2-лекция

C# тілінің элементтері

Сұрақтар:

1. Тілдің құрамы
2. Атаулар, айнымалылар және константалар
3. C# тіліндегі мәліметтер типтері
4. C# тілінің константалары
5. C# тілінің негізгі операциялары
6. Қарапайым енгізу-шығару операцияларын орындау

1. Тілдің құрамы

Кез келген табиғи тілдің мәтінінде *символдар, сөздер, сөз тіркестері* және *сөйлемдер* болады. Осындай элементтер программалау тілдерінде де болады, мұнда бірақ сөздер – *лексемдер* (қарапайым конструкциялар), сөз тіркестері – *өрнектер* деп, ал сөйлемдер – *операторлар* деп аталады.

Лексемдер симолдардан тұрады, өрнектер – лексемдер мен символдардан, ал операторлар – символдардан, өрнектерден және лексемдерден тұрады.

• Тілдің алфавиті немесе оның символдары — бұл бөлінбейтін негізгі белгілер, солардан тілдің барлық мәтіндері құрастырылады. C# тілінің алфавиті ретіне Unicode символдары қолданылады. Unicode кодтары барлық қолданылатын алфавиттерді бірден бейнелей алады. Оның алғашқы 128 символы ANSI-кодтар кестесіне сәйкес келеді.

• Лексем (token-токен) немесе қарапайым конструкция — өзіндік мағынасы бар тілдің ең кіші бірлігі. Олардың құрамы:

1. Идентификаторлар;
2. Түйінді сөздер;
3. Операция таңбалары;
4. Айыру таңбалары;
5. Константалар (литералдар).

2. Атаулар, айнымалылар және константалар

C# тілінің алфавиті Си ++ тілімен бірдей деуге болады:

- ✓ бас және кіші латын әріптері;
- ✓ 0-ден 9-ға дейінгі араб цифрлары;
- ✓ арнайы белгілер: [] {} , . ( ) + - / * \ | % ; : ? < > = ! & # ^ " ' ;
- ✓ айыру символдары: бос орын (пробел), табуляция символы, жаңа жолға көшу таңбасы;

Символдардан лексемдер құралады, олар:

- ➢ идентификаторлар;
- ➢ түйінді сөздер;
- ➢ операция таңбалары;
- ➢ константалар, яғни тұрақтылар;
- ➢ бөлгіштер (жақша, нүкте, үтір, айыру символдары).

Идентификаторлар – программалық объектінің аты (атауы). Әріп немесе астын сызу таңбасы идентификатордың бірінші символы болуы мүмкін, бірақ цифр бола алмайды. Идентификаторлар мысалдары:
X, bc, A12, Bagasy, BITES_PER_WORD , aty_goni

Түйінді сөздер – компилятор үшін арнайы мәні бар қордағы (резервтегі) идентификаторлар.

Операция таңбалары – операндтармен (мәндермен) іс-әрекет атқару үшін қолданылатын бір немесе бірнеше символдар.

Тұрақтылар – программа орындалу барысында өзгермейтін шамалар.

Комментарийлер – компилятор үшін маңызы жоқ программаның бөлігі және программа мәтінін оқу ыңғайлы болуы үшін қолданылады. Ол /* және */ қоршалып тұрады немесе жол соңында // символынан басталады.

C# тілінің түйінді сөздері тізімі

abstract	do	in	protected	true
as	double	int	public	try
base	else	interface	readonly	typeof
bool	enum	internal	ref	uint
break	event	is	return	ulong
byte	explicit	lock	sbyte	unchecked
case	extern	long	sealed	unsafe
catch	false	namespace	short	ushort
char	finally	new	sizeof	using
checked	fixed	null	stackalloc	virtual
class	float	object	static	void
const	for	operator	string	volatile
continue	foreach	out	struct	while
decimal	goto	override	switch	
default	if	params	this	
delegate	implicit	private	throw	

3. C# тіліндегі мәліметтер типтері

C# тілі қатаң түрде типтелген тіл, яғни әрбір айнымалы немесе объект данасы белгілі бір типке жатқызылуы тиіс, бұл орындалатын амалдардың дұрыстығын тексеру мүмкіндігін береді.

Егер элементтер құрастырылуын негізге алсақ, онда типтер қарапайым және *құрылымды* болып бөлінеді.

Жасаушыға байланысты типтер құрамдас және программалаушы анықтаған болып бөлінеді.

Статикалық типтегі мәліметке жады бірден бөлінеді де, динамкалық типтегі мәліметке жады оны жариялау кезінде емес, оны программада пайдалану кезінде бөлінеді.

C# тіліндегі мәліметтер типтерін жіктеу түрлері

C# тілінде құрамдас 15 тип бар, олардың сегізі бүтін сандық типке жатады. Олар — C# түйінді сөздерімен анықталады да, кез келген программада қолданыла береді.

bool	Логикалық немесе бульдік тип, оның мәндері АҚИҚАТ/ЖАЛҒАН болып келеді
byte	8-разрядты таңбасыз бүтін санды тип
char	Символдық тип
string	Тіркестік тип
decimal	Қаржылық есептеулерге арналған сандық тип
double	Екі еселенген дәлдіктегі жылжымалы нүктелі сандық тип
float	Жылжымалы нүктелі сандық тип
int	Бүтін сандық тип
long	Ұзын бүтін санды бейнелеуге арналған тип
sbyte	8-разрядты таңбалы бүтін санды тип
short	Қысқа бүтін санды бейнелеуге арналған тип
uint	Таңбасыз бүтін санды тип
ulong	Ұзын таңбасыз бүтін санды бейнелеуге арналған тип
ushort	Қысқа таңбасыз бүтін санды бейнелеуге арналған тип
object	Барлық құрамдас және тұтынушы типтері осыдан бастау алады

C# тілінің құрамдас типтері

Құрамдас типтер

bool	Логикалық тип
byte	8 бит, таңбасыз бүтін тип
sbyte	8 бит, таңбалы бүтін тип
string	тіркестік тип
decimal	қаржылық сан типі
double	екі есе дәлдіктегі нақты
float	нақты сан типі
int	бүтін сан типі
uint	таңбасыз бүтін тип
char	символдық тип
long	ұзын бүтін сан типі
ulong	ұзын таңбасыз бүтін
short	қысқа бүтін сан типі
ushort	қысқа таңбасыз бүтін
object	объектілік тип

C# тіліндегі 8 сандық типтер бар, олар .NET кітапханасындағы анықталған стандартты типтермен (бұл Common Language Specification – CLS ортасында анықталған негізгі типтер) сәйкестендіріп жасалған. Бұл типтер келесі кестеде көрсетілген, онда олардың жадыдан алатын байт көлемі және өзгеру диапазондары да келтірілген.

Кестеден олардың таңбалы және таңбасыз нұсқаларының бар екені көрініп тұр. Таңбалы типтер оң және теріс сандарды сақтайды, ал таңбасыз типтер тек оң мәндерді сақтайды.

Айнымалылардың int, long,ulong типтеріне мысал:

```
int myInt = 1;
long myLong = -1234;
ulong myUlong = 9;
```

Логикалық тип			
Тип аты	**Жүйелік тип**	**Мәндері**	**Көлемі (ені)**
Bool	System.Boolean	true, false	8 бит
Арифметикалық бүтін сан түріндегі типтер			
Тип аты	**Жүйелік тип**	**Өзгеру диапазоны**	**Көлемі (ені)**
Sbyte	System.SByte	-128 — 127	Таңбалы, 8 Бит
Byte	System.Byte	0 — 255	Таңбасыз, 8 Бит
Short	System.Short	-32768 —32767	Таңбалы, 16 Бит
Ushort	System.UShort	0 — 65535	Таңбасыз, 16 Бит
Int	System.Int32	≈(-2*10^9 — 2*10^9)	Таңбалы, 32 Бит
Uint	System.UInt32	≈(0 — 4*10^9)	Таңбасыз, 32 Бит
Long	System.Int64	≈(-9*10^18 — 9*10^18)	Таңбалы, 64 Бит
Ulong	System.UInt64	≈(0— 18*10^18)	Таңбасыз, 64 Бит
Арифметикалық жылжымалы нүктелі сандар түріндегі типтер			
Float	System.Single	+1.5*10^-45 - +3.4*10^38	7 цифр
Double	System.Double	+5.0*10^-324 - +1.7*10^308	15-16 цифр
Арифметикалық бекітілген нүктелі сандар түріндегі тип			
Decimal	System.Decimal	+1.0*10^-28 - +7.9*10^28	Цифрлардың 28-29 таңбасы

Программада кездесетін константалардың жазылуына қарай, яғни солардың сыртқы бейнесіне сәйкес белгілі бір тип тағайындалады. Егер ол типті өзгерту керек болса, онда санның соңына жалғастырылып керекті типтің атына сәйкес бір әріп – **L, l** (long) немесе **U, u** (unsigned) жазылады. Мысалы, 32L константасының типі **long** және ол компьютердің жедел жадында 4 байт орын алады. Қажет болса, **L** және **U** әріптерін қатарластыра да қолдануға болады, мысалы, **0x22UL** немесе **05Lu**.

Мәліметтердің decimal типі үтірден кейінгі 28 таңбаға дейін сақтай алады, ол көбінесе ақшалық мәндерді сақтау үшін қолданылады.

Айнымалыға немесе константаға float типін бергенде, мән соңына "F" или "f " символын қоюға, мысалы:

float myFloat1 = 1.2f;

Ал тип double болса, "D" или "d" символын қоюға болады, бірақ айнымалы нақты анықталған соң, әріптерді қоймаса да болады. Мысалы:

double myDoublel = 1234.5678;

double myDouble2 = 1234.5678d;

Символдық тип

Символдық тип char 16-биттік Unicode символ болып табылады.

Unicode – әлем тілдерінің барлық символдарын электрондық формада бейнелеуге арналған стандарт.

Символдық типтер

Тип аты	Жүйелік тип	Өзгеру диапазоны	Дәлдігі
`Char`	`System.Char`	U+0000 – U+ffff	16 биттік Unicode символдары
`String`	`System.String`	Unicode символдары тіркесі	

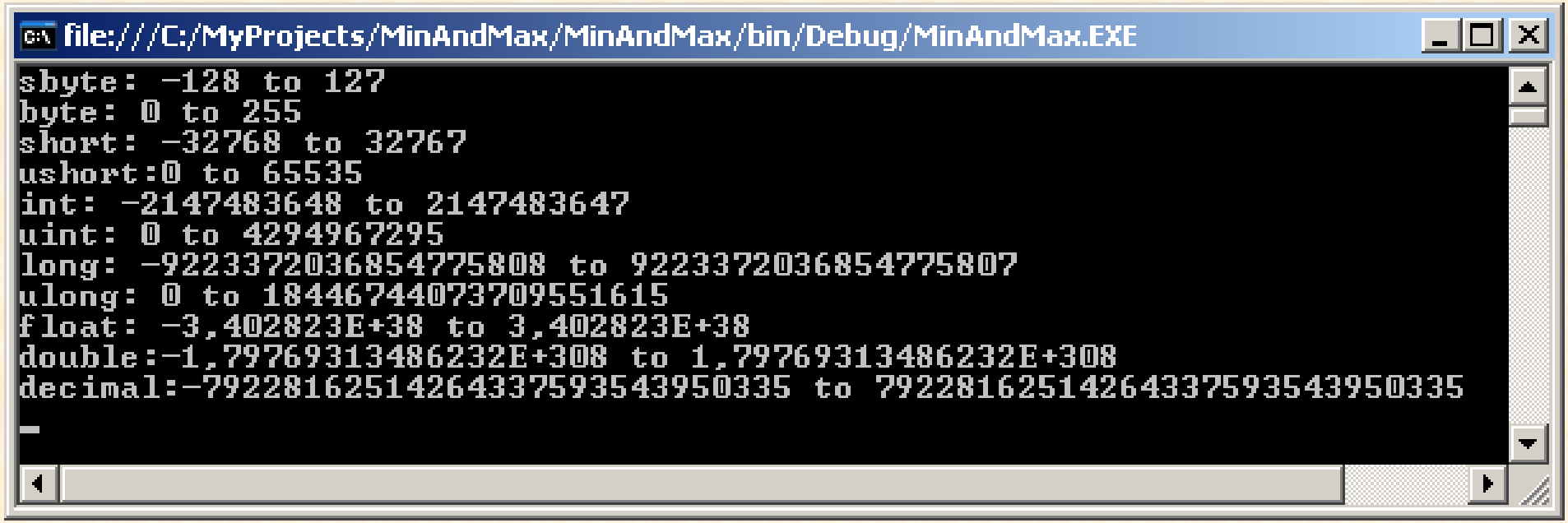
Төменде мәліметтердің 11 сандық типтерінің әрқайсысы үшін ең кіші (минимал) және ең үлкен (максимал) мәндерін көрсететін программа мысалы келтірілген.

```
// MinAndMax.cs программасы
 using System;
 class MinAndMax
 {  public static void Main()
     { Console.WriteLine("sbyte: {0} to {1}",sbyte.MinValue,
                                                 sbyte.MaxValue);
       Console.WriteLine("byte: {0} to {1}",byte.MinValue,
                                                 byte.MaxValue);
       Console.WriteLine("short: {0} to {1}",short.MinValue,
                                                 short.MaxValue);
       Console.WriteLine("ushort:{0} to {1}",ushort.MinValue,
                                                 ushort.MaxValue);
       Console.WriteLine("int: {0} to {1}",int.MinValue,
                                                 int.MaxValue);
```

```
        Console.WriteLine("uint: {0} to {1}",uint.MinValue,
                                              uint.MaxValue);
        Console.WriteLine("long: {0} to {1}",long.MinValue,
                                              long.MaxValue);
        Console.WriteLine("ulong: {0} to {1)",ulong.MinValue,
                                              ulong.MaxValue);
        Console.WriteLine("float: {0} to {1}",float.MinValue,
                                              float.MaxValue);
        Console.WriteLine("double:{0} to {1}",double.MinValue,
                                              double.MaxValue);
        Console.WriteLine("decimal:{0} to {1}",decimal.MinValue,
                                              decimal.MaxValue);
    }
}
```

Программа жұмысы нәтижесі:

sbyte : -128 to 127
byte: 0 to 255
short: -32768 to 32767
ushort: 0 to 65535
int: -2147483648 to 2147483647
uint: 0 to 4294967295
long: -9223372036854775808 to 9223372036854775807
ulong: 0 to 18446744073709551615
float : -3.402823E+38 to 3.402823E+38
double: -1.7976931348623E+308 to
1.797693134862Э2E+308
decimal: -79228162514264337593543950335 to
79228162514264337593543950335


file:///C:/MyProjects/MinAndMax/MinAndMax/bin/Debug/MinAndMax.EXE
sbyte: -128 to 127
byte: 0 to 255
short: -32768 to 32767
ushort:0 to 65535
int: -2147483648 to 2147483647
uint: 0 to 4294967295
long: -9223372036854775808 to 9223372036854775807
ulong: 0 to 18446744073709551615
float: -3,402823E+38 to 3,402823E+38
double:-1,7976931348623E+308 to 1,7976931348623E+308
decimal:-79228162514264337593543950335 to 79228162514264337593543950335

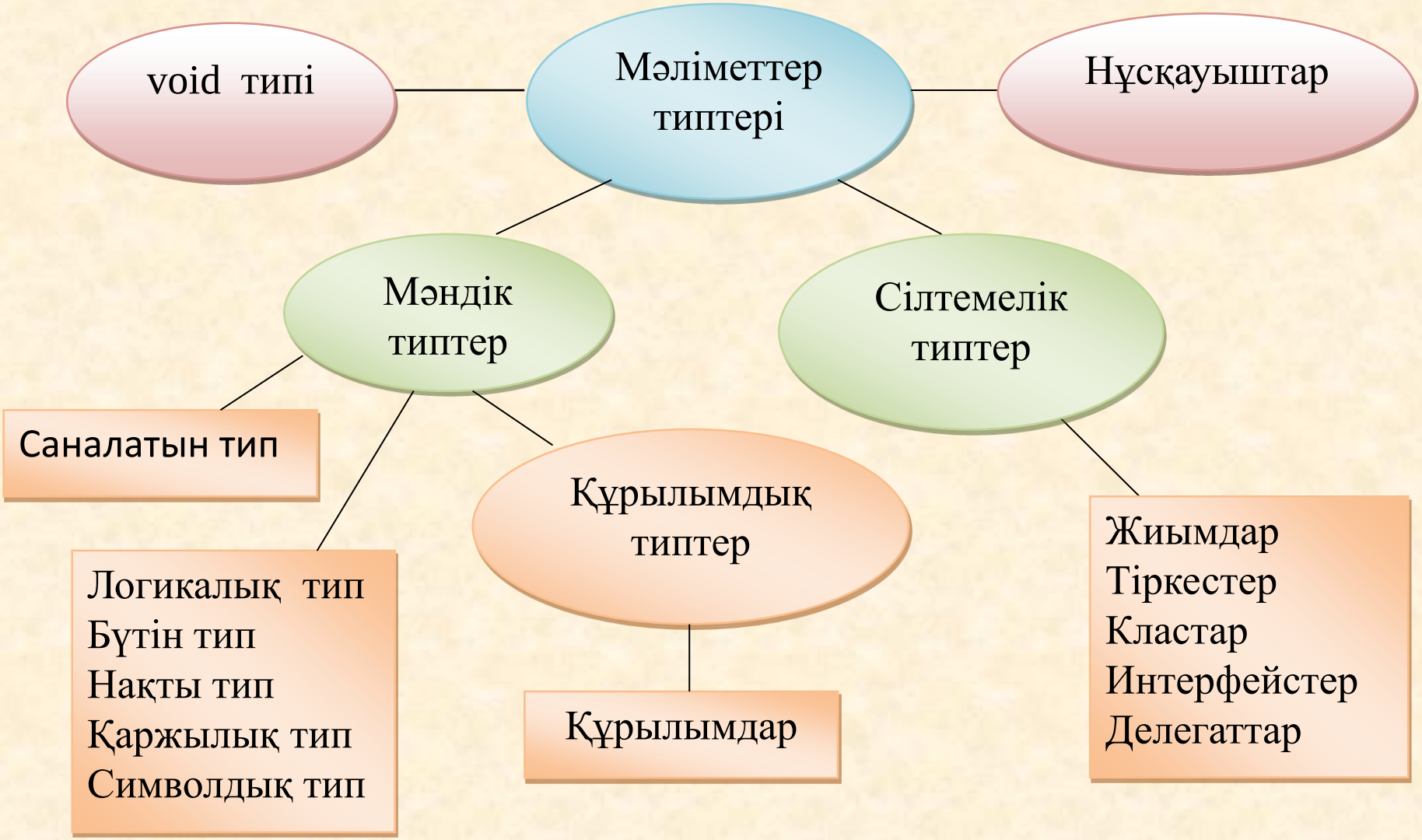
C# тілінде bool типі бар, ол true немесе false сияқты екі мәнді қабылдай алады. Салыстыру операцияларының нәтижелері (==, !=, <, >, <= и >=) — bool типінде болады. Bool типі бүтін типке келтіріледі (true – 1, ал false — 0), бірақ ол тікелей орындалуы тиіс.

Char типі бір символды, ал string — бірнеше символдардан тұратын сөз тіркестерін сақтау үшін қажет.

Char типі бүтін типтерден өзгеше болып келеді және де оны **sbyte** немесе **byte** типтерімен шатастырмау керек. Char типіндегі айнымалылар 16 бит (бірақ ол short немесе ushort типтерінен өзгеше) орын алады.

C# тілінде мәліметтер типтерінің элементтерді сақтау тәсіліне қарай тағы екі категориясы (санаты) бар: *мәндер типтері* (value types) және *сілтемелік типтер*. *Мәндер типтері* дегеніміз шамаға компилятор бөліп берген компьютер жадындағы биттер тізбегі.

Сілтемелік типтер мәліметтердің өздерін емес, олардың адрестерін сақтайды, мұндағы мән компьютердің динамикалық жадындағы басқа бір объектіге (үйіндіге – кучаға) сілтеу арқылы жасалады. Мәндер типтеріне *логикалық* тип, *арифметикалық* тип, *құрылымдар* және *саналатын* (перечисления) типтер жатады. Сілтемелік типтерге *жиымдар*, *сөз тіркестері* (жолдар) және *кластар* жатады.


void типі
Мәліметтер типтері
Нұсқауыштар
Мәндік типтер
Сілтемелік типтер
Саналатын тип
Логикалық тип
Бүтін тип
Нақты тип
Қаржылық тип
Символдық тип
Құрылымдық типтер
Құрылымдар
Жиымдар
Тіркестер
Кластар
Интерфейстер
Делегаттар

4. C# тілінің константалары

Константа	Форматы	Мысалдар
Бүтін Соңында тип-ті беретін U,L,UL,LU,IU тәрізді жұр-нақтар болуы мүмкін	Ондық: ондық цифрлар тізбесі, нөлден басталмайды (егер сан 0 болмаса) Он алтылық: 0x немесе 0X таңбаларынан басталатын он алтылық цифрлар (0,l,2,3,4,5,6,7,8,9,A,B,C,D,E,F)	8, 0, 199226 8u, 0LU, 199226L 0xA, 0x1B8, 0X00FF 0xAU, 0x1B8LU, 0X00FFFL
Нақты Соңында тип-ті беретін F, D, M тәрізді жұрнақтар болуы мүмкін	Ондық: [цифрлар].[цифрлар][1] Экспоненциалдық [цифрлар][.][цифрлар]{E|e}[+|-][цифрлар] [2]	5.7, .001, 35. 5.7F, .001d, 35m 0.2E6, .11e-3, 5E10 0.2E6D, .11e-3m, 5E10d
Символдық	Апостроф таңбасымен қоршалған бір немесе екі символ	'A', 'ю', '*', 'db', '\0', '\n\', \012', '\x07\x07'
Тіркестік	Тырнақшаға алынған символдар тіркесі	"Мұнда Азат болды", "\tНәтижесі:r=\0xF5\n"

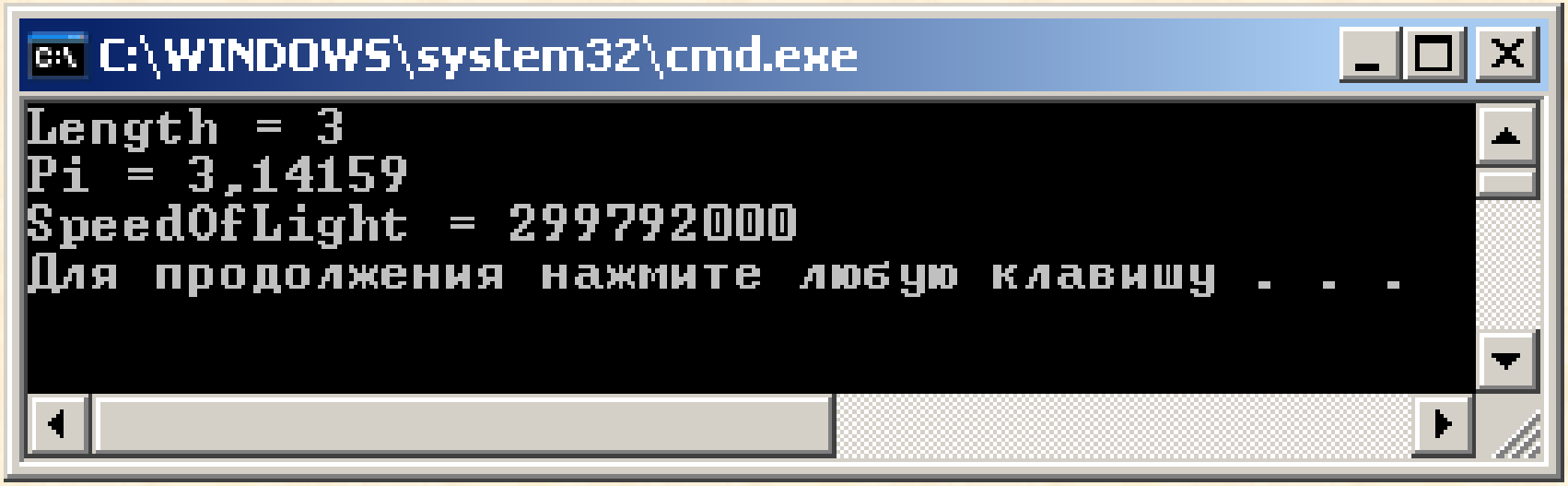
```
/*  Example2_7.cs   константаларды қолдану  */
class Example2_7
{  public  static void Main()
   {
     const  int  Length =   3;
     //  математикалық константа  Пи
     const double Pi = 3.14159;
      //  жарық жылдамдығы   метр/секунд
     const double SpeedOfLight  =  2.99792e8;
     Console.WriteLine("Length = " + Length);
     Console.WriteLine("Pi = " + Pi);
     Console.WriteLine("SpeedOfLight = " +
                                  SpeedOfLight);
  }
}
```

Программа нәтижесі

Length = 3

Pi = 3.14159

SpeedOfLight = 299792000



5. C# тілінің негізгі операциялары

Төменде C# негізгі операциялары олардың *приоритеттері* бойынша берілген. Операндтарының санына қарай олар унарлық, бинарлық және тернарлық болып бөлінеді.

Унарлық (бір орынды) операциялар	
Операция	**Қысқаша сипаттамасы**
`++`	1-ге арттыру
`--`	1-ге кеміту
`~`	разрядтар бойынша терістеу
`!`	логикалық терістеу
`-`	арифметикалық терістеу (унарлық минус)
`+`	унарлық плюс
`new`	жадыны бөлу
`typeof`	типті алу
`checked`	тексерілетін код
`unchecked`	тексерілмейтін код
`(type)x`	типті түрлендіру

Бинарлық (екі орынды) және тернарлық (үш орынды) операциялар	
Операция	**Қысқаша сипаттамасы**
`/`	бөлу
`%`	бөлгендегі қалдық
`+`	қосу
`-`	азайту
`<<`	солға ығыстыру
`>>`	оңға ығыстыру
`<`	аз
`<=`	аз немесе тең
`>`	көп
`>=`	көп немесе тең
`is`	типке жататындығын тексеру
`as`	типті келтіру
`==`	тең
`!=`	тең емес
`&`	разрядтар бойынша конъюнкция (ЖӘНЕ)
`^`	разрядтар бойынша аластайтын НЕМЕСЕ
`|`	разрядтар бойынша дизъюнкция (НЕМЕСЕ)

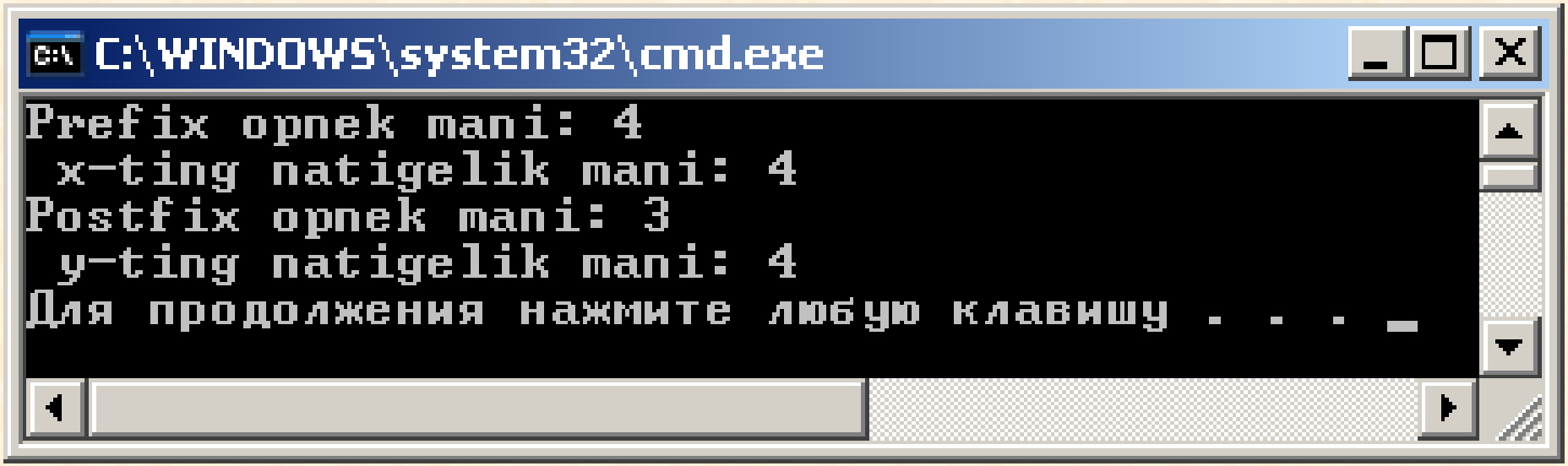
Бинарлық (екі орынды) және тернарлық (үш орынды) операциялар	
Операция	**Қысқаша сипаттамасы**
? :	шартты операция (тернарлық)
*=	көбейтіп алып меншіктеу
/=	бөліп алып меншіктеу
%=	бөлгендегі қалдықты тауып меншіктеу
+=	қосып алып меншіктеу
-=	азайтып алып меншіктеу
<<=	солға ығыстырып меншіктеу
>>=	оңға ығыстырып меншіктеу
&=	разрядтар бойынша ЖӘНЕ операциясын орындап меншіктеу
|=	разрядтар бойынша НЕМЕСЕ операциясын орындап меншіктеу
^=	разрядтар бойынша арифметикалық НЕМЕСЕ операциясын орындап меншіктеу
.	тізбекті түрде есептеу

Қалған операциялар жайлы кейін айтылады.

```
// Инкремент (++) және декремент (--) операциялары
using System
namespace Increment1
{  class Class1
   {  static void Main()
      {
         int x = 3, y = 3;
         Console.Write("Prefix opnek mani: ");
         Console.WriteLine( ++x);
         Console.Write(" x-ting natigelik mani: ");
         Console.WriteLine( x);
         Console.Write("Postfix opnek mani: ");
         Console.WriteLine( y++);
         Console.Write(" y-ting natigelik mani: ");
         Console.WriteLine( y);
      }
   }
}
```

Программа жұмысы нәтижесі:

Prefix opnek mani: 4
x-ting natigelik mani: 4
Postfix opnek mani: 3
y-ting natigelik mani: 4



Көбейту операциясы *int*, *uint*, *long*, *ulong*, *float*, *double*, *decimal* тәрізді арифметикалық типтегі операндтарға қолданылады. Егер екі операнд та бүтін сан болса, бөлінді де бүтін сан болады.

Бөлу операциясы да жоғарыдағы арифметикалық типтегі операндтарға қолданылады. Егер екі операнд та бүтін сан болса, бөлінді де бүтін сан болады, әйтпесе бөлінді типі типтерді түрлендіру ережесіне сәйкес тағайындалады.

Қалдық табу (%) *операциясы* әртүрлі типтегі - бүтін, нақты, қаржылық шамаларға қолданылады. Егер екі операнд та бүтін сан болса, нәтиже x-(x/y)*y формуласымен анықталады. Егер бір операнд нақты сан болса, нәтиже x-n*y (n – x-ті y-ке бөлгендегі бүтін сан) формуласымен анықталады. Мысалы:

```
using System;
namespace ConsoleApplication1
{ class Class1
{  static void Main()
   { int x = 11, y = 4;
     float z = 4;
     Console.WriteLine(  z*y );          // Нәтиже  16
     Console.WriteLine(  z*1e308 );  // Нәтиже  шексіздік
     Console.WriteLine(  x/y );          // Нәтиже  2
     Console.WriteLine(  x/z );          // Нәтиже  2.75
     Console.WriteLine(  x%y );          // Нәтиже  3
     Console.WriteLine(  1e-324/1e-324 ); // Нәтиже  NAN
  }}}
```

Екілік разрядтар бойынша ығыстыру операциялары (<< және >>) тек бүтін санды типтегі екілік жүйедегі операндтарға қолданылады. Бұл операцияларды орындау барысында бірінші операндтар биттері екінші операндта көрсетілген санға сәйкес оңға немесе солға ығыстырылады. Солға ығыстыру (<<) кезінде босатылған разрядтар нөлмен толтырылады. Ал оңға ығыстыру (>>) кезінде бірінші операнд таңбасыз типте болса, босатылған разрядтар нөлмен толтырылады. Басқаша жағдайда мәлімет ығыстырылады. Бұлар int, uint, long,ulong типтері үшін анықталған. Мысал:

```
using system;
namespace ConsoleApplication1
{ class Class1
{  static void Main( )
   { byte a = 3, b = 9;  sbyte  c = 9, d = -9;
     Console.WriteLine(  a<< 1 );       // Нәтиже  6
     Console.WriteLine(  a<< 2);        // Нәтиже  12
     Console.WriteLine(  b >> 1 );      // Нәтиже  4
```

```
    Console.WriteLine(  c >> 1 );       // Нәтиже  4
    Console.WriteLine(  d >> 1 );       // Нәтиже  -5
   }
 }
}
```

Қатынас операциялары (<, <=, >, >=,==,!=) бірінші операндты екінші операндпен салыстырады.

Операндтар арифметикалық типтерде болуы тиіс.

Операция нәтижесі – `true` немесе `false`. Мысалы:

x = = y болғанда, x пен y тең болса, нәтиже `true`

x != y болғанда, x пен y тең болмаса, нәтиже `true`

x < y болғанда, x y-тен кіші болса, нәтиже `true`

x > y болғанда, x y-тен үлкен болса, нәтиже `true`

x <= y болғанда, x y-тен кіші не тең болса, `true`

x >= y болғанда, x y-тен үлкен не тең болса, `true`

Екілік разрядтар бойынша орындалатын операциялар (&, | , ^) тек бүтін санды типтегі екілік жүйедегі операндтарға қолданылады. Бұл операцияларды орындау барысында операндтар биттер бойынша қарастырылады (бірінші операндтың бірінші биті екінші операндтың бірінші битімен, бірінші операндтың екінші биті екінші операндтың екінші битімен, т.с.с. салыстырылады). Бұлар int, uint, long,ulong типтері үшін анықталған.

Разрядтық конъюнкцияда немесе *разрядтық ЖӘНЕ* (операция & болып белгіленеді) операциясында логикалық көбейту амалы орындалады да, тек екі операндтың да сәйкес орындардағы разрядтары 1-ге тең болған жағдайда нәтижелік бит 1-ге тең болады.

1-операнд	0	0	1	1		1-операнд	1	0	1	0	0	1	1
2-операнд	0	1	0	1		2-операнд		1	1	0	1	0	1
Нәтиже	0	0	0	1		Нәтиже	0	0	1	0	0	0	1

Разрядтық дизъюнкцияда немесе *разрядтық НЕМЕСЕ* (операция | болып белгіленеді) операциясында логикалық қосу амалы орындалады да, екі операндтың тек біреуінің немесе екеуінің де сәйкес орындардағы разрядтары 1-ге тең болған жағдайда нәтижелік бит 1-ге тең болады.

1-операнд	0	0	1	1	1-операнд	1	0	1	0	0	1	1
2-операнд	0	1	0	1	2-операнд		1	1	0	1	0	1
Нәтиже	0	1	1	1	Нәтиже	1	1	1	0	1	1	1

Разрядтар бойынша аластайтын НЕМЕСЕ (операция ^ болып белгіленеді) операциясында екі операндтың тек біреуінің ғана сәйкес орындарындағы разряды 1-ге тең болған жағдайда нәтижелік бит 1-ге тең болады.

1-операнд	0	0	1	1	1-операнд	1	0	1	0	0	1	1
2-операнд	0	1	0	1	2-операнд		1	1	0	1	0	1
Нәтиже	0	1	1	0	Нәтиже	1	1	0	0	1	1	0

Мысал:

```
using System;
namespace ConsoleApplication1
{ class Class1
{  static void Main()
   { Console.WriteLine(  6 & 5 );      // Нәтиже  4
     Console.WriteLine(  6 | 5 );      // Нәтиже  7
     Console.WriteLine(  6 ^ 5 );      // Нәтиже  3
   }
 }
 }
```

Программа нәтижесі:

6 & 5 = 4

6 | 5 = 7

6 ^ 5 = 3

Операция:	&	|	^
1-операнд	1 1 0	1 1 0	1 1 0
2-операнд	1 0 1	1 0 1	1 0 1
Нәтиже	1 0 0	1 1 1	0 1 1

Шартты операция (? :) – тернарлық операция үш операндтан тұрады. Форматы:

`1_операнд ? 2_операнд : 3_операнд`

Бұл C++ тіліндегідей болып орындалады.

Мысалы:

```
using system;
namespace ConsoleApplication1
{ class Class1
   {  static void Main()
      { int a = 11, b = 4;
        int max = b > a ? b : a
        Console.WriteLine(  max );      // Нәтиже  11
      }
   }
}
```

6. Қарапайым енгізу-шығару операцияларын орындау

Кез келген программада мәліметтер пернелерден енгізіліп, экранға шығарылады. Стандартты енгізу-шығаруы құрылғылары консоль деп аталады.

C# тілінде енгізу-шығару операторлары жоқ, оның орнына стандартты объектілер қолданылады. Консольмен жұмыс істеу үшін **`Console`** класы пайдаланылады, ол **`System`** атаулар кеңістігінде анықталған. Осы кластың **`Write`** және **`WriteLine`** тәсілдері мысалдарда қолданылған болатын.

Енді осы класқа тағы бір мысал қарастырайық.

```
using System;
namespace ConsoleApplication1
{ class Class1
  {static void Main()
    {  int  i = 3;
       double  y = 4.12;
       decimal  d = 600m;
       string  s = "Азат";
       Console.WriteLine( "i = " + i );        //  1
       Console.WriteLine( "y = {0} \nd = {1}", y, d );  // 2
       Console.WriteLine( "s = " + s );        // 3
    }
  }
}
```

Нәтижесі:

```
i = 3
y = 4.12
d = 600
s = Азат
```

3. C# тілінің операторлары

Кез келген программаны алгоритм блоктарының өзара байланысуына қарай үш түрлі басқару құрылымынан жасауға болады. Оларды құрылымдық программалаудың базалық құрастырғыштары (конструкциялары) деп атайды.

Бірнеше операторлардың тізбектей орындалуынан тұратын құрастырғыш *реттік* (сызықтық) деп аталады. Қандай да бір шарттың орындалуына тәуелді құрастырғышты *тармақталу* деп атаймыз. *Цикл* операторлар тізбегінің бірнеше рет қайталап орындалуын білдіреді.

Оператор – тілдің қарапайым сөйлемі, ол белгілі бір әрекет немесе амал орындап, **;** таңбасымен аяқталады.

Сызықтық (реттік) құрылым бірінен кейін бірі орындалып тізбектеле орналасқан бір-неше операторлардан тұрады.

Тармақты – шартқа байланысты екі оператордың бірінің орындалуы

Цикл – операторлар бөлігінің бірнеше рет қайталана орындалуы.

Тармақталу операторлары: if, switch.

Цикл операторлары: while, do while, for, foreach

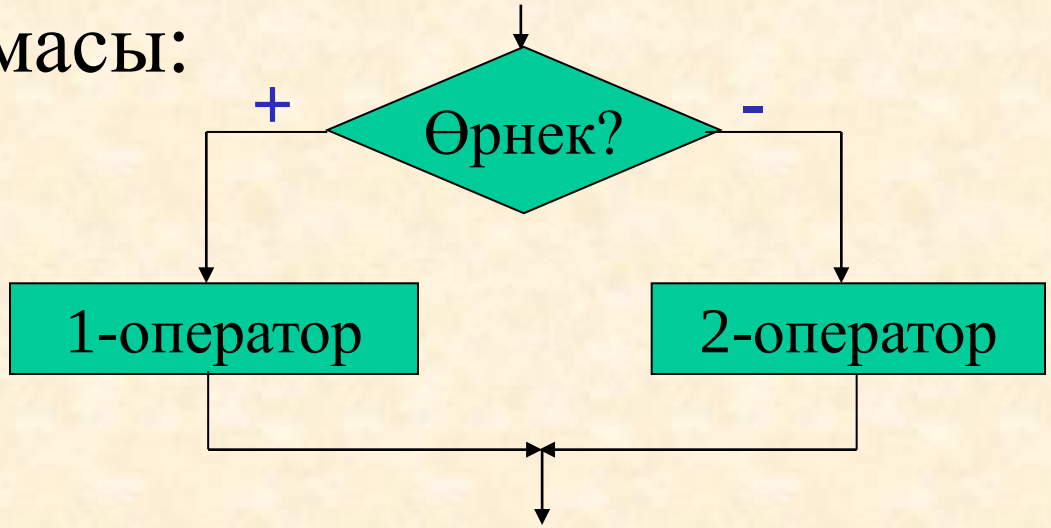
Басқаруды беру операторлары: goto, break

if шартты операторы есептеу жолының екі бағытта тармақталуын жүзеге асырады.

Оператор форматы:

if (өрнек) 1-оператор; [else 2-оператор;]

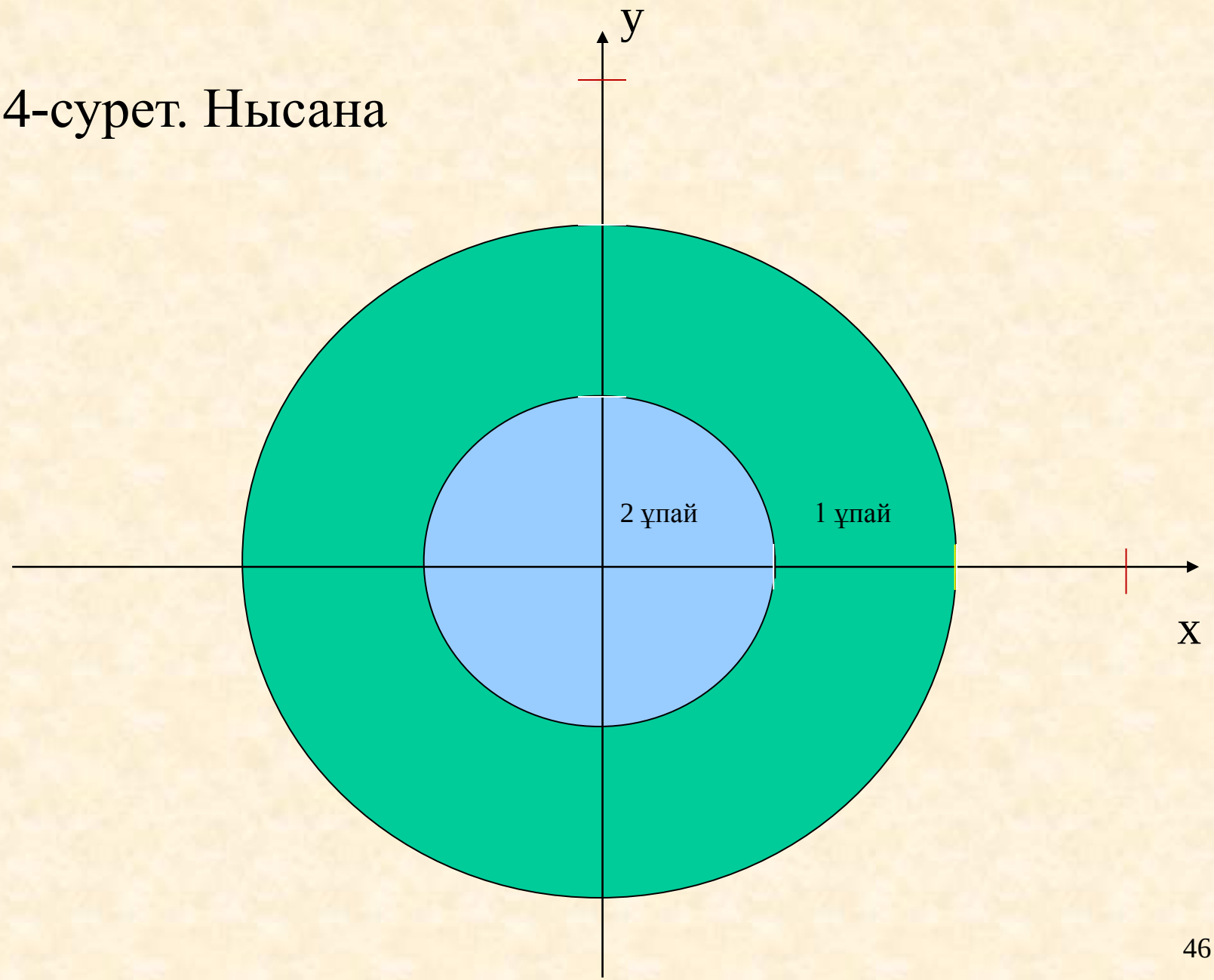
3-сурет. Шартты операторлық құрылымдық схемасы:



Мысалы:

```
using System;
namespace ConsoleApplication1
{  class Class1
   {  static void Main( )
      {  Console.WriteLine( "Введите координату x: " );
         string buf = Console.ReadLine();
         double x = Convert.ToDouble( buf );
         Console.WriteLine( "Введите координату y: " );
         buf = Console.ReadLine();
         double y = double.Parse( buf );
         int kol = 0;
         if      ( x * x + y * y < 1 ) kol = 2;
         else if ( x * x + y * y < 4 ) kol = 1;
         Console.WriteLine( "Результат = {0} очков", kol );
      }
   }
}
```

4-сурет. Нысана

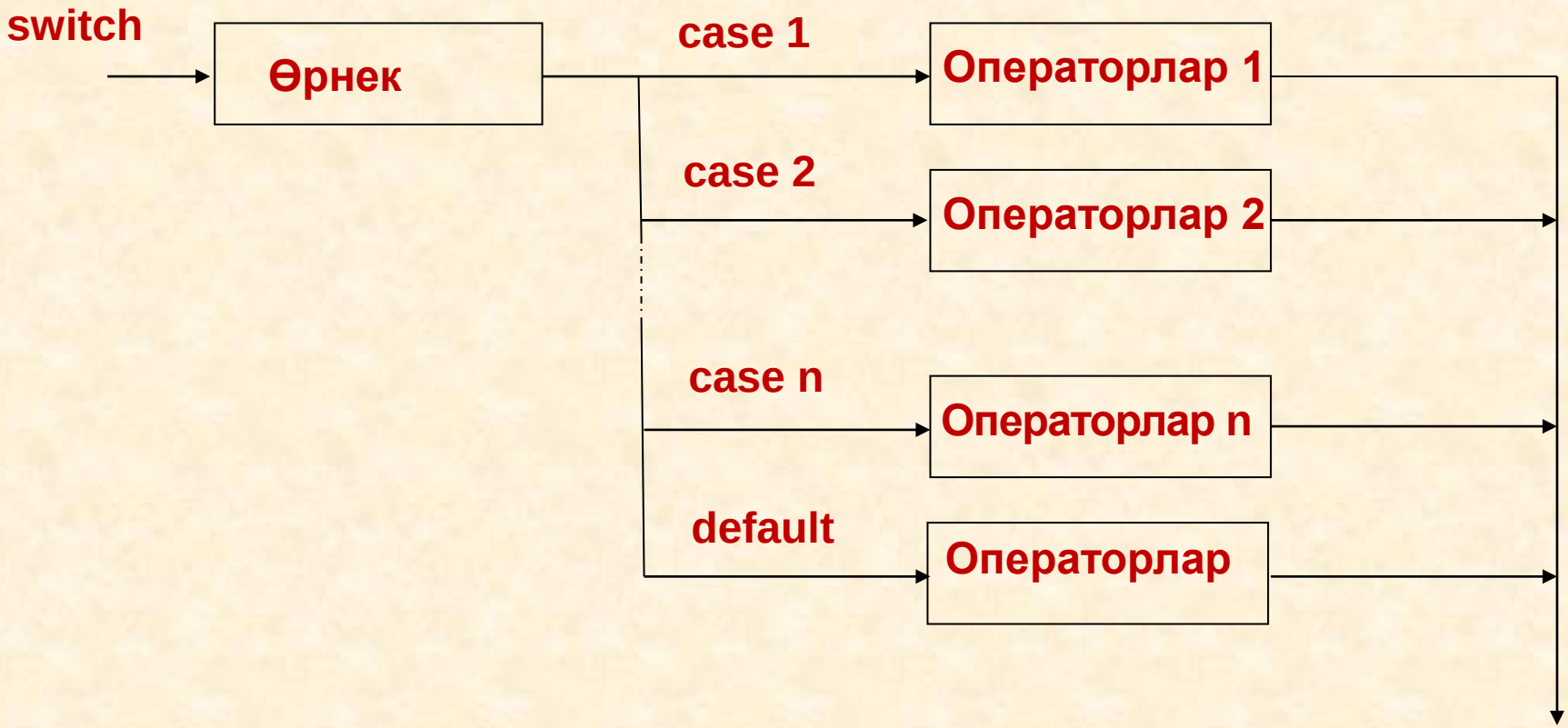


Switch операторы

switch (ауыстырғыш) операторы есептеу процесін бірнеше тармаққа бөліп жібереді. Оның алгоритмдік схемасы келесі слайдта көрсетілген. Оператор форматы:

```
switch ( өрнек )
{
  case 1_тұрақты_өрнек: [1_операторлар_тізімі]
  case 2_тұрақты_өрнек: [2_операторлар_тізімі]
  ...
  case n_тұрақты_өрнек: [n_операторлар_тізімі]
  [default: операторлар ]
}
```

Оператордың құрылымдық схемасы суретте көрсетілген.



Қарапайым 4 амалды орындайтын калькулятор жұмысын атқаратын программа:

```
using System;
namespace ConsoleApplication1
{  class Class1
   {  static void Main( )
      {   string buf;
          double a, b, res;
          Console.WriteLine (" 1-operandti engiz : ")
          buf = Console.ReadLine( );
          a = double.Parse( buf );
          Console.WriteLine(  " Operacia tangbasin engiz : "  );
          char op = (char)Console.Read();
          Console.ReadLine():
          Console.WriteLine( " 2-operandti engiz :"  );
          buf = Console.ReadLine( );
          b = double.Parse( buf );
          bool  ok = true;
```

```
switch (op)
   { case '+': res = a + b; break;
case '-': res = a - b; break;
case '*': res = a * b; break;
case '/': res = a / b; break;
default : res = double.NAN; ok = false; break;
    }
if (ok) Console.WriteLine(" Natigesi : " + res); else
Console.WriteLine(" Belgisiz amal ");
}
 }
}
```

7. Циклдік алгоритмдер

C++ тілінде үш циклдік оператор бар, олардың жазылу форматтары:

while (өрнек) оператор;
do оператор while өрнек;
for (инициализация; өрнек; модификация) оператор;

Үйге тапсырма: Циклдік операторларға мысал келтіру.

C# тілінде 4 басқаруды беру операторы бар:

- ✓ goto шартсыз өту операторы;
- ✓ break циклдан шығу операторы;
- ✓ continue циклдың келесі итерациясына көшу операторы;
- ✓ return функциядан қайтару операторы.

goto шартсыз өту операторының форматы:

goto белгі;

break циклдан шығу операторы цикл операторларының ішінде қолданылады немесе switch операторынан шығуды қамтамасыз етеді.

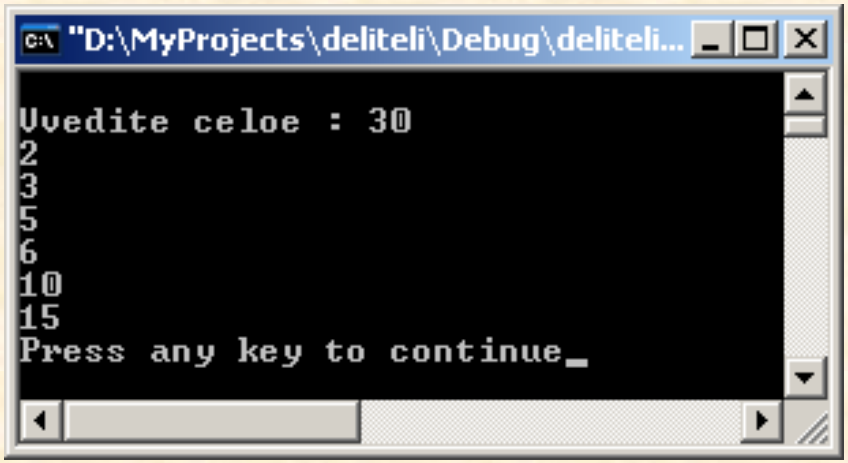
continue циклдың келесі итерациясына көшу операторы цикл денесінің аяғына дейінгі операторларды бос жіберіп, басқаруды келесі итерацияға береді.

return функциядан қайтару операторы функцияның орындалуын аяқтап, басқаруды шақыру нүктесіне береді.

цикл – әзірше

Келесі мысалда енгізілген **num** бүтін санының барлық бөлгіштері анықталады.

```
// Берілген оң бүтін санның бөлгіштерін табу
#include <iostream.h>
int main(){
  int num;
  cout << "\nSan engizingiz : "; cin >> num;
  int half = num/2;      // санның жартысы
  int div = 2;    // бөлгішке тексеру
  while (div <= half) {
     if (!(num % div)) cout << div <<"\n";
     div++;
  }
  return 0;
}
```



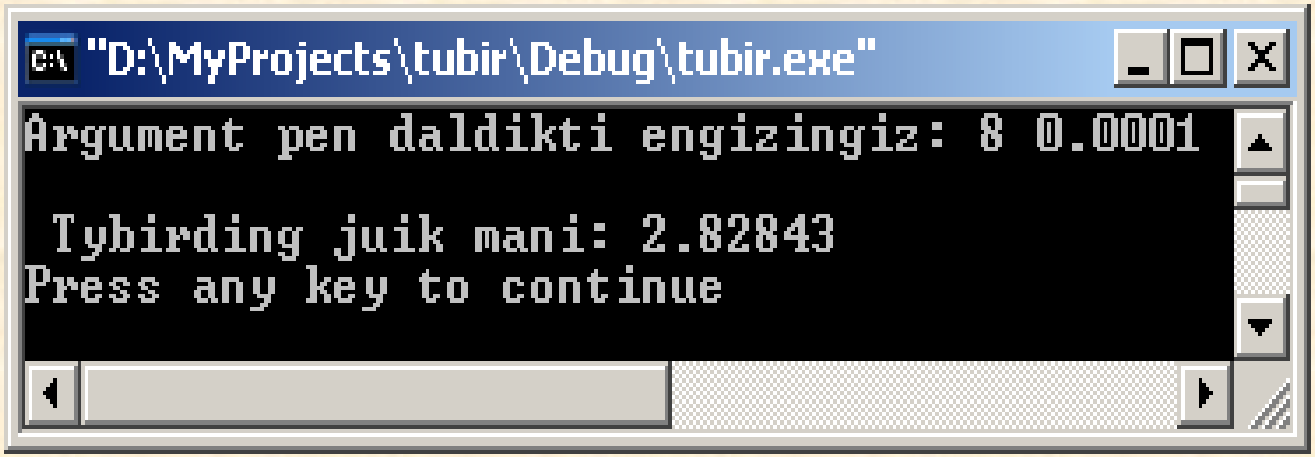
цикл – дейін

Мысал. Бұл программада енгізілген нақты аргумент – **x**-тің түбірін берілген дәлдікпен – eps жуық шамамен итерациялық формула арқылы анықтаймыз:

$y_n = (y_{n-1} + x/y_{n-l})$,

мұндағы y_{n-l} – түбірдің алдыңғы жуық мәні (есептеу алдында бұл мән кез келген оң сан ретінде таңдалады), y_n – түбірдің келесі табылған жуық мәні. Есептеу процесі түбірдің анықталған екі жуық мәндері айырмасының абсолюттік мәні берілген дәлдіктен төмен болған сәтте тоқталады. Абсолюттік мәнді табу үшін стандартты fabs() функциясы қолданылады, ол <math.h> тақырыптық файлында анықталады.

```
  #include <iostream.h>
#include <math.h>
int main(){
double x, eps;          // аргумент пен далдік
double Yp, Y = 1;   /* тубірдін алдынгы жане
                                  келесі жуык мандері  */
cout << "Argument pen daldikti engizingiz: ";
cin >> x >> eps;
do{
Yp = Y;
Y = (Yp + x/Yp)/2;
}while (fabs(Y - Yp) >= eps);
cout << "\n Tybirding juik mani: "
        << Y << endl;
return 0;
  }
```



Келесі мысалда енгізілген **num** бүтін санының барлық бөлгіштері анықталады.

```
#include <iostream.h>
int main(){
int num, half, div;
cout << "\nSan engizingiz : ";  cin >> num;
for (half = num / 2, div = 2; div <= half; div++)
  if  (!(num % div))   cout << div << "\t";
return 0;
  }
```

Тыңдағандарыңызға

рахмет!