Ալգորիթմի մշակումը՝ որպես խնդրի լուծման կորևորագույն փուլ

Օհանյան Հեղինե
Գյումրեցյան Արմենուհի
Քանարյան Գայանե

Հանգուցային բառեր. *խնդրի լուծման գործընթացի հիմնական փուլերը, խնդրի դրվածքի սահմանում, խնդրի մաթեմատիկական դրվածքի ձևակերպում, խնդրի լուծման մեթոդի ընտրություն, ալգորիթմների մշակում, ծրագրավորման տեխնոլոգիա, մասնավոր (անհատական) նպատակների մեթոդ, վերընթացի մեթոդ, էվրիստիկ ալգորիթմ*

Ծրագրավորումը վաղուց արդեն դարձել է ամենապահանջված և տարածված մասնագիտություններից մեկն աշխարհում: Սակայն և՛ դասավանդման, և՛ առօրյա պրակտիկայում հաճախ, ցավոք, գերիշխում է մակերեսային, դեղատոմսային մոտեցում ծրագրավորման ուսումնառության և կիրառման նկատմամբ, երբ ալգորիթմական լեզվի տիրապետման տարրական գրագիտությունը ներկայացվում է որպես մասնագիտության գրեթե սպառիչ բովանդակություն: Խզումը մաթեմատիկոս-ծրագրավորողի ստացած բարձր մաթեմատիկական լիցքի և ծրագրի կազմման ամենօրյա սովորական աշխատանքի միջև որոշ մասնագետների մոտ ստեղծում է իրենց մաթեմատիկական գիտելիքների որոշակի արժեզրկման զգացում:

«Պատկերավոր ասած, ծրագրավորման դասավանդման մեր ներկայիս մեթոդները սահմանափակվում են նրանով, որ մենք առաջին կուրսի ուսանողներին տալիս ենք քերականության դասագիրք ու բառարան և ասում նրանց, որ այսուհետ իրենք մեծ գրողներ են: Իսկապես, մինչև այժմ ես չեմ հանդիպել ոչ ոճաբանության ձեռնարկ, ոչ ծրագրերի անթոլոգիա: Եվ այստեղ, և այնտեղ նախապատվությունը տրվում է հակիրճությանը: Վերլուծելով լավ գրական լեզվի ուսուցման մեթոդը՝ շարադրությունների գրելը, զեկույցների և ելույթների կազմելը, ըստ որոնց գնահատվում է ուսանողների նվաճումները, մենք միանգամից կնկատենք, որ ծրագրավորման դասավանդման մեջ նման մոտեցումը բացակայում է: Ցավոք, քիչ ծրագրավորողներ իրոք ունեն այն, ինչ ես անվանում եմ ոճ, և ձգտում են ստեղծել իսկական արվեստի գործեր ծրագարավորման ոլորտում» [8]:

Այն լեզուն, որով ուսանողին սովորեցնում են արտահայտել իր մտքերը, խորը ազդեցություն է գործում նրա մտածողության հմտությունների և ստեղծագործական ընդունակությունների վրա: Ներկայումս կիրառվող ծրագրավորման լեզուների կառուցվածքները և հատկությունները հաճախ չեն ենթարկվում համոզիչ ու տրամաբանական բացատրության, չեն համապատասխանում համակարգված դատողություններին սովոր մտածողությանը: Արդյունքում ծրագրավորման լեզուներում տիրող անկանոնություններն անմիջականորեն ազդում են ուսանողների մոտ ձևավորվող ծրագրավորման ոճի վրա:

Ծրագրավորման «արդյունավետ ոճ» հասկացությունը պարզաբանելու համար ուսումնասիրենք համակարգչի վրա խնդրի լուծման գործընթացի հիմնական փուլերը [6].

