Подпрограммы C#

Бағыныңқы программа – атауы бар және өзінің жеке тапсырмасын шешетін программаның жеке бөлігі. Ішкі программа негізгі программаның басында орналасады және атын көрсету арқылы негізгі бағдарламадан іске қосуға (шақыруға) болады.

Ішкі бағдарламаларды пайдалану, егер бір кодты бағдарламаның әртүрлі орындарына жазу қажет болса, кодтың қайталануын болдырмауға мүмкіндік береді. Бағдарламаға импортталған кітапханалар (мысалы, Sistem) әлдеқашан құрастырған тәртіптерден тұрады.

Әрбір ішкі бағдарлама тек бір тапсырманы шешуі керек,
не есептеуі, не кейбір деректерді көрсетуі, не басқа әрекетті орындауы керек.

Ішкі бағдарламалар немесе әдістер екі түрлі болады - функциялар (жұмыс нәтижесін қайтаратындар) және процедуралар (қайтпайтындар).

Мысал1

Қарапайым мысал жазуды тырысайық. Пайдаланушының кінәсінен кодта қате орын алуы мүмкін (мысалы, ол дұрыс емес деректерді енгізген кезде) экранда «Error» жолын көрсеткіміз келеді делік. Мұны ,былай жасауға болады

```
Console.WriteLine("Error")
```

Ал енді мұндай жолды бағдарламаның көптеген жерлеріне енгізу керек. Әрине, сіз оны барлық жерде жаза аласыз. Бірақ бұл шешімнің екі кемшілігі бар. 1) берілген жол жадта көп рет сақталады; 2) егер қате бойынша шығысты өзгерткіміз келсе, бұл жолды бүкіл бағдарлама бойынша өзгертуге тура келеді, бұл өте ыңғайсыз. Мұндай жағдайлар үшін әдістер мен процедуралар қажет. Процедурасы бар бағдарлама келесідей болуы мүмкін:

```
using System;
class Program {
    static void PrintError() {
        Console.WriteLine("Error");
    }
    static void Main() {
        PrintError();
    }
}
```

Процедура void сөзінен басталады. Процедураның атынан кейін бос жақшалар бар. Процедурада орындалатын барлық мәлімдемелер пробел Жазылады. Әдістер мен процедуралар Main() негізгі әдісінің алдында жазылады. Процедураға сілтеме жасау үшін негізгі бағдарламада оны атымен шақырып, жақшаларды жазуды есте сақтау керек. Бағдарламадағы процедураны кез келген рет шақыруға болады.

Енді пайдаланушының қатесіне байланысты оның қандай қателік жібергеніне байланысты әртүрлі хабарламаларды көрсету керек деп елестетіп көрейік.
Бұл жағдайда әрбір қате үшін өз процедураңызды жаза аласыз

```
void printErrorZero()
{
   Console.WriteLine("Error. Division by zero!");
}


void printErrorInput()
{
   Console.WriteLine("Error in input!");
}
```

А если возможных ошибок будет намного больше? Тогда такое решение нам не подойд
Надо научиться управлять процедурой, указывая ей, какое сообщение на ошибку нужн
Для этого нам понадобятся параметры, которые мы будем записывать в круглых скобк
процедуры

```
void printError(string s)
{
   Console.WriteLine(s);
}
```

В данной процедуре *s* - это параметр - специальная переменная, которая позволяет уп

Параметр — мәні ішкі бағдарламаның жұмысы тәуелді болатын айнымалы. Параметр атаулары ішкі бағдарлама тақырыбында үтірмен бөлінген тізімде берілген. Параметр түрі параметрдің алдында жазылады.

Теперь при вызове процедуры нужно в скобках указывать фактическое значение, которое будет присвоено параметру (переменной s) внутри нашей процедуры

printError("Error! Division by zero!");

Такое значение называется аргументом.

Аргумент - бұл шақырылған кезде ішкі бағдарламаға берілетін параметр мәні. Аргумент тек тұрақты мән емес, айнымалы немесе арифметикалық өрнек болуы мүмкін.

Локальды айнымалының ауқымы – ирек (фигурные скобки) жақшалармен шектелген блокта жарияланған. C# тіліндегі негізгі бағдарлама да ішкі бағдарлама болып табылады, сондықтан void Main() ішінде жарияланған барлық айнымалылар локальды айнымалылар болып табылады.

Бағдарламаның кез келген жерінде (кез келген ішкі бағдарламада) көрінетін айнымалыны жариялау қажет болса, онда мұндай айнымалылар барлық ішкі бағдарламалардан тыс жарияланады.

Мұндай айнымалылар глобальный айнаымалылар деп аталады.

1) Бұл бағдарламада і айнымалысы жергілікті болып табылады. Локальды айнымалы мән ішкі бағдарлама ішінде жарияланады.	2) Мұнда негізгі бағдарламада і айнымалысы бар болса да (7 мәні бар), 5 мәні бар жаңа і локальды айнымалысы құрылады. Бұл бағдарлама орындалғанда экранда 75 мәні пайда болады.	3) Мұнда і глобальный айнымалысы бар. Оның мәні ішкі бағдарлама ішінде де, негізгі бағдарлама ішінде де өзгертілуі мүмкін. Процедура і глобальный айнымалысымен жұмыс істейді және оған 2-ге тең жаңа мән тағайындалады. Экранда 2 мәні көрсетіледі.

static void test()
 {
  int i = 5;
 Console.Write("i");
}
	static void test()
	 { int i = 5;
	  Console.Write("i");
	}
	static void Main()
	 {
	 int i = 7;
	  Console.Write("i");
	test();
	}
		using System;
		class Program
		 {
		 int i;
		 static void test()
		{
		 i = 2;
		 }
		static void Main()
		 {
		 test();
		 Console.Write("i");
		 }
		 }

- Тапсырма: екі айнымалының мәндерін ауыстыратын процедураны жазыңыз. Бұл тапсырманың ерекшелігі шақырушы бағдарламаға белгілі болу үшін процедураға енгізілген өзгерістер қажет.