10 ПОЛИМОРФИЗМ ПРИНЦИПІ

10.1 Полиморфизм ұғымы

Алдынғы бөлімде қарастырылған мұрагерлік ұғымы базалық кластың деректері мен әдістерін қолдануға мүмкіндік береді, бірақ сонымен қатар мұрагерлік екі түрге бөлінеді – статикалық және динамикалық мұрагерлік (әдістерді статикалық және динамикалық байланыстыру).

Статикалық мұрагерлік дегеніміз - барлық байланыстары қосымшаны компиляциялау барысында құрылатын және өзі кластарды сипаттау құрылымдарында жазылатын мұрагерлік.

Динамикалық мұрагерлік және онымен байланысты полиморфизм қасиеті бойынша кейбір байланыстар бағдарламаны орындау процесінде қалыптасады.

Полиморфизм дегеніміз – мұраланатын кластар тізбегіндегі аттас әдістердің сан алуан жолмен орындалу түрлері.

Полиморфизм қасиетін жүзеге асыру арнайы виртуальды әдістер және абстрактлы базалық кластар арқылы орындалады.

Абстрактлы базалық кластар ұғымын қарастырайық.

Кластардты мұралану нәтижесінде артықшылықтарға ие болу үшін ОББ әзірлеушілері базалық кластарды құрай бастады. Базалық кластарда белгілі бір объекттер жиынының деректерін өңдейтін барлық мүмкін әдістері болады, бірақ базалық кластарда әдетте, деректер элементтері болмайды.

Мысалы, «геометриялық фигуралар» базалық класын құрғанда оған аудан мен көлемді табу әдістерін қосуға болады. Әлбетте, егер «нүкте» класы немесе «кесінді» класы туынды кластар болса, онда осы объекттер үшін жоғарыда келтірілген әдістердің еш маңызы болмайды.

Базалық кластарда объекттерді құруға болмайтын болса немесе оның маңызы жойылса, онда ондай кластар абстрактлы базалық кластар деп аталады. Абстрактылы базалық кластар тек қана ұрпақты құру үшін ғана

қолданылады. Әдетте, оларда тек қана әдістер жиынтығы жазылады және осы әдістерді оның әрбір ұрпағы өзінше жүзеге асыратын болады. Абстрактылы базалық кластардың осындай әдістері қолданыста жоқ немесе виртуалды деректер элементтеріне арналған (яғни, мұрагерлік тізбектегі келешекте құрылатын кластардың деректер элементтеріне арналған).

Мұрагерлік тізбек бойынша келешекте құрылатын кластардың қолданыста жоқ, деректердің виртуальды элементтеріне арналған әдістері виртуальды әдістер деп атала бастады.

Виртуальды әдістерді белгілеу үшін C# тілінде virtual арнайы термині қолданылады. Мысалы:

```
virtual public double ploc() {return 0.0;}
```

virtual сөзі ағылшын тілінен аударғанда «нақты» («фактический») дегенді білдіреді. Әдісті виртуальды деп жариялау мынаны анықтайды: осы әдіске бағытталған барлық сілтемелерге рұқсат әдісті шақыру фактісі бойынша беріледі, яғни компиляция кезеңінде емес, қосымшаның орындалу барысында. Бұл механизм әдістерді динамикалық немесе кейінге қалтырып байланыстыру деп аталады.

Компилятор виртуалды әдістерді кездестіргеннен кейін виртуалды әдістер кестесін (Virtual Method Table, VMT) құрайды, онда виртуальды әдістердің атауы және кіру нүктелерінің адрестері жазылады. Әрбір класс үшін бір виртуальды әдістер кестесі құрылады.

Сонымен қатар, қосымшаның орындалуы барысында әрбір құрылған объектке нұсқауыш қосылады, ол нұсқауыш әзірленген VMT кестесіне бағытталады.

Виртуальды әдісті шықыру былай орындалады: объекттен VMT кестесінің адресі алынады, VMT-нен әдістің адресі алынады, ал одан кейін басқару осы әдіске көшеді. Сонымен, барлық аттас әдістердің ішінен виртуальды әдістерді қолданған кезде объект шақырған нақтылы типке сәйкес виртуальды әдіс таңдап алынады.

Егер туынды класс аттас виртуальды әдісті өз нұсқасында орындаған болса, онда класта осы әдіс override атрибутымен бірге жазылады және орнын басу немесе қалқалау (жабу) әдісі болып жарияланады. Мысалы,

```
override public double ploc() { . . . }
```

Әрбір туында класта виртуальды әдісті қайта анықтау міндетті емес. Егер әдіс туынды класта керекті әрекеттерді орындайтын болса, онда әдіс жай ғана мұраланады.

Қосымшаның орындалуы барысында кез келген базалық класс орнын басу әдісін шақырған кезде виртуальды әдістер кестесінде туынды кластың сәйкес әдісіне сілтеме жазылады және бұл әдіс аталық кластың әдепкі бөлігіндей қызмет атқарады.

Орнын басу виртуальды әдісінде базалық кластың аттас әдісінде сияқты параметрлер жиынтығы болуы керек.

Полиморфизм принципі абстрактылы базалық кластың виртуальды әдістерін орнын басу әдістері арқылы «қалқалауға» («перекрытии»)

негізделеді. Сонымен қатар әрбір туынды кластың мұраланған виртуальды немесе орын басу әдістері өзгеше жолмен жүзеге асырылады.

Сонымен бірге полиморфизм қасиеті дегеніміз – әр түрлі мұрагерлік кластардың объекттері үшін базалық кластың аттас виртуальды функциясын қолданған кезде түрлі тәсілмен орындау мүмкіндігі.

Полиморфизм сөзі грек тілінен аударғанда «түрлі тәсілдер» деген мығынаны білдіреді, ал қарастырылып отырған жағдайда - «бір шақыруға — бірнеше әдістер» деген мағынаны қолдануға болады.

Егер әдіс туында кластарда өзгеше орындалуы тиіс болса, онда базалық кластарды сипаттаған кезде осы әдістерді виртуальды әдістер ретінде анықтауға кеңес беріледі. Егер иерархияның барлық кластарