ՀԱՅԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅԱՆ ԳԻՏՈՒԹՅԱՆ ԵՎ
ԿՐԹՈՒԹՅԱՆ ՆԱԽԱՐԱՐՈՒԹՅՈՒՆ

Հայաստանի Պետական Ճարտարագիտական Համալսարան

Հաշվողական
Համակարգերի
Մաթեմատիկական
Ապահովման Ամբիոն

Ինֆորմատիկա
(ծրագրավորում C/C++ լեզուներով)

Լաբորատոր աշխատանքների
մեթոդական ցուցումներ

ԵՐԵՎԱՆ 2007

ՀՏԴ 681.3.06

Ինֆորմատիկա (ծրագրավորում C/C++ լեզուներով)

Լաբորատոր աշխատանքների մեթոդական ցուցումներ

Կազմողներ՝ պրոֆ. Արմենակ Պալյան,
տ.գ.թ. Արեգ Պալյան

Հայաստանի Պետական Ճարտարագիտական Համալսարան:
Երևան 2007թ: 31 էջ:

Լաբորատոր աշխատանքների նպատակն է՝ յուրացնել ծրագրավորման մեթոդները, C/C++ լեզուներով ծրագրերի կազմումը և կարգաբերումը՝ օգտագործելով Visual C++ ծրագրերի մշակման համակարգ:

Գրախոս՝ դոց. Ս.Հովհաննիսյան
Խմբագիր՝ Ն. Խաչատրյան

1.Ներածական ընդհանուր տեղեկություններ

«Ինֆորմատիկա - ծրագրավորում C/C++ լեզուներով» դասընթացի շրջանակներում ուսումնասիրվում են ժամանակակից ծրագրավորման և ծրագրերի կարգաբերման մեթոդները:

Դասընթացը հիմնվում է C/C++ ծրագրավորման լեզուների վրա: Այդ լեզուները լայնորեն կիրառվում են համակարգային և կիրառական ծրագրավորման բնագավառներում՝ բիտային դաշտերի և հասցեների հետ աշխատելու լայն հնարավորությունների և օբյեկտային կողմնորոշման շնորհիվ:

C/C++ ծրագրավորման լեզուների գործնական ուսումնասիրությունը ամենաարդյունավետ ձևով կարող է կատարվել կոնսոլային երկխոսություն կիրառող ծրագրերի օրինակներով : Այդ պատճառով որպես ծրագրերի մշակման միջավայր ընտրված է Visual C++ (Console Application ռեժիմում):

2.Լաբորատոր աշխատանքների տեխնիկական ապահովումը

1.Համակարգիչ՝ IBM/PC տիպի:
2.Օպերացիոն համակարգ՝ Windows 2000(XP):
3.Visual C++ ծրագրերի մշակման համակարգ:

Լաբորատոր աշխատանք 1

Երկխոսության կազմակերպում

1.1 Ծանոթություն VC++ միջավայրի հետ պարզագույն ծրագրի օրինակով:

1.2 printf( ) և scanf( ) ֆունկցիաների ուսումնասիրումը:

1.3 Ծրագիրը հայտարարում է փոփոխականներ և դուրս է բերում էկրանի վրա 3 տողով.
float a = 35.6877 (տպագրության դաշտը՝ 8, ճշտությունը՝ 3 նիշ),
int b = -785 (տպագրության դաշտը՝ 6 նիշ) ,
float c = 645.362 (տպագրության դաշտը՝ 6, ճշտությունը՝ 0 նիշ) :

1.4 Ծրագիրը հաշվարկում է գնված մատիտների և տետրերի ընդհանուր արժեքը: Հայտարարել հետևյալ փոփոխականները.
float kar, tetr - մատիտի և տետրի գինը,
int nk, nt - մատիտների և տետրերի քանակը:
Ծրագիրը ստեղնաշարից մուտքագրում է նշված 4 տվյալները, ընդ որում՝ գներն ունեն կոտորակային մաս: Ընդհանուր արժեքը **2** նիշի ճշտությամբ արտածվում է էկրանին: Տվյալների ներմուծումից առաջ ծրագիրը արտածում է համապատասխան հուշում:

Լաբորատոր աշխատանք 2

Պայմանական օպերատորներ

2.1 Ծանոթացում VC++ միջավայրում փոփոխականների արժեքների հսկման միջոցների հետ:

2.2 Ծրագիրը լուծում է քառակուսի հավասարումը և արտածում նրա արմատները:
int a, b, c գործակիցները ներմուծվում են ստեղնաշարից:
Եթե a = 0 կամ d < 0, ապա ծրագիրը էկրանին դուրս է բերում համապատասխան հաղորդագրություն ;

2.3 Ծրագիրը ներմուծում է 3 թիվ՝ float a, b, c (եռանկյան կողմերը), ծրագիրը որոշում է, թե կարելի՞ է արդյոք այդ 3 կողմերով կազմել եռանկյունի և էկրանին դուրս է բերում համապատասխան հաղորդագրություն (Yes/No) :

2.4 Ծրագիրը ներմուծում է ամսի անվանման համարը և ըստ դրա որոշում է տարվա համապատասխան եղանակի անվանումը - ամառ, աշուն և այլն:
Օգտագործել (if … else if …) օպերատորը :

2.5 Ծրագիրը հաշվարկում է Ֆունկցիա.

$$y = \begin{cases} (1+a^2)^4 & -5 \le x \le 5 \\ \cos \ln^2 x + x^2 & x > 5 \\ a & x < -5 \end{cases}$$

a և x փոփոխականները double տիպի են և ներմուծվում են ստեղնաշարից:

Ստուգել հետևյալ արժեքների համար՝

x=10.0 a=2.0

x=3.0 a=2.0

Լաբորատոր աշխատանք 3

Ցիկլային ծրագրեր

3.1 Ծրագիրը հաշվարկում է PI թիվը հետևյալ շարքի միջոցով: Շարքի անդամների թիվը ներմուծվում է.

$$\mathbf{PI} = 4 * \sum_{\mathbf{n}=1}^{\infty} (-1)^{\mathbf{n}-1} \frac{1}{2\mathbf{n}-1} = 4 * (1 - 1/3 + 1/5 - .....)$$

Չօգտագործել pow() ֆունկցիա: Շարքի հերթական անդամի նշանը որոշելու համար օգտագործել պայմանական գործողություն (?): Հաշվարկները կատարել միտողանի օպերատորի միջոցով, որը *for* ցիկլի մարմինն է: Ծրագրի արդյունքները ստանալ 1000, 1000000 թվով անդամների դեպքում:

3.2 Ծրագիրը հաշվարկում է

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

ֆունկցիայի արժեքները:
s և x փոփոխականները float տիպի են, n -ը՝ int: n և x -ի արժեքները ներմուծվում են ստեղնաշարից:
Ծանոթություն Ֆակտորիալի հաշվարկն իրականացնել մարմին չունեցող ներքին ցիկլով: Հաշվի առնել, որ 15!-ն ամենամեծ արժեքն է, որը կարող է տեղավորվել int տիպի

փոփոխականում, այդ պատճառով էլ $n \le 7$: Ստուգել ծրագրի աշխատանքը n=10 դեպքում և բացատրել այն:

3.3 Ծրագիրը հաշվարկում է ֆունկցիայի արժեքները:

$$\mathbf{y} = \begin{cases} 2^{5-\mathbf{x}} & \mathbf{x} > 1 \\ 7 - \mathbf{x} & \mathbf{x} \le 1 \end{cases}$$

որտեղ $\mathbf{x} \in [-5.5, 4.5]$, փոխվելով $\Delta \mathbf{x} = 2$ քայլով: Արժեքները տպվում են հետևյալ աղյուսակի տեսքով (ստորակետից հետո 1 նիշի ճշտությամբ).

__Func		___Arg
ֆունկ. արժ.		առգ. արժ.
-"-		-"-
-"-		-"-

6 6 դիրքերի քանակը

3.4. Ծրագիրը հաշվում և տպում է օգտագործողի կողմից ներմուծված թվից փոքր բոլոր պարզ թվերը: Մեկ տողի վրա տպել 10 թիվ՝ յուրաքանչյուրին հատկացնելով 5 նիշ: Ծանոթություն. p թվի պարզ լինելը որոշվում է p/2-ից փոքր կամ հավասար թվերի վրա իր հաջորդաբար բաժանելով: Օգտագործել թվի տիպի նշանը (flag) կրկնակի ցիկլում:

3.5 Ծրագիրը հաշվարկում է ֆունկցիայի արժեքը.

$$y = \sum_{k=1}^{n} (k+5) \sum_{i=1}^{k} (i+k)$$

y, i, k, n - int տիպի են: n -ի արժեքը ներմուծում է օգտագործողը:

Լաբորատոր աշխատանք 4

Զանգվածներ, continue օպերատորը

4.1. Ծրագիրը հաշվարկում և տպում է սկզբնավորված unsigned arr [10] զանգվածի կենտ տարրերի միջին թվաբանականը: Օգտագործել continue օպերատորը՝ զույգ տարրերը բաց թողնելու համար:

4.2 Օգտագործելով unsigned arr[10] սկզբնավորված զանգվածը, ծրագիրը հաջորդաբար արտագրում է զանգվածի այն տարրերը, որոնք բազմապատիկ են p -ին, unsigned arr1[10] զանգվածի մեջ: p - ն ներմուծվում է: Տպել arr1[10] զանգվածի բոլոր արտագրված տարրերը:

4.3 Ծրագիրը ներմուծում է եռանիշ ամբողջ թիվ, որոշում է նրա բոլոր թվանշանները և տեղադրում int dig[3] զանգվածում: Ծրագիրը տպում է թվի ամենամեծ թվանշանը: Եթե ներնուծվել է ոչ եռանիշ թիվ, ծրագիրը սխալի հաղորդագրություն է արտածում:
Ծանոթություն. թվի թվանշանները կարելի է գտնել աջից 10-ի բաժանելով:

4.4. Ծրագիրը ներմուծում է դրական ամբողջ տասական թիվ և նոր հիմք՝ base $\leq$ 16: Ծրագիրը թիվը ներկայացնում է նոր

հիմքով և տպում այն: Ծրագրում ստեղծել char digit[16] զանգված, որտեղ 0 — 15 ինդեքսներին համապատասխանում են տասնվեցական 0 — F սիմվոլները: Սիմվոլը արտածելու համար կիրառել putchar( ) կամ printf( ) ֆունկցիաները:

Լաբորատոր աշխատանք 5

Տողեր, երկչափ զանգվածներ, switch օպերտորը

5.1. Ծրագիրը ներմուծում է կամայական երկարությամբ տող, տեղում շրջում է այն և տպում ստացված տողը: Օգտագործել երկու (i,j) փոփոխականներով for ցիկլը:

5.2. Ծրագիրը ներմուծում է կամայական երկարությամբ տող, տեղում շրջում յուրաքանչյուր զույգ սիմվոլները՝ սկսած առաջինից: Ցիկլի ավարտի պայմանն է՝ փոխանակվող զույգ բայտերի մեջ թեկուզ մեկ ‘\0’ սիմվոլի առկայությունը:

5.3. Ծրագիրը ներմուծում է երկչափ char names[M][N] զանգվածի մեջ M անուններ (M և N -ը նախապրոցեսորի հաստատուններն են):
Ծրագիրն անունները տեսակավորում է այբբենական աճման կարգով՝ օգտագործելով “Տեսակավորման պղպջակավոր-1 ալգորիթմը” ([1], էջ 270): Ծրագիրն արտածում է անունների տեսակավորված զանգվածը: Օգտագործել gets( ), puts( ), strcmp( ), strcpy( ) ստանդարտ ֆունկցիաները: Ուսումնասիրել երկչափ սիմվոլային զանգվածի կազմակերպումը՝ որպես տողերի զանգված:

5.4 Ծրագիրը ներմուծում է ապրանքի կոդը և ապրանքի քանակը: Ըստ ապրանքի կոդի switch օպերատորով

որոշում է ապրանքի գինը: Կոդ-գին համապատասխանությունը հետևյալն է.

կոդ	գին
5	22.45
10	15.17
15, 18	45.36
20, 25	61.25

Ծրագիրն արտածում է գնման արժեքը: Սխալ կոդ ներմուծելու դեպքում արտածվում է հաղորդագրություն սխալի մասին:

Լաբորատոր աշխատանք 6

Ֆունկցիաներ և ցուցիչներ

6.1. Ծրագիրը 5.3 ծրագրի տարբերակն է՝ ստանդարտ ֆունկցիաների փոխարեն օգտագործվում են ինքնուրույն մշակված cmp1( ) և copy1( ) ֆունկցիաները: cmp1( ) և copy1( ) ֆունկցիաները ստանդարտ strcmp( ) և strcpy( ) ֆունկցիաների նմանակն են: Ֆունկցիաներն օգտագործում են ցուցիչներ և ցուցիչների թվաբանություն:

6.2. Ծրագիրը ներառում և ստուգում է sort( ) ֆունկցիան, որն իրականացնում է ամբողջ թվերի սկզբնավորված զանգվածի տեսակավորումը՝ օգտագործելով “Տեսակավորման պղպջակավոր-2 ալգորիթմը” ([2], էջ 425): Օգտագործել ներքին ցիկլում վերադասավորման լրիվ բացակայության նշան (flag) ծրագիրը վաղաժամ ավարտելու համար (չօգտագործել break օպերատորը):

6.3. Ծրագիրը ներառում և ստուգում է strint( ) ֆունկցիան, որն իրականացնում է թվային տողի ձևափոխումը int տիպի թվի (ստանդարտ atoi( ) ֆունկցիայի նմանակը):

Լաբորատոր աշխատանք 7

Կառուցվածքներ

7.1 Ծրագիրը ներմուծում է ընթացիկ ամսաթիվը Date (ամիս, օր, տարի) կառուցվածքային փոփոխականի մեջ: Ծրագիրը հաշվում է մյուս օրվա ամսաթիվը, տեղադրում մեկ այլ Date տիպի կառուցվածքային փոփոխականում և դուրս է բերում էկրանի վրա - “ամիս-օր-տարի” տեսքով : Ամսվա վերջին օրը որոշելու համար օգտագործել հաստատունների զանգված, որը պարունակում է յուրաքանչյուր ամսվա օրերի թիվը: Փետրվարի օրերի թիվը ծրագիրը որոշում է ըստ ներմուծված տարվա և գրում զանգվածում:

7.2 Ծրագիրը ներմուծում է ժամը (ժամ, րոպե, վայրկյան) Time կառուցվածքային փոփոխականի մեջ: Կանչվում է Time տիպի կառուցվածքային փոփոխականն ընդունող և վերադարձնող ֆունկցիա: Ֆունկցիան ստացված ժամանակը մեծացնում է 1 վայրկյանով և վերադարձնում նոր ժամը : Օրվա վերջը հասնելիս,ժամը հավասարվում է 00-00-00-ի: Ծրագիրը դուրս է բերում ստացված ժամանակը “ժամ – րոպե – վայրկյան” տեսքով, էկրանի վրա դուրս բերումը կատարվում է “ժժ : րր : վվ” ֆորմատով, ավելացնելով պակասող առաջատար զրոները:

7.3 Ծրագիրը ստեղծում է Book տիպի կառուցվածքների զանգված (4 տարր): Book կառուցվածքը ներառում է 2 անդամ - գրքի անվանումը և գինը: Ծրագիրը լրացնում է զանգվածը օգտագործողի կողմից ներմուծվող տվյալներով: Ծրագիրը դուրս

է բերում “B” տառով սկսվող գրքերի անունները և դրանց գումարային արժեքը:

Լաբորատոր աշխատանք 8

Կարգաբերող ծրագիր, ծրագրի Release տարբերակը

8.1 Debug կարգաբերողի ընդհանուր հնարավորությունների ուսումնասիրությունը

Կատարել գործողությունների հետևյալ հաջորդականությունը.

1. Թողարկել 6.2 ծրագիրը Debug ռեժիմում:
2. Ցույց տալ arr[ ] զանգվածի պարունակությունը Variables, Watch պատուհաններում:
3. Քայլ առ քայլ ռեժիմում հասնել sort( ) ֆունցիային: Բացել ծրագրի պատուհանը:
4. Մտնել sort( ) ֆունկցիա, կատարել մի քանի քայլ: Ցույց տալ sort( ) ֆունկցիայի լոկալ փոփոխականների արժեքները: Ցույց տալ main( ) ֆունկցիայի arr [ ] զանգվածի արժեքները՝ ակտիվացնելով այդ ֆունկցիան ստեկում (Context պատուհան): Ցույց տալ arr[ ] զանգվածի արժեքները Quick Watch օպցիայով: Դուրս գալ sort( ) ֆունկցիայից:
5. Ցույց տալ arr[ ] զանգվածի պարունակությունը sort( ) ֆունկցիայից դուրս գալուց հետո:
6. Կուրսորը տեղադրել վերջին printf( ) ֆունկցիայից առաջ, կատարել Run to cursor գործողությունը: Բացել ծրագրի պատուհանը:
7. Հանել և վերականգնել Debug ռեժիմը:
8. sort( ) ֆունկցիայից առաջ տեղադրել կանգառի կետ (Breakpoint): Հասնել այդ կետին: Հանել կանգառի կետը: Հանել և վերականգնել Debug ռեժիմը:

9. Առաջին *for* ցիկլում տեղադրել կանգառի պայմանական կետ (Breakpoint) printf( ) ֆունկցիայից առաջ: Տեղադրել ցիկլի չորրորդ անցումը որպես կանգառի պայման: Թողարկել ծրագիրը:

8.2 Ծրագրում scanf( ) ֆունկցիայի և Debug-ի միջոցներով տվյալների մուտքի կազմակերպման ուսումնասիրությունը:

Կատարել հետևյալ գործողությունները.

1. Հեռացնել մեկնաբանությունը scanf( ) ֆունկցիայի առջևից: Կատարել Rebuild:
2. Հասնել scanf( ) ֆունկցիային և arr [0]-ի մեջ ներմուծել 33 արժեքը: Ձախ պատուհանում ցույց տալ, որ արժեքը ներմուծված է:
3. Ձախ պատուհանում arr [1] -ի արժեքը փոխել (-4) Debug - ի միջոցներով: Ծրագիրը կատարել մինչև վերջին printf( ) օպերատորը՝ օգտագործելով Run to cursor գործողությունը: Ծրագրի պատուհանում ցույց տալ arr [0] և arr [1] -ի փոփոխված արժեքները:
4. Վերականգնել մեկնաբանությունը scanf( ) օպերատորի առջև և կատարել Rebuild:

8.3 Ծրագրի Release տարբերակի ստեղծումը

Կատարել հետևյալ գործողությունները.

1. Ստեղծել ծրագրի երկու exe - ֆայլ Debug և Release ռեժիմներում:
2. Համեմատել 2 exe - ֆայլերի ծավալները :
3. Դուրս գալ VC++ միջավայրից, թողարկել exe- ֆայլի Release տարբերակը: Բացատրել ծրագրում getch( ) ֆունցիայի անհրաժեշտությունը:

Լաբորատոր աշխատանք 9

Ֆունկցիաների ցուցիչներ, բիտային գործողություններ

9.1 Ծրագրի երկու տարբերակները, որոնք օգտագործում են qsort() ստանդարտ ֆունկցիան, կատարում են.

- ամբողջ թվերի տեսակավորում (1 տարբերակ),
- տողերի տեսակավորում (2 տարբերակ):

Ծրագրի երկու տարբերակները բաղկացած են.

- Տեսակավորվող օբյեկտների գլոբալ զանգվածից (1 տարբերակ՝ int տիպի զանգված, 2 տարբերակ՝ տողերի զանգված char տիպի երկչափ զանգվածի տեսքով): Երկու զանգվածներն էլ սկզբնավորված են:
- main() ֆունկցիայից, որը կանչում է qsort() ֆունկցիային օբյեկտների տեսակավորման համար և արտածում օբյեկտները տեսակավորումից հետո:
- Օբյեկտների համեմատման ֆունկցիա: Ֆունկցիան ունի երկու const void* տիպի ֆորմալ պարամետրեր, որոնց միջոցով ֆունկցիան ստանում է համեմատող օբյեկտների հասցեները: Այդ ֆունկցիայի հասցեն փոխանցվում է qsort() ֆունկցիային:

Բացատրել C և C++-ում void-ցուցիչների վերագրման առանձնահատկությունները:

9.2 Ծրագիրը ներմուծում է unsigned տիպի ամբողջ դրական թիվ, դուրս է բերում էկրանին նրա տասական արժեքը և համապատասխան երկուական բիտային կոդը՝ հետևյալ տեսքով.

<u>թիվ</u>		<u>բիտային կոդ</u>
65042	=	11111110 00010010 xxxxxxxx xxxxxxx:

Բիտային կոդը ներկայացվում է որպես միմյամցից բացակով անջատված բիտերի 4 խումբ, յուրաքանչյուրում՝ 8 բիտ: Օգտագործել տրամաբանական բիտային և տեղաշարժման գործողություններ: Առանձին նիշերի արտածումն իրականացնել մեկ putchar() ֆունկցիայով՝ նիշերը ներկայացնելով որպես պայմանական ? արտահայտություն:

Լաբորատոր աշխատանք 10

Տեսանելիություն, հիշողության դասեր և բազմաֆայլ նախագծեր

10.1. Ծրագիրը ստուգում է փոփոխականների գործողության տիրույթը և հիշողության դասերը (Ծրագրի [3], էջ 222, 3.12 նկար, fig 03_12 cpp նմանակը). Ծրագրում ավելացնել main() ֆունցիայի ներքին բլոկից x գլոբալ փոփոխականին դիմումը և նրա արտածումն էկրանին: Ամբողջ ծրագիրն անցնել Debug ռեժիմում՝ հետևելով փոփոխականներին: Բացատրել ծրագրի աշխատանքը և լոկալ ստատիկ փոփոխականների ստեղծումը:

10.2. Ծրագիր 7.2 բաժանվում է 3 ֆայլի: Առաջին ֆայլը (.h) ներառում է Time կառուցվածքի հայտարարությունը: Երկրորդ ֆայլը ներառում է main() ֆունկցիան: Իսկ երրորդը՝ ընթացիկ ժամանակը 1 վայրկյանով ավելացնելու ֆունկցիան: Կազմել նախագիծ և թողարկել ծրագիրը: Ֆայլերում ներառել անհրաժեշտ #include “my.h” հրահանգը: Ծրագրի բաժանումն իրականացնել Visual C++ խմբագրի միջոցներով:

10.3. Ծրագիրը (main () ֆունցիան) երեք անգամ կանչում է func() ֆունկցիային՝ նրան փոխանցելով կանչի համարը՝ i (i = 1, 2, 3):
func() ֆունկցիան դինամիկ հիշողության տիրույթ (ԴՀՏ) է հատկացնում (256 բայթ ծավալով) և այնտեղ արտագրում է “Text i” տողը: Երեք “Text i” տողերը հայտարարել՝ օգտագործելով ցուցիչների զանգված: func() ֆունկցիան վերադարձնում է ցուցիչ ԴՀՏ-ին: main() ֆունկցիան դուրս է բերում էկրանին ստացված հասցեն և նրանով հասցեավորվող տողը, որից հետո ազատում է ԴՀՏ-ն free() ֆունկցիայով: Թողարկել ծրագիրը:
Ծրագրից հեռացնել free() ֆունկցիան: Թողարկել ծրագիրը: Բացատրել ծրագրի երկու անցումների արդյունքները:

Լաբորատոր աշխատանք 11

Սիմվոլային մուտք-ելք: Կոմպիլիատորի օպցիաներ Բացառիկ իրադարձությունների մշակումը

11.1 Ծրագիրն իրականացնում է սիմվոլների ներմուծումը էկրանից բուֆերացումով (getchar() ֆունկցիան) և արտածում է էկրանին (putchar() ֆունկցիան): Մուտք-ելքի ցիկլն ավարտվում է EOF հայտանիշով (ստեղների Ctrl+Z համախումբ): Ներմուծվող նիշը վերագրվում է char տիպի փոփոխականին (լռելյայն դիտվում է որպես signed): Ստուգել ծրագրի աշխատանքը: Նախագիծը վերակառուցել, նախապես տեղադրելով կոմպիլյատորի օպցիան՝ char տիպը մեկնաբանելով որպես unsigned char: Թողարկել ծրագիրը և բացատրել, թե ինչու՞ է ծրագրում առաջանում անվերջ ցիկլ:

11.2 Ծրագիրը ներմուծում է նշանաբառը (սիմվոլների կամայական հաջորդականություն) առանց բուֆերացման՝ օգտագործելով getch() ֆունկցիա: Յուրաքանչյուր ներմուծված սիմվոլից հետո (‘*’) նիշը արտածվում է էկրանի վրա: Ներմուծման ավարտից հետո նշանաբառը արտածվում է:

11.3 Ծրագրի try բլոկում ստեղծվում են հետևյալ իրադարձությունները.

- ամբողջ թվերի գերլցում,
- (float) թվի բաժանումը 0-ի,
- (float) թվի բաժանումը INF թվի,
- (int) թվի բաժանումը 0-ի:

Բացառիկ իրադարձությունների գրավումն իրականացվում է catch բլոկում: Ծրագրի աշխատանքը իրականացնել սովորական և Debug ռեժիմներում: Բացատրել փոփոխականների արժեքները:
Կատարել ծրագրի նոր տարբերակը՝ հանելով try - catch օպերատորները: Բացատրել արդյունքը:

Լաբորատոր աշխատանք 12

Նախապրոցեսոր, *inline* ֆունկցիա

12.1 Ծրագիրն ընդգրկում է ֆունկցիա, որը գտնում է երկու թվի առավելագույն արժեքը: Ֆունկցիան իրականացվում է 3 տարբերակով.
1. Մակրոֆունկցիա 2 պարամետրերով:
2. Հասարակ ֆունկցիա, որն ընդունում է 2 արգումենտ *int* տիպի և վերադարձնում է մաքսիմալ արժեք *int* տիպի:
3. Նույն ֆունկցիան *inline* տարբերակով:

Բոլոր տարբերակներին փոխանցել 2 արգումենտ *int* տիպի, ապա 2 արգումենտ *float* տիպի և բացատրել արդյունքը: Համեմատել 3 լուծումները:

12.2 Ծրագիրն իրականացնում է սիմվոլների ներմուծումը էկրանից բուֆերացումով (getchar() ֆունկցիան): Մուտք-ելքի ցիկլն ավարտվում է EOF հայտանիշով (ստեղների Ctrl+Z համախումբ): Մուտքի ընթացքում ծրագիրը հաշվում է դատարկ սիմվոլների քանակը, որն արտածվում է էկրանի վրա:

Ծրագիրն ընդգրկում է ֆունկցիա, որն ընդունելով սիմվոլ, որոշում է՝ դատարկ է այն, թե՝ ոչ: Դասարկ են համարվում սիմվոլներ \n, \t և բացակի սիմվոլը: Ֆունկցիան վերադարձնում է բուլյեան արժեք - *ճիշտ* կամ *սխալ*:

Ծրագիրն իրականացվում է 3 տարբերակով.

1. Անուններ BOOL, TRUE, FALSE հայտարարվում են մակրոսի տեսքով:
2. BOOL անունը հայտարարվում է որպես վերանվանված *int* տիպ, իսկ TRUE, FALSE - մակրոսի տեսքով:
3. Օգտագործվում են *bool, true, false* C++ լեզվի առանցքային բառերը:

12.3 Ծրագիրն ընդգրկում է մակրոս TEST (p), որն ստուգում է *p* պայմանը: Մակրոսի համապատասխան արգումենտ կարող է լինել կամայական արտահայտությունը, որն ունի բուլյեան արժեք: Մակրոսը մշակվում է 2 տարբերակով: Տարբերակի ընտրությունը կատարվում է ըստ ծրագրի աշխատանքի ռեժիմի՝ կարգաբերման կամ նորմալ: Նորմալ ռեժիմում մակրոսը դատարկ է, իսկ կարգաբերման ռեժիմում մակրոսը կատարում է հետևյալ գործողություններ:

Եթե պայմանը *ճիշտ է*, ապա մակրոընդլայնումը ոչինչ չի կատարում: Եթե պայմանը *սխալ է* , ապա մակրոընդլայնումը արտածում է հետևյալ հաղորդագրությունը.

Error! False Condition: < condition>
In file <source file name>
On line <line number>

որտեղ < condition> - պայմանն է, այսինքն՝ մակրոսի արգումենտը, օրինակ՝ i == 5:

Հաղորդագրության առաջին տողը ձևավորել՝ օգտագործելով նախապրոցեսորի # գործողությունը և տողերի կոնկատենացիան:

Աշխատանքի ռեժիմը (կարգաբերման կամ նորմալ) որոշվում է DEBUG նախապրոցեսորի ֆլագով: Ֆայլի անունը և տողի համարը որոշվում են ստանդարտ մակրոսների միջոցով:

main() ֆունկցիան ընդգրկում է հետևյալ գործողությունները.

- արտածում է՝ First test,
- կանչում է մակրոսը *ճիշտ* պայմանով,
- արտածում է՝ Second test,
- կանչում է մակրոսը *սխալ* պայմանով:

Գործարկել ծրագիրը կարգաբերման և նորմալ ռեժիմներում: Ցույց տալ ծրագրի սկզբնական կոդը մակրոընդլայնումներով՝ օգտագործելով կոմպիլյատորի օպցիա /E:

Լաբորատոր աշխատանք 13

Դասեր և օբյեկտներ

13.1 Ծրագիրը պարունակում է int թվերի ցիկլիկ հերթ իրականացնող դաս: Դասի մեջ ներառել կոնստրուկտոր, հերթի մեջ նոր տարր ներառելու և նրանից հեռացնելու

ֆունկցիաները: main() ֆունկցիայի մեջ ստեղծել դասի երկու օբյեկտ և ստուգել դրանց անկախ աշխատանքը:

13.2 Ծրագիրն ներառում է DateInfo դասը: Դասը ներառում է.

1. private բաժինը.

   int Month, Day, Year;

2. public բաժինը.
   - դասի տվյալները սկզբնավորող կոնստրուկտոր.
   - GetDate() ֆունկցիան, որը Month, Day, Year տվյալները տեղադրում է իրեն փոխանցվող երեք հասցեներով.
   - GetStringDate() ֆունկցիան, որն ստանում է char տիպի զանգվածի հասցեն և ըստ նրա փոխանցում ընթացիկ ամսաթիվը “mm-dd-yy” ֆորմատով, (որտեղ yy-ն տարվա վերջին երկու թվանշանն է): Ամսաթվի փոխանցումն իրականացվում է sprintf() ֆունկցիայի կանչով.
   - GetMonth() ֆունկցիան, որն ստանում է char տիպի զանգվածի հասցեն և ըստ նրա փոխանցում ընթացիկ ամսվա անունը՝ որպես տող: Օրինակ, եթե Month = 4, ապա այն փոխանցում է “April”:

main() ֆունկցիան ներառում է.

- int m, d, y փոփոխականներ GetDate()-ով ամսաթիվն ստանալու համար.
- char buf [80] զանգվածը՝ Date օբյեկտից տողեր ստանալու համար.
- DateInfo դասի Date օբյեկտի հայտարարությունը և նրա սկզբնավորումը կամայական ամսաթվով.
- GetStringDate(), GetMonth() և GetDate() ֆուկնցիաների հաջորդական կանչերը՝ այդ ֆունկցիաներով ստացվող տվյալներն արտածելով էկրանին:

Լաբորատոր աշխատանք 14
Ֆունկցիաներին օբյեկների փոխանցում: Դինամիկ հիշողության օգտագործումը

14.1 Ծրագիրը պարունակում է List դասը, որն իրականացնում է շաղկապված ցուցակ: Դասն ընդգրկում է ցուցիչներ ցուցակի առաջին և վերջին տարրերի վրա: Ցուցակի տարրերը գտնվում են դինամիկ հիշողությունում: Մեկ տարրն ընդգրկում է int տիպի արժեք և հաջորդ տարրի հասցե:

Դասն ընդգրկում է հետևյալ ֆունկցիա - անդամներ.

1. List() – կոնստրուկտոր, որը վերագրում է 0 դասի երկու ցուցիչներին (ցուցակը դատարկ է):
2. void append(int val) – մտցնում է ցուցակի վերջից նոր տարր, որն ունի val արժեք:
3. bool del() – հեռացնում է ցուցակից վերջին տարրը՝ վերադարձնելով false, եթե ցուցակը դատարկ էր:
4. bool ispresent(int val) – որոնում է val արժեքը ցուցակի տարրերում և վերադարձնում է համապատասխանաբար true/false (առկա է այս արժեքը, թե՝ ոչ):
5. void display() – արտածում է էկրանի վրա ցուցակում գտնվող արժեքները: Արտածումը կատարվում է մեկ տողով բացարկ բաժանիչով:
6. ~List() - դեստրուկտոր, որն ազատում է դինամիկ հիշողությունը, զբաղեցրած ցուցակի տարրերով: Դասի երկու ցուցիչներին վերագրում է 0 (ցուցակը դատարկ է):

Դասի աշխատանքը ստուգում է դրայվերը (main() ֆունկցիա), որը տրամադրում է դասախոսը: Դրայվերում գտնվում են կանչեր Disp() friend – ֆունկցիայի, որն արտածում է էկրանի վրա ցուցակում գտնվող արժեքները՝ կրկնելով display() ֆունկցիայի ալգորիթմը: Disp() friend – ֆունկցիան պետք է մշակվի համապատասխան իր կանչի տիպի դրայվերում (by value).

