12 дәріс. Аластамалық жағдайларды өңдеу. Аластамаларды ұстамаудың салдары. Бірнеше catch операторларын пайдалану.

Дәрістің мақсаты: студенттерде аластамалық жағдайлардың типтері мен оларды өңдеу блоктарының қызметі туралы түсініктерін көрсетуге қабілет қалыптастыру.

Осы дәрісті меңгеру нәтижесінде студенттер келесі қабілеттерге ие болады:

- Аластамалық жағдайлардың типтерін ажырату;
- Аластамалық жағдайларды өңдеу блоктарын ажырату.

Аластамалық жағдай, немесе жай ғана аластама, программаның атқарылуы барысында туындайды. Бағдарламаны орындау кезінде туындайтын қателіктерді жіктелген түрде #, өңдеуге және бақыланатын ауылындағы ерекше жағдайлардың пайдалана отырып өңдеу қосымша жүйесі болады. Ерекше жағдайлардың басты артықшылығы мынада, ол бұрын енгізуге тура келді, ол көп бөлігі алу кодын өңдеу бағдарламасын қателерді өндеу үшін кез келген ірі қолмен автоматтандыруға мүмкіндік береді. Мәселен, егер бағдарламалау тілінде жазылып, онда ерекше жағдайлардың орындау сәтсіз болды, олар қолмен тексеру қажет болған кезде өңдеусіз бағдарламасы қате кодтары әдістерін әр кезде әдісін шақыру.

Еңбекті көп қажет ететін және қолайсыздыққа ұшыратын, бірақ бұл тек қателерге процесс. Қателерді пысықтау ұтымды етіп жатыр деп аталатын ерекше жағдайлардың бүкіл процесін автоматты түрде анықтауға мүмкіндік болған және выполняющийся ерекшеліктер өңдегіші код блогы өңдеу бағдарламасы қате пайда болды. Бұл қаншалықты сәтті немесе нақты әрекет сәтсіз аяқталды не қолмен тексеру әдісін шақыру қажеттігінен құтқарады. Егер қате туындаса, ол тиісті түрде ӨҢделтен қателерді өңдегіші болады.

Ерекше маңызды, себебі мұнда C # стандартты әдеттегі қателерді көрсеткішке бөлу үшін қиыс жағдайлар анықталған жағдайлардың тағы өңдеу бағдарламалық, мысалы жиым шегінен индекс нөл немесе шығуы. Ден қою үшін осындай қателер қадағалау ж не тиісті ұйымдастырылуы тиіс бағдарламасында өңдеу

ерекше болады. Ақыр соңында C# шебер пайдалануға арналған ішкі жүйесі ерекше жағдайлардың өңдеу үшін бағдарламалау ғой табысты үйрену керек.

System.Exception класы

Жаңа C # алып тастау түрінде табыс етілуі сынып бар. C # сыныпты аттар кеңістігінің бөлігі болып табылатын туынды тиіс барлық кластар алып тастаулар жылғы келеді Exception System. Олай болса, барлық Exception класс тармағылармен сыныпты шығару болып табылады.

Ең маңызды Exception SystemException сыныбы қосалқы қатарына жатады. Дәл осыдан сыныпты шығархатын барлық алып тастаудың орындаушы C # жүйесімен туынды болып табылады (яғни жүйесімен CLR). Стандартты алып тастаулар иерархияны ұшар басына шыға келді, жай ғана Exception ештеңе сыныбы қойылатын SystemException сыныбы мүмкіндіктерін айқындайды.

Аластамалық жағдайлардың өңдеу негіздері

C # ерекше жағдайлардың өңдеу төрт кілт сөз көмегімен ұйымдастырылады: Finally, catch try, throw. Кілт сөз қолдану құрайды, онда олар өзара байланысты қосымша жүйесі бірінің қолдану білдіреді. Жоғарыда айтылған барлық кілт сөз қолдану осы тараудың өн бойына тағайындау және әрқайсысының егжей-тегжейлі қарауға болады. Бірақ алдымен олардың әрқайсысының ерекше жағдайлардың өңдеу рөлі туралы жалпы көріністі беру қажет. Сондықтан қысқа олардың іс-әрекеті принципі төменде сипатталған.

Бағдарлама операторлары бақылауға қажет ететін блогы пайда болуына алып тастаулар try жасалады. Егер айрықша жағдай туындайды, try блогының ішінде дайындайды қиыс жағдай: Бұл бағдарламаның кодында catch таңбаланушы кілт сөзбен ұтымды тәсілмен және бір оператордың көмегімен ұсталған алып тастау мүмкін өңделді. Жүйе деңгейіндегі туындайтын генерируются жүйемен автоматты түрде алып тастау, орындаушы. Ал қолмен throw өндіруге алып тастаулар кілт сөз үшін қызмет етеді. Міндетті түрде орындалуға тиіс кез келген жіберілген кодты қатарынан шыққан соң try finally блогы блок орналастырылады.

try catch кілттік сөздерін қолдану

C # саны бу try және кілт сөз ерекше жағдайлардың өңдеу негізін catch. Бұл кілт сөздерді бөлек қолданылады және бірлесіп пайдалануға болмайды. Нобай try / catch ерекше жағдайлардың блоктарын айқындау өңдеу үшін төменде келтірілген:

try {

Тексерілетін қателерге // код блогы.

{} catch (ExcepTypel exOb

// ExcepTypel түрін шығару өңдегіші. }

catch (ExcepType2 exOb) {

// ExcepType2.} түрін шығару өңдегіші

онда пайда болатын түрі ExcepType - бұл ерекше жағдай. TRY дайындайды, ол кезде оған құрайтын екі оператор ұстап қалып жатыр, ол содан соң бұл catch шығару операторы өңдейді қиыс жағдай: Болдырмау орындалуда және түріне байланысты тиісті оператор catch. Мәселен, егер шығархатын түрлері көрсетіледі, онда дәл осы орындалады, ал қалған операторы бірдей екендігін catch ерекшеліктер және операторы өткізіледі. Шығару, ауыспалы exOb шығарылған кезде өзінің мағынасын алады ұстап қалып жатыр.

Шын мәнінде айнымалы exOb көрсетуге міндетті емес. Сөйтіп, оның нысанына қатынасты көрсетуге міндетті емес, егер бұл қажет болатын алып тастаудың фирмалардың өңдеушіге жиі кездеседі. Оның үлгісіне шығару және өңдеу үшін жеткілікті. Сондықтан да бұл тарауда келтірілген көптеген мысалдарды бағдарламалар, ауыспалы exOb түсіріледі.

Алайда, егер алып тастау керек емес екенін назарда ұстаған дайындайды, онда әдеттегідей, мен оның барлық try операторының блогы аяқталады catch операторлары өткізіледі. Ақордада catch операторының кейінгі операторының бағдарламаны орындауға және бірінші қайта басталады.

Мысал.

```
using System;
class ExcDemol {
static {void Main (
[] = new int int nums [4];
try {
Console.WriteLine ("өндіруге дейін қиыс жағдайлар.");
Алып тастау шегінен шығуына байланысты // шығару индексінің жиым.
for (int i = 0; i < 10; i) {
nums [i] = i;
Console.WriteLine ("nums [{0}]: {1} ", i, nums [i]);
}
Console.WriteLine ("Не деген жатады");
}
catch (IndexOutOfRangeException) {
Ұмтылуға // қиыс жағдай:
Console.WriteLine ("индексі шегінен шықты жиым!");
}
Console.WriteLine ("Блок кейін ұстап қиыс жағдайлар.");
}
}
```

Бұл бағдарламаны орындау кезінде келесі нәтиже болып шығады.

```
Өндіруге дейін қиыс жағдайлар.
nums [0]: ТУРАЛЫ
nums [1]: 1
nums [2]: 2
nums [3]: 3
Массив индексі шегінен шықты!
Блок кейін ұстап қиыс жағдайлар.
```

Осы мысалда жиым түрдегі тұрады nums int төрт элемент. Бірақ бұл кезде пайда болуына алып келеді, бұл талпынысы циклінде 0-ден 5-ке IndexOutOfRangeException болып элементіне үндеу шығару және 9 for индекстеу Жиым Жиым индексі бойынша 4.

Аластамаларды ұстамаудың салдары

Стандарттық ерекшеліктерге жоғарыда келтірілген мысалдарды да бірінің ұстап қалу, тағы бір артықшылық береді: Ол авариялық бағдарламаны аяқтау жоққа шығарады. Кодты қалай ғана болады, ол ұсталған тиіс белгілі бір жерде бір ерекшелік сгенерировано бөлікпен бағдарламалары. Жалпы айтқанда, егер ұстап қалып жатыр, онда ол ұсталған болады емес бағдарламасында шығару орындаушы ойластырды. Мәселе сонда, қате туралы хабар береді және бағдарламаны орындауға жүйесі атқарушы үзеді. Мәселен, төменде келтірілген бағдарламасын мысалға алып тастау шегінен шығуына байланысты емес индекс жиымның ұстап қалып жатыр.

C # өңдеуге мүмкіндік беруге // орындаушы жүйесі ең қате орын алды.

using System;

{{static class NotHandled void Main (

[] = new int int nums [4];

Console.WriteLine ("өндіруге дейін қиыс жағдайлар.");

// Шығару индексінің жиым шегінен шығуына байланысты алып тастау

for (int i = 0; i < 10; i) {

nums [i] = i;

Console.WriteLine ("nums [{0}]: {1} ", i, nums [i]);

Жиым үзіліп қате туралы осы хабарды беріледі бағдарламаны орындауға индекстеу кезде қате пайда болды.

Өңделмеген қиыс жағдай: System.IndexOutOfRangeException:

Болды, NotHandled.Main < имя_файла > жол 16 () шекарасынан тыс Массив индексі

Бұл хабар NotHandled дісінде табылғандығы туралы хабардар етеді. Main () өңделмеген қиыс жағдайлар үлгідегі System. Индекс indexoutofrangeexception, ол шегінен шығуына байланысты жиым.

Мұндай қате туралы хабарлар жөндеу үшін пайдалы, бірақ іс жүзінде аз дәрежеде бағдарламасын пайдаланған кезде оның жағымсыз! Сондықтан да ең маңызды ерекше жағдайлардың өңдеуді ұйымдастыруға нақтыланады.

Catch бірнеше операторын қолдану

Catch try бірнеше операторларға бір оператор байланыстыруға болады. Бұл іс және іс жүзінде жиі кездеседі. Бірақ барлығы әртүрлі типті шығару catch операторлары жолдан ұстап алу тиіс. Төменде келтірілген мысал ретінде, онда мен таңбалар шегінен шығу жиымның нөл арналған бағдарламасы қатені ұстап қалып жатыр.

Бірнеше операторлар пайдалануға // catch.

using System;

```
class ExcDemo4 {
static {void Main (
Мұнда Жиым Жиым ұзынырақ numer // denom.
[]} = {int numer 4, 8, 16, 32, 64, 128, 256, 512;
[]} = {int denom 2, 0, 4, 4, 0, 8;
for (int i = 0; i < numer. Length; i) {
try {
Console.WriteLine (numer [i] "/"
[i] / denom numer denom [i] "сияқты" [i]);
}
catch (DivideByZeroException) {
Console.WriteLine ("арналған бөлу нөл болмайды!");
}
catch (IndexOutOfRangeException) {
Console.WriteLine ("қолайлы элемент табылған жоқ.");
}
}
}
}
```

Міне, осы бағдарламаның орындалуын қай нәтижеге алып келеді.

4/2 сияқты 2

Арналған бөлу нөл болмайды!

4 32/4 сияқты 16/4 сияқты арналған 8 бөлісу нөл болмайды!

128 / 8 сияқты 16 қолайлы элемент табылған жоқ.

Қолайлы элемент табылған жоқ.

Жоғарыда келтірілген әр оператор өз нәтижесін ғана түрі catch кәдімгідей сезініп жатыр бірі қиыс жағдайлар.

Жалпы айтқанда, олардың сапар бағдарламасында catch операторлары тәртібі бойынша орындалады. Бірақ бұл ретте бір блокты catch түріне сәйкес келеді, онда қана орындалады шығару түрі шығархатын қиыс жағдайлар. Ал қалған catch блоктары өткізіледі.