- խնդրի դրվածքի սահմանում,
- խնդրի մաթեմատիկական դրվածքի ձևակերպում,
- խնդրի լուծման մեթոդի ընտրություն,
- ալգորիթմի մշակում,
- ծրագրի կազմում,
- ծրագրի թեստավորում և կարգաբերում,
- ելակետային տվյալների պատրաստում և ծրագրի անմիջական կատարում:

Յուրաքանչյուր փուլն ունի իր առանձնահատկություններն ու իրականացման դժվարությունները:

Խնդրի դրվածքի սահմանում: Այս փուլում պետք է իրականացվեն խնդրի վերաբերյալ տեղեկությունների հավաքագրում, խնդրի պայմանի ձևակերպում, խնդրի լուծման վերջնական նպատակների սահմանում (որոշում), արդյունքների արտածման ձևի որոշում, տվյալների՝ դրանց տիպերի, հնարավոր արժեքների տիրույթների, կառուցվածքների նկարագրում (ինչ է տրված, ինչ է պահանջվում):

Խնդրի մաթեմատիկական դրվածքի ձևակերպում: Դա նշանակում է, որ պետք է կառուցվի ուսումնասիրվող խնդրի մաթեմատիկական մոդելը: Մաթեմատիկական մոդելը շատ պարզ և միաժամանակ կարևոր հասկացություն է, որը սահմանվում է որպես կամայական իրադրության մաթեմատիկական նկարագրություն: Մաթեմատիկական մոդելի կառուցվածքը կարող է լինել և՛ շատ պարզ, և՛ շատ բարդ՝ կախված ուսումնասիրվող իրադրությունից: Կազմել խնդրի մաթեմատիկական մոդել, նշանակում է խնդրի պայմանները ներկայացնել մաթեմատիկական բանաձևերով, որոշել խնդրի բոլոր տվյալների միջև առկա մաթեմատիկական կապերը [5]:

Այս փուլը բնութագրվում է խնդրի մաթեմատիկական ձևայնացումով (ֆորմալիզացիայով), որի ժամանակ արդյունքը որոշող տվյալների միջև հարաբերությունները արտահայտում ենք մաթեմատիկական բանաձևերով: Այսպես ձևավորվում է մաթեմատիկական մոդելը՝ սահմանված ճշտությամբ, ենթադրություններով և սահմանափակումներով: Մաթեմատիկական մոդելը պետք է բավարարի առնվազն երկու պահանջի՝ ռեալիզմի և իրացնելիության: Ռեալիզմի պահանջը ենթադրում է ուսումնասիրվող երևույթի ամենակարևոր հատկանիշների արտացոլում մոդելով, ինչպես նաև խիստ մանրամասներից ողջամտորեն ազատում: Իրացնելիության պայմանը հնարավոր ռեսուրսների մատչելի ծախսերով տրամադրված ժամանակահատվածի համար անհրաժեշտ հաշվարկների գործնական իրականացման հնարավորությունն է:

Հատկանշական է, որ ուսանողներից շատերը դժվարությամբ են մշակում խնդրի մաթեմատիկական մոդելը: Առանձնացնենք այդ դժվարությունը հիմնավորող մի քանի փաստարկ: Նախ, գոյություն ունեն անվերջ քանակությամբ խնդիր-

ներ, ուստի անհնար է առաջարկել մաթեմատիկական մոդելի կառուցման համապիտանի, հստակ, քայլային հրահանգներ: Յուրաքանչյուր խնդրի մոդելի մշակման համար ուսանողը պետք է հիմնվի ոչ միայն իր գիտելիքների, այլ նաև ստեղծագործ, ալգորիթմական մտածելակերպի և, ինչու չէ, նաև ինտուիցիայի վրա:

Երկրորդը, ուսումնառության ժամանակ ուսուցանողները հաճախ գործում են սխալ՝ առաջարկելով խնդիրներ, որոնց պայմանները արդեն գրառված են մաթեմատիկական տերմիններով: Օրինակ, տրված է 5 իրական թիվը, կամ տրված է 10x7 երկչափ զանգված, որի տարրերը կենտ թվեր են, կամ գտնել զանգվածի մեծագույն տարրը և այլն: Շատ ժամանակ ուսանողը նույնիսկ չի էլ կասկածում, որ այդ ամենը կրում է որոշակի գործնական, ֆիզիկական իմաստ՝ կախված խնդրի առարկայական ոլորտից:

Խնդրի լուծման մեթոդի ընտրություն: Անհրաժեշտ է նշել, որ խնդրի արդեն կառուցված մաթեմատիկական մոդելը թելադրում է խնդրի մեթոդի ընտրություն: Օրինակ, եթե խնդրի մաթեմատիկական մոդելը մենք կառուցել ենք այնպես, որ խնդրի լուծումը հանգեցվում է գծային հավասարումների համակարգի լուծմանը, ապա պետք է ընտրենք համակարգի լուծման եղանակներից մեկը: Որպես կանոն, խնդիրների մեծամասնության համար լուծման մեթոդները արդեն մշակված են, և հաճախ մի քանի տարբերակով: Պարզ խնդիրների դեպքում կարելի է օգտագործել որոշակի սխեմաներ, օրինակ, հաշվողական մաթեմատիկայից, որում կուտակված է տարբեր հաշվողական խնդիրների լուծման տարիների (և երբեմն դարերի) փորձ: Կարիք չկա կրկին մշակել արդեն հայտնի մեթոդները, պարզապես պետք է ուսումնասիրել դրանք և գործնականում կիրառել սեփական խնդիրների լուծման համար: Նման մեթոդներից են, օրինակ, ոչ գծային հավասարումների արմատների որոշման, որոշյալ ինտեգրալների հաշվման, դիֆերենցիալ հավասարումների թվային ինտեգրման, տվյալների վերադասավորման և շատ ուրիշ մեթոդներ:

Մնում է միայն ընտրել այն տարբերակը, որը լավագույնս համապատասխանում է մեր խնդրի պահանջներին: Շատ դեպքերում խնդիրը կարող է լուծվել մի քանի թվային մեթոդներով: Խնդրի լուծման կոնկրետ թվային մեթոդի ընտրությունը սովորաբար կատարվում է հետևյալ չափանիշներով.

- խնդրի լուծման արդյունավետ ժամանակի ապահովում,
- առկա ռեսուրսների (հիշողության) արդյունավետ օգտագործման ապահովում,
- հաշվարկների անհրաժեշտ ճշգրտության ապահովում,
- նվազագույն արժեքային ծախսերի ապահովում,
- ստանդարտ ենթածրագրերի օգտագործման հնարավորության ապահովում:

Խնդրի լուծման համար փորձում են ընտրել այնպիսի մեթոդ, որը ապահովում է համակարգչի առավել արդյունավետ օգտագործումը:

Այս փուլում ուսանողի մոտ բավարար մաթեմատիկական բազայի առկայությունը խնդրի լուծման մեթոդի հաջող ընտրության գրավականն է, մինչդեռ ուսա-

նողներից շատերն անտեսում են խնդիրների լուծման մեթոդների և հնարքների ուսուցումը, ուշադրությունը բևեռելով միայն ծրագրավորման լեզուների վրա: Հաշվի առնելով նաև, որ ուսանողներից շատերը չեն կարևորում այս փուլի իրական նշանակությունը ծրագրի կազմման պրոցեսում, հոդվածում ուշադրությունը կբևեռենք ալգորիթմների մշակման սկզբունքների վրա, ներկայացնելով համապատասխան օրինակներ:

Ալգորիթմի մշակում: Ալգորիթմների մշակման փուլը թերևս ամենակարևորներից է: Այս փուլում կազմվում է խնդրի լուծման մանրամասն նախագիծը (պլանը): Հաճախ ուսանողների կողմից այս փուլը դիտվում է որպես անմիջապես ծրագրավորումից առաջ կատարվող օժանդակ գործողություն: Ուսանողներից շատերը նույնիսկ փորձում են սկզբից կազմել ծրագիրը, նոր խնդրի լուծման բլոկսխեման կամ բառաբանաձևային ալգորիթմը: Հաշվի չի առնվում, որ ալգորիթմի հաջող մշակումը հնարավորություն է տալիս խուսափել շատ սխալներից, քանի որ հենց այս փուլում է որոշվում ապագա ծրագրի տրամաբանությունը: Իսկ տրամաբանական սխալները առավել դժվար է գտնել և ուղղել արդեն կազմած ծրագրի մեջ:

Եթե ալգորիթմի մշակմանը վերաբերվենք ոչ անհրաժեշտ ուշադրությամբ, ապա հետագայում, ծրագրավորման փուլում կառաջանան դժվարություններ, ալգորիթմը կպահանջի լրացուցիչ վերամշակում, նոր ջանքեր և այլն: Իսկ կարգաբերման փուլում կարող է պարզվել, որ, ցավոք, ալգորիթմը տալիս է սխալ արդյունք կամ ընդհանրապես իրացնելի չէ, և պետք է մշակել նոր ալգորիթմ:

Խնդիրների լուծման ալգորիթմների մշակումը ստեղծագործ աշխատանք է: Ցավոք, չկա համապիտանի մեթոդ, որը թույլ կտա հեշտությամբ կազմել ցանկացած ալգորիթմ, քանի որ կյանքի իրավիճակներն ու խնդիրները բազմազան և անկանխատեսելի են: Եթե նման մեթոդ գոյություն ունենար, ապա կառաջանար իրական հնարավորություն ալգորիթմիզացիայի պրոցեսը ավտոմատացնելու համար՝ հանձնարարելով այն որոշակի կատարողի, հավանաբար գերհզոր համակարգչի: Այնուամենայնիվ, կարելի է տալ ալգորիթմների մշակման մեթոդաբանությանը վերաբերվող որոշ խորհուրդներ:

Ինչքան ալգորիթմները դառնում են ավելի բարդ, այնքան ավելի դժվար է դառնում դրանց աշխատանքի տրամաբանության ընկալումը: Եվ այդ պարագայում, նման բարդ ալգորիթմներում ավելի դժվար է հայտնաբերել, ուղղել սխալները կամ կատարել որոշ փոփոխություններ: Ժամանակի կեսից ավելին ծրագրավորողը ծախսում է ծրագրերը շտկելու և ձևափոխելու վրա: Այս կապակցությամբ, ծրագրային ապահովման ինդուստրիան առաջարկում է ավելի համակարգված մոտեցում ծրագրավորմանը (և այդպիսով նաև խնդիրների ալգորիթմիզացիային), այսինքն առաջարկում է տեխնոլոգիաներ, որոնց օգտագործումը նվազեցնում է ծրագրերում սխալների հավանականությունը, նպաստում է ծրագրերի հասկանալուն և հեշտացնում է փոփոխման պրոցեսը:

Կառուցվածքային ծրագրավորումն այդ տեխնոլոգիաներից առաջինն է, դրա հիմքում ընկած է կառուցվածքայնության վերաբերյալ Բեմի և Ջեկոպինի կողմից

ապացուցված թեորեմը [7]: Ըստ այս թեորեմի, անկախ նրանից, թե որքան բարդ է խնդիրը, համապատասխան ալգորիթմի բլոկ-սխեման միշտ էլ կարող է ներկայացվել սահմանափակ թվով հիմնային ղեկավարող կառուցվածքներով՝ գծային, ճյուղավորման և ցիկլային:

Այս թեորեմի ապացուցման հիմնական գաղափարը կայանում է ալգորիթմի յուրաքանչյուր մասի ձևափոխմանը այդ երեք հիմնական կառուցվքներից մեկին կամ դրանց համակցմանը այնպես, որ ալգորիթմի չկառուցվածքայնացված մասը նվազի: Բավարար թվով նման ձևափոխություններից հետո մնացած չկառուցվածքայնացված մասը կամ կանհետանա, կամ կդառնա ավելորդ: Ապացուցվում է, որ արդյունքում կստացվի միայն նշված ղեկավարող կառուցվածքները օգտագործող և ելակետայինին համարժեք ալգորիթմ:

Կառուցվածքային ծրագրավորման նպատակն է՝ ընտրել ծրագրի կառուցվածքը ելակետային խնդրի ենթախնդիրների տրոհման միջոցով: Ծրագրերը պետք է ունենան պարզ կառուցվածք: Բարդ, խճճված ծրագրերը, որպես կանոն, հիմնականում աշխատունակ չեն և դրանց թեստավորումը պահանջում է մեծ ծախսեր:

Կառուցվածքային ծրագրավորման տեխնոլոգիայի տրամաբանական զարգացումն են հանդիսանում մոդուլային ծրագրավորման տեխնոլոգիան, օբյեկտ-կողմնորոշված ծրագրավորման տեխնոլոգիան և այլն:

Գոյություն ունեն ալգորիթմերի մշակման մի քանի մեթոդներ, որոնցից ամենատարածվածն են հանդիսանում մասնավոր (անհատական) նպատակների մեթոդը, վերընթացի մեթոդը և էվրիստիկ ալգորիթմի մեթոդը:

Մասնավոր նպատակների մեթոդով ալգորիթմը մշակելու համար անհրաժեշտ է որոշել խնդրի լուծման հնարավորությունների տարբերակները.

- հնարավոր է արդյոք լուծել գոնե խնդրի մի մասը՝ հաշվի չառնելով որոշակի պայմաններ,
- հնարավոր է արդյոք լուծել խնդիրը մասնավոր դեպքերի համար,
- արդյոք կա մի բան, որը բավարար հասկանալի չէ,
- հանդիպել ենք արդյոք նման խնդրի և հնարավոր է այն փոփոխել տրված խնդիրը լուծելու համար:

Վերընթացի մեթոդի համար նախ անհրաժեշտ է ենթադրել, որ գոյություն ունի խնդրի լուծման որոշակի նախնական վիճակ: Այնուհետև պետք է որոշել, թե որքան հնարավոր է շարժվել առաջ՝ նախնական լուծումից դեպի լավագույնը:

Էվրիստիկ ալգորիթմը մշակելիս պետք է հիշել, որ նման ալգորիթմը սովորաբար օգնում է գտնել լավ, բայց ոչ պարտադիր օպտիմալ լուծումը: Ընդհանուր մոտեցումը ենթադրում է ճշգրիտ լուծման վերաբերյալ բոլոր պահանջների թվարկումը՝ նշումով, թե այդ պահանջներից որոնք պետք է պարտադիր ապահովվեն, իսկ որոնց համար հնարավոր է փոխզիջում:

Այժմ փորձենք ներկայացնել ալգորիթմի ընտրության ազդեցությունը ծրագրի իրականացման արդյունավետության վրա մի քանի պարզ օրինակներով:

Ահա Y մեծության հաշվման պարզ օրինակ.

У=AX³ + BX²+CX+D

Լուծման ուղիղ ձևը հետևյալն է.

У=A*X^3 + B*X^2 + C*X + D

Սակայն կրկնակի բազմապատկումն ավելի արդյունավետ է, քան ամբողջ փոքր թվով աստիճան բարձրացումը, հետևաբար ալգորիթմը կարող է ունենալ հետևյալ տեսքը.

У=A*X*X*X + B*X*X + C*X + D:

Նման ալգորիթմի դեպքում պահանջվում է կատարել վեց բազմապատկման և երեք գումարման գործողություն: Թվում է, թե այլևս ոչինչ հնարավոր չէ կատարելագործել: Բայց իրականում այդ ալգորիթմը կարելի է պարզեցնել, եթե օգտվենք, օրինակ, բազմանդամի տրոհման մեթոդից (Հորների մեթոդ).

У=AX³ + BX²+CX + D =X(AX² + BX + C) + D =X(X(AX+B) + C) + D:

Այժմ ալգորիթմի հաշվումն ունի այս տեսքը.

У = X*(X*(A*X + B) + C) + D

Տվյալ դեպքում պահանջվում են երեք բազմապատկման և երեք գումարման գործողություններ: Այսպիսով մենք կրճատեցինք գործողությունների քանակը՝ մեծացնելով ճշգրտությունը: Այս արտահայտությունը կոչվում է պոլինոմի ներդրված տեսք:

Ալգորիթմի կատարելագործման այս օրինակը ցուցադրում է հետևյալ կանոնը՝ ալգորիթմի ընտրության համար ծախսված մեկ ժամը համարժեք է ծրագրավորման հինգ ժամին: Հաճախ անտեսվում է ալգորիթմի մանրամասն ընտրությունը և ծրագրավորվում է միտքն եկած հենց առաջին ալգորիթմը:

Քննարկման արժանի ևս մեկ օրինակ՝ հանդիսանում է արդյոք N թիվը պարզ:

Պարզ թիվը դա այն թիվն է, որը կարող է առանց մնացորդի բաժանվել մեկի և իր վրա, բացի 1-ից: Օրինակ՝ 3, 5, 7, 11, 13, 17-ը պարզ թվեր են, իսկ 4, 9, 21, 35-ը պարզ թվեր չեն հանդիսանում:

Խնդրի լուծման պարզագույն ալգորիթմը մուտքային N թիվը բաժանում է 2, 3, 4 …. N-1 այնքան ժամանակ, քանի դեռ N-ը չի բաժանվում որևէ մի թվի վրա առանց մնացորդի, կամ բաժանարարի կարգավիճակով այն դեռևս փորձարկված չէ: Այս ալգորիթմն, իհարկե, չի հանդիսանում տվյալ խնդրի լուծման ամենաարդյունավետը, սակայն հանդիսանում է ամենաբացահայտը:

Դժվար չէ ենթադրել, որ անիմաստ է ստուգել, թե N թիվը բաժանվում է արդյոք իրենից փոքր բոլոր զույգ թվերի վրա թե ոչ, բացի 2-ից: Ճիշտ կլինի նման ստուգումը կազմակերպել միայն կենտ թվերի համար: Դա կփոքրացնի քայլերի քանակը կիսով չափ: Կարելի է մշակել նոր ալգորիթմ, ըստ որի N թիվը բաժանվում է 2-ի և բոլոր կենտ թվերի վրա, քանի դեռ այն չի բաժանվում նրանցից որևէ մեկի վրա առանց մնացորդի կամ, քանի դեռ չենք հասել N-ին: Ծրագրավորողների մեծ մասը, ազատվելով հաշվումների գրեթե կեսից, անմիջապես կանցնեն ծրագրավորմանը: Սակայն, որպեսզի ընդգծենք ալգորիթմի մանրակրկիտ ընտրության կարևորությունը, մինչև ծրագրավորումը, անցկացնենք հետագա վերլու-

ծություն:

Դժվար չէ նկատել, որ բավական է ստուգել, թե հանդիսանում են արդյոք թվի հնարավոր բաժանարարները փոքր կամ հավասար N-ի քառակուսի արմատին: Այսպիսով մենք ստանում ենք մի նոր ալգորիթմ:

Այս ալգորիթմը բաժանում է N-ը 2-ի և կենտ թվերի վրա, որոնք փոքր կամ հավասար են N-ի քառակուսի արմատին, մինչև որ N-ը առանց մնացորդի չբաժանվի կամ դեռ N-ի քառակուսի արմատը չի փորձարկվել բաժանարարի կարգավիճակում: Հաշվելով այն թվերի քանակը, որոնք մենք պետք է ստուգենք բաժանարարի կարգավիճակում երեք ալգորիթմներից յուրաքանչյուրի համար, կտեսնենք, որ առաջին ալգորիթմը ստուգում է 62 անգամ շատ թիվ, քան երրորդը՝ N=1001 համար առաջին ալգորիթմը ստուգում է 1000 թիվ, իսկ երրորդը ստուգում է ընդամենը 16 թիվ:

Հարց է ծագում. ինչպես ընտրել կամ մշակել արդյունավետ ալգորիթմ: Առաջարկում ենք մի քանի կանոններ:

1. Մի՛ սկսեք ծրագրավորումը միտքը եկած առաջին ալգորիթմով, քննարկե՛ք գոնե մի քանի տարբերակներ: Ընտրե՛ք դրանցից լավագույնը: Եթե դուք քննարկել եք միայն մեկ ալգորիթմ ձեր խնդրի լուծման համար, դժվար թե այն լինի լավագույնը:

2. Ընտրե՛ք խնդրի ալգորիթմը մանրակրկիտորեն: Գոյություն ունեն գրականության ոչ քիչ աղբյուրներ, որտեղ նկարագրվում և ուսումնասիրվում են հաշվողական շատ ալգորիթմներ: Դ. Կնուտի եռահատորյակում, «Կոնկրետ մաթեմատիկա» և այլ գրքերում [1, 2, 3, 4] կան մեծ քանակությամբ հիմնական հաշվողական ալգորիթմներ: Բոլոր պրոֆեսիանալ ծրագրավորողները պետք է ծանոթ լինեն այդ հրատարակություններին:

Այսպիսով ճիշտ մաթեմատիկական մեթոդի ընտրությամբ ալգորիթմի մշակումը խնդրի լուծման կարևորագույն փուլ է և արդյունավետ ծրագրի մշակման անհրաժեշտ նախապայման:

Այսպիսով, ծրագրավորումն ամենապահանջված և տարածված մասնագիտություններից մեկն է աշխարհում: Սակայն նրա ուսումնառության և կիրառման ընթացքում ալգորիթմական լեզվի տիրապետման տարրական գրագիտությունը, հաճախ, ներկայացվում է որպես մասնագիտության գրեթե սպառիչ բովանդակություն: Այն լեզուն, որով ուսանողն արտահայտում է իր մտքերը, խորը ազդեցություն է գործում նրա մտածողության և ստեղծագործական ընդունակությունների վրա: Կիրառվող ծրագրավորման լեզուների կառուցվածքներն ու հատկությունները հաճախ չունեն համոզիչ ու տրամաբանական բացատրություն և արդյունքում նրանցում տիրող անկանոնություններն անմիջականորեն ազդում են ուսանողների մոտ ձևավորվող ծրագրավորման ոճի վրա: Ծրագրավորման «արդյունավետ ոճ» հասկացությունը պարզաբանելու համար հոդվածում ուսումնասիրվում են համակարգչի վրա խնդրի լուծման գործընթացի հիմնական փուլերը, ներկայացվում են ծրագրային ապահովման ինդուստրիայի կողմից առաջարկվող կառուցվածքային, մոդուլային, օբյեկտկողմնորոշված տեխնոլոգիաների կիրառման հիմ-

նական սկզբունքները, որոնց օգտագործումը նվազեցնում է ծրագրերում սխալների հավանականությունը, նպաստում է ծրագրերի հասկանալուն և հեշտացնում է փոփոխման պրոցեսը:

Աշխատանքում նկարագրվում են ալգորիթմերի մշակման ամենատարածված մեթոդները՝ մասնավոր նպատակների, վերընթացի և էվրիստիկ ալգորիթմի: Ալգորիթմի ընտրության ազդեցությունը ծրագրի իրականացման արդյունավետության վրա ներկայացվում է մի քանի պարզ օրինակներով: Առաջարկվում են արդյունավետ ալգորիթմներ ընտրելու կամ մշակելու մի քանի կանոններ:

Այս ամենի արդյունքում հանգում ենք շատ կարևոր եզրակացության. ճիշտ մաթեմատիկական մեթոդի ընտրությամբ ալգորիթմի մշակումը խնդրի լուծման կարևորագույն փուլ է և արդյունավետ ծրագրի մշակման անհրաժեշտ նախապայման:

ԳՐԱԿԱՆՈՒԹՅՈՒՆ

1. Ахо А., Хопкрофт Дж., Ульман Дж., Построение и анализ вычислительных алгоритмов. М.: Мир, 1979, 362 с.
2. Грэхем Р., Кнут Д., Паташник О., Конкретная математика. Основание информатики: Пер. с англ. М.: Мир, 1998, 703 с.
3. Кнут Д. Искусство программирования для ЭВМ: В 3 т. М.: Мир, 1978.
4. Кормен Томас Х., Лейзерсон Чарльз И., Ривест Рональд Л., Штайн Клиффорд. Алгоритмы: построение и анализ, 2-е издание.: Пер. с англ. М.: Издательский дом «Вильямс», 2005, 1296 с.
5. Мышкис А. Д., Элементы теории математических моделей. 3-е изд., испр. М.: КомКнига, 2007, 192 с. ISBN 978-5-484-00953-4.
6. Самарский А. А., Михайлов А. П., Математическое моделирование. Идеи. Методы. Примеры. 2-е изд., испр. М.: Физматлит, 2001. ISBN 5-9221-0120-X.
7. Успенский В.А., Семенов А.Л., Теория алгоритмов: основные открытия и приложения. М.: Наука, 1987, 288 с.
8. Эшенхерст Р. Н., Лекции лауреатов премии Тьюринга за первые двадцать лет 1966-1985. Пер. с англ., 1993 г., 560 с.

Разработка алгоритма как важнейший этап решения задачи

Оганян Гегине
Кюрегян Арменуи
Канарян Гаяне

Резюме

Ключевые слова: *основные этапы процесса решения задачи, определение постановки задачи, формулировка математической постановки задачи, выбор метода решения задачи, разработка алгоритма, технология программирования, метод частных целей, метод подъема, эвристический алгоритм*

Программирование – одно из самых популярных и востребованных профессий в мире. Однако, как в преподавании программирования, так и в повседневной практике доминирует поверхностный подход к изучению и применению программирования, когда элементарная грамотность, степень владения алгоритмическими языками представляется как исчерпывающее содержание профессии.

Язык, на котором студент выражает свои мысли, оказывает глубокое воздействие на его мышление и творческие способности. Структура и свойства используемых языков программирования часто не имеют убедительного и логического объяснения, и, в результате, царящий в них беспорядок непосредственно влияет на формирующийся у студентов стиль программирования.

С целью прояснения понятия «эффективный стиль» программирования в статье рассматриваются основные этапы процесса решения задачи на компьютере, представляются предлагаемые индустрией программного обеспечения основные принципы применения структурного, модульного, объектно-ориентированного технологий, использование которых уменьшает вероятность ошибок в программах, упрощает их понимание и облегчает модификацию. В статье описаны наиболее распространенные методы разработки алгоритмов: метод частных целей, метод подъема и эвристический алгоритм. Влияние выбора алгоритма на эффективность реализации программы иллюстрируется несколькими простыми примерами. Предлагается несколько рекомендаций по выбору или разработке эффективных алгоритмов. В результате мы приходим к очень важному выводу: разработка алгоритма путем выбора правильного математического метода является важнейшим этапом решения задачи и необходимым предварительным условием разработки эффективной программы.

Development of Algorithms as the Main Stage of Problem Solution

Ohanyan Heghine
Kyureghyan Armenuhi
Kanaryan Gayane

Summary

Key words: *the main stages of problem solution process, determination(definition) of problem formulation, formulation of mathematical order of the problem, choice of method of problem solution, development of algorithms, programming technology, method of private aims, ascending method, heuristic algorithm*

Programming is the most demanded and popular profession. But while studying and implementing it, the elementary literacy of mastering the algorithmic language is often introduces as its exhaustive contents. The language the students use to express their thoughts influence their thinking, creativity. The constructions, properties of applied programming languages don't often have convincing and logical explanation and thus disorders existing in them directly influence the programming style shaping within students. To explain effective programming style, the phases of problem solution on the computer are studied in the article, the main principles of constructional, modular and object-oriented technology use offered by program providing industry are introduced. Its usage minimizes the possibility of mistakes in programs, helps understand programs and facilitates changing process. Here are described the most common methods of processing algorithms, those are for private aims, ascending and heuristic algorithms. The influence of the algorithmic choice on the program implementation effectiveness is introduced with simple examples. A few rules are introduced to choose and implement effective algorithms.

Finally we have come to a very important conclusion: the implementation of the algorithm with the choice of right mathematical method is an important stage of problem solution and a necessary precondition for effective program implementation.