Ստացված ծրագիրը, որն ընգրկում է main(), Disp(), List դասը, պետք է կառուցել և թողարկել: Ծրագրի աշխատանքի ժամանակ առաջանում է տրամաբանական սխալ: Բացատրել սխալի պատճառը: Փոխել main() – Disp() համագործակցությունը սխալը վերացնելու համար:

14.2 Ծրագրում օգտագործվում է DateInfo դասը 13.2 ծրագրից: main() ֆունկցիան ընդգրկում է հետևյալ փոփոխականներ.

- փոփոխականներ 13.2 ծրագրից
- DateInfo դասի ցուցիչ, որն օգտագործվում է DateInfo դասի օբյեկտի ստեղծման համար դինամիկ հիշողությունում:

Օբյեկտը ստեղծման պահին սկզբնավորվում է պատահական ամսաթվով:
main() ֆունկցիան ընդգրկում է հետևյալ գործողության օպերատորներ.

- դասի ֆունկցիա - անդամների կանչեր, արտածելով ստացված տվյալները 13.2 ծրագրի նման
- նոր func1() ֆունկցիա կանչ՝ հաղորդելով իրեն ստեղծված օբյեկտի հասցեն որպես միակ արգումենտ
- GetMonth() – ի նոր կանչ ամսվա փոփոխությունը, կատարված func1() –ով, ստուգելու համար
- օբյեկտի հեռացումը դինամիկ հիշողությունից:

func1() ֆունկցիան ընդգրկում է.

- char buf[80] զանգված օբյեկտից տողեր ստանալու համար
- դասի ֆունկցիա - անդամների կանչեր ստացված օբյեկտի հասցեի միջոցով:

Կանչվում են ֆունկցիաներ.

- GetMonth() ընթացիկ ամսվա անունը տողի տեսքով ստանալու և էկրանի վրա արտածելու համար

- SetDate() օբյեկտում ամսաթիվը փոխելու համար: Ամիսը պետք է փոխվի:
- GetMonth() – ի նոր կանչ ամսվա փոփոխությունը ստուգելու համար:

Կատարվում է վերադարձ func1() ֆունկցիայից:

Լաբորատոր աշխատանք 15

Կոնստրուկտորների գերբեռնում, հղումներ

15.1 Ծրագրում օգտագործվում է List դասը 14.1 ծրագրից: Ընդգրկել դասի մեջ copy – կոնստրուկտոր, որը կատարում է շաղկապված ցուցակի խորը պատճենահանում: Ստուգել նոր List դասի աշխատանքը ծրագիր 14.1 – ի main(), Disp() ֆունկցիաների հետ, օգտագործելով օբյեկտի փոխանցումը Disp() ֆունկցիային by value ռեժիմով: Բացատրել արդյունքը:

15.2 Ծրագիրը կրկնում է 13.2 ծրագիրը հետևյալ լրացումներով.

1. DateInfo դասն ընդգրկում է 2 նոր կոնստրուկտոր.
   - լռելիության կոնստրուկտոր, որը վերագրում է օբյեկտի ամսաթվին հաստատուն արժեք – 2004 թվի նոյեմբերի 11-ը
   - copy – կոնստրուկտոր, որը վերագրում է ստեղծվող օբյեկտին սկզբնական օբյեկտը՝ օգտագործելով *this* ցուցիչը:
2. Մտցնել նոր տիպ string80 – char[80]:
3. Բոլոր ֆունկցիաներում հասցե-պարամետրեր փոխարինվում են հղումներով:
4. main() ֆունկցիայում հայտարարվում են DateInfo դասի հետևյալ օբյեկտներ.
   - Date1 օբյեկտ, որը համանման է 13.2 ծրագրի Date օբյեկտին,

- Date2 օբյեկտ առանց սկզբնավորողների՝ լռելիության կոնստրուկտոր կանչելու համար,
- Date3 օբյեկտ, որը սկզբնավորվում է Date1 օբյեկտով copy – կոնստրուկտոր կանչելու համար:

Ստեղծված օբյեկտներում կանչվում են GetDate(), GetStringDate() ֆունկցիաները: Ստացված ամսաթիվը արտածվում է էկրանի վրա օբյեկտի պարունակությունը ստուգելու համար:

Հավելված

Ստանդարտ ֆունկցիաների համառոտ
նկարագրություն

1. Մուտքի-ելքի ֆունկցիաներ

1.1 int printf (“Control string”, el1, el2,….)

որտեղ՝

Control string – ղեկավարող տող, որը կարող է ընդգրկել սպեցիֆիկատորներ *el* էլեմենտների արտածան համար:

el1, el2….. – արտահայտություններ (փոփոխականներ, հաստատուններ), որոնք պետք է արտածվեն:

Հաճախ կիրառվող *տպման սպեցիֆիկատորներ*.

%d %i - ամբողջ տիպ, տասական արժեք
%x - ամբողջ տիպ, տասնվեցական արժեք
%f - float տիպ
%lf - double տիպ
%s - տող
%c - սիմվոլ
%p - ցուցիչ

Տպման մոդիֆիկատորներ.

% m1m2m3 *տառ*

m1 – արժեք (–) նշանակում է, որ տվյալները տպվում են դաշտում ձախից, բացակայության դեպքում - աջից:

m2 – տպման դաշտը: Որոշում է դիրքերի քանակը էկրանի վրա, որոնք հատկացվում են տվյալները տպելու համար:

m3 – ճշտություն: Տրվում է *.թիվ* ֆորմատով (օրինակ, .2) float տիպի արժեքների համար:

տառ - տպման սպեցիֆիկատոր:

Օրինակ՝ %-10.2f

1.2 int scanf (“Control string”, el1, el2,….)

որտեղ՝

Control string – ղեկավարող տող, որն ընդգրկում է մուտքի սպեցիֆիկատորներ *el* էլեմենտների ներմուծման համար:

el1, el2….. – փոփոխականների հասցեներ, որոնց մեջ ներմուծվում են արժեքները:

Օրինակ՝

scanf(“%d %f”, &ia, %fa);

որտեղ՝
ia – int տիպի փոփոխական
fa- float տիպի փոփոխական

Ներմուծվող արժեքները բաժանվում են բացակով կամ *Enter- ով:*

1.3 int getchar ()

Սիմվոլը մուտքագրվում է ստեղնաշարից որպես *unsigned char* և վերադարձվում է որպես *int*: Մուտքագրումից հետո սեղմել *Enter:*

1.4 int getch ()

Սիմվոլը մուտքագրվում է ստեղնաշարից որպես *unsigned char* և վերադարձվում է որպես *int*: Մուտքագրումը կատարվում է՝ սեղմելով ստեղնը առանց *Enter-ի:*

1.5 int putchar (int p)

Արտածվում է արգումենտի կրտսեր բայտը՝ որպես սիմվոլի կոդ:

1.6 char * gets (char *str)

Տողը մուտքագրվում է ստեղնաշարից և տեղադրվում է *str* հասցեով: Ֆունկցիան վերադարձնում է *str* հասցեն:

1.7 int puts (char *str)

Տողը *str* հասցեով արտածվում է էկրանի վրա:

1.8 int sprintf (char *buf, “Control string”, el1, el2,….)

Ֆունկցիան *printf()* ֆունկցիայի նմանակն է այն տարբերությամբ, որ արտածումը կատարվում է *buf* տողի մեջ:

2. Տողային ֆունկցիաներ

2.1 unsigned strlen (char * str)

Վերադարձվող արժեքը՝ *str* տողի երկարությունը բայտերով առանց \0-ի:

2.2 int strcmp(const char * s1, const char * s2)

Համեմատում է 2 տող: Վերադարձվող արժեքը՝
0 – տողերը հավասար են
>0 – տող s1 > տող s2-ից
<0 – տող s1 < տող s2-ից
Անհավասարությունը վերաբերում է առաջին ոչ հավասար սիմվոլների ASCII կոդերին:

2.3 char * strcpy(char * s1, const char * s2)

Պատճենահանում է *s2* տողը *s1* հասցեով: Ֆունկցիան վերադարձնում է *s1* հասցեն:

3. Մաթեմատիկական ֆունկցիաներ

3.1 double pow(double base, double exp)

Ֆունկցիան վերադարձնում է *base*-ի *exp* աստիճանի արժեքը:

3.2 double cos(double arg)

Ֆունկցիան վերադարձնում է *arg*-ի *cos* արժեքը:

arg-ը տրվում է ռադիանով:

3.3 double sin (double arg)

Ֆունկցիան վերադարձնում է *arg-ի sin* արժեքը:
arg-ը տրվում է ռադիանով:

3.4 double tan (double arg)

Ֆունկցիան վերադարձնում է *arg-ի* տանգենսի արժեքը:
arg-ը տրվում է ռադիանով:

3.5 double log(double arg)

Ֆունկցիան վերադարձնում է *arg-ի* բնական լոգարիթմի արժեքը:

3.6 double sqrt (double arg)

Ֆունկցիան վերադարձնում է *arg-ի* քառակուսի արմատի արժեքը:

3.7 double fabs (double arg)

Ֆունկցիան վերադարձնում է *arg-ի* բացարձակ արժեքը:

3.8 double exp (double arg)

Ֆունկցիան վերադարձնում է *e* հաստատունի *arg-ի* աստիճանը:

4. Այլ ֆունկցիաները

4.1 int atoi(char *str)

Ֆունկցիան վերադարձնում է *int* տիպի արժեքը, որը ստացվել է *str* տողի թվանշանային սիմվոլներից՝ սկսված առաջինից: Հասնելով արաջին ոչ թվանշանային սիմվոլին՝ ֆունկցիան ավարտում է աշխատանքը:

4.2 void * malloc(unsigned size)

Ֆունկցիան հատկացնում է *size* քանակով բայտեր դինամիկ հիշողությունում և վերադարձնում է առաջին բայտի հասցեն:

4.3 void free(void *ptr)

Ֆունկցիան ազատում է դինամիկ հիշողության տարածք՝ *ptr* հասցեով:

Գրականություն

1. Прата С. Язык программирования С. Лекции и упражнения. Диасофт. Киев. 2004.
2. Каррано Ф., Причард Дж. Абстракция данных и решение задач на С++. Вильямс. Москва. 2003.
3. Дейтел Х.М., Дейтел П.Дж. Как программировать на С++. 3-е издание. БИНОМ. Москва. 2001.

Ինֆորմատիկա (ծրագրավորում C/C++ լեզուներով)
Լաբորատոր աշխատանքների մեթոդական ցուցումներ

Կազմողներ՝ պրոֆ. Արմենակ Պալյան,
տ.գ.թ. Արեգ Պալյան

Խմբագիր՝ Ն. Խաչատրյան