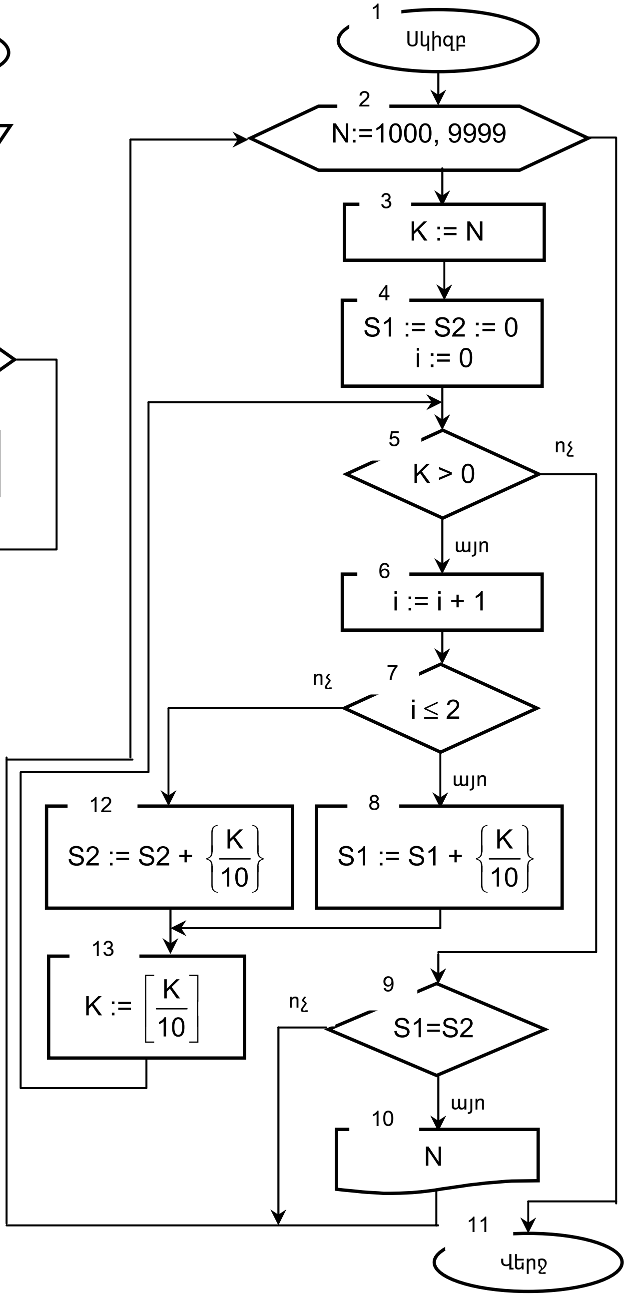


Խնդիր 8:



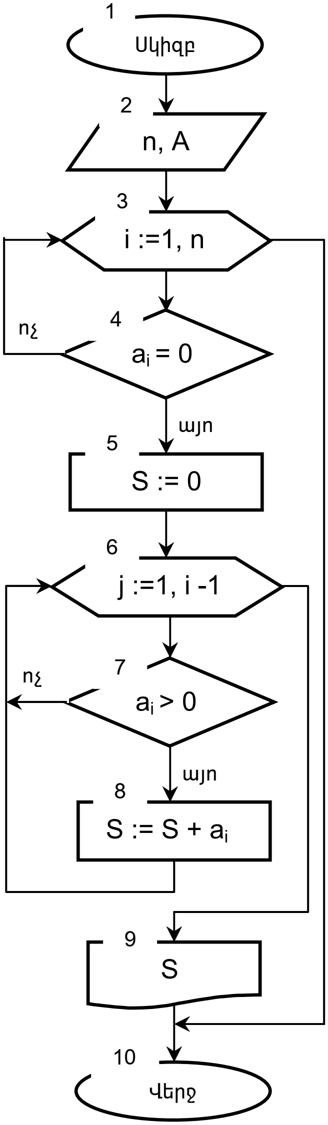


Խնդիր 19:

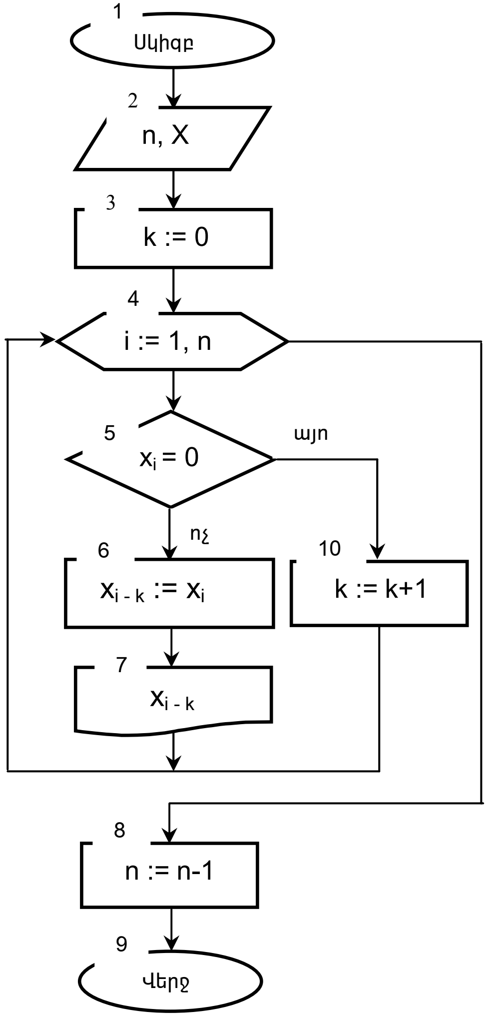


Խնդիր 27:

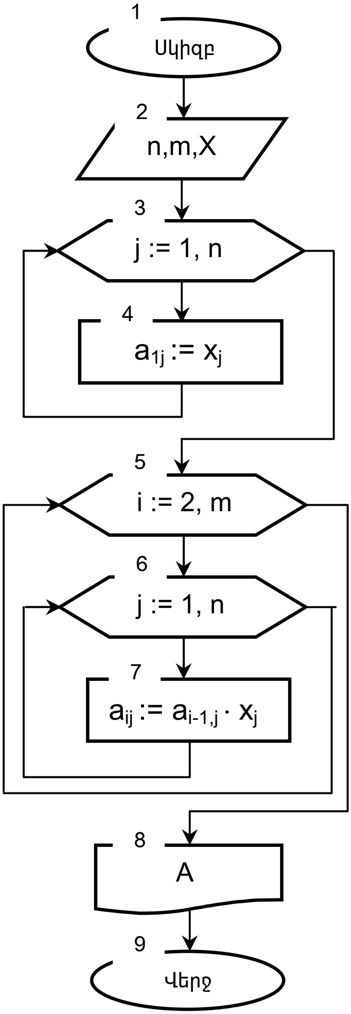
Խնդիր 32:



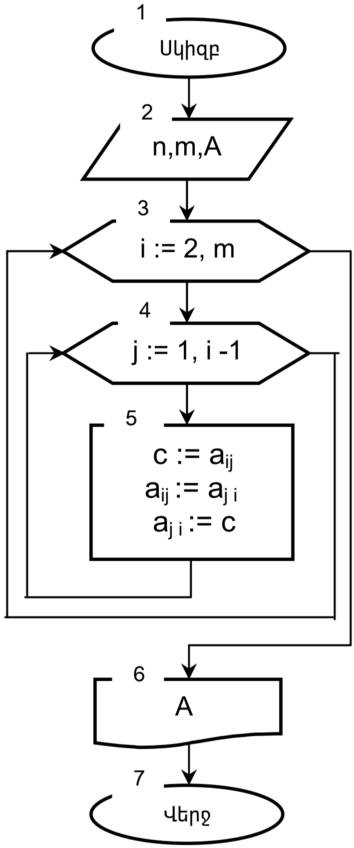
Խնդիր 34:



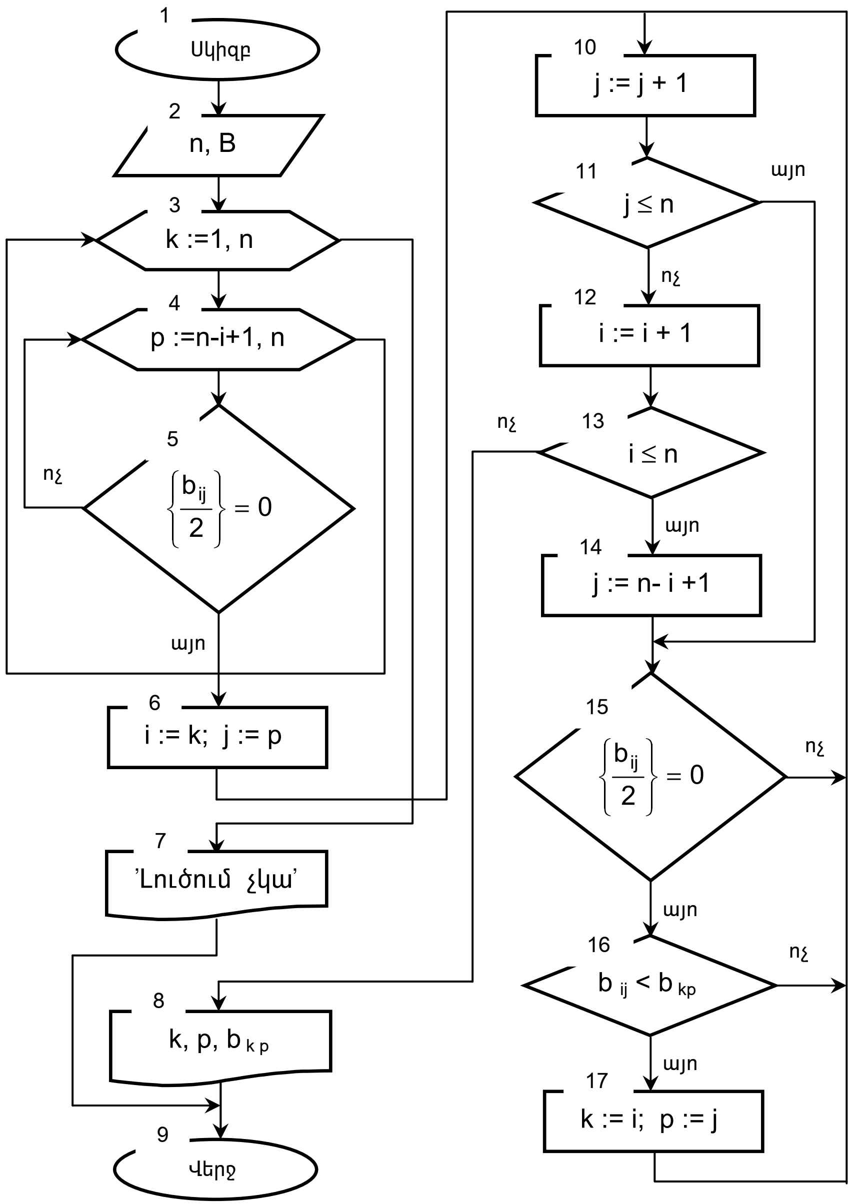
Խնդիր 44:



Մատրիցի տրանսպոնացում:



Խնդիր 48:



1
Սկիզբ
2
n, B
3
k :=1, n
4
p :=n-i+1, n
5
$\left\{\frac{b_{ij}}{2}\right\} = 0$
ոչ
այո
6
i := k; j := p
7
'Լուծում չկա'
8
k, p, b_{kp}
9
Վերջ
10
j := j + 1
11
j ≤ n
այո
ոչ
12
i := i + 1
13
i ≤ n
ոչ
այո
14
j := n- i +1
15
$\left\{\frac{b_{ij}}{2}\right\} = 0$
ոչ
այո
16
$b_{ij} < b_{kp}$
ոչ
այո
17
k := i; p := j

ԳՐԱԿԱՆՈՒԹՅՈՒՆ

1. Демидович Б.П., Марон И.А. Основы вычислительной математики. М., “Наука”, 1970.

2. С.А.Абрамов, Г.Г.Гнездилова и др. Задачи по программированию. М., “Наука”, 1988.

3. Գասպարյան Է.Գ., Այվազյան Յու.Ա.: Էլեկտրոնային հաշվիչ մեքենաների կիրառումը ուսումնական պրոցեսում: Երևան, «Լույս», 1990:

4. Ռ.Վ.Աղգաշյան, Ա.Հ.Առաքելյան, Ս.Ս.Ավետիսյան, Ս.Վ.Դանիելյան: Քոմփյութերների կիրառում և ծրագրավորում (խնդիրների ժողովածու): ՀՊՃՀ, 1998:

5. Ա.Հ.Առաքելյան, Ռ.Վ.Աղգաշյան, Ս.Ս.Ավետիսյան, Ս.Վ.Դանիելյան, Լ.Ա.Մանուկյան: ՀՊՃՀ ընդհանուր կրթության պետական ամփոփիչ քննության քննահարցերի շտեմարանի խնդիրների լուծման ուղեցույց: Երևան, «Նոյյան Տապան», 1996:

6. Ա.Աբրահամեյան–Եու.Բարոեան, Գ.Մարկարով–Կ.Մարկարեան: Ինֆորմատիկայի հիմունքներ: Վենետիկ–Ս.Ղազար, Մխիթարեան հրատարակչություն, 1991:

7. С.А.Немнюгин. Turbo-Pascal. Учебник. С.-П., “Питер”, 2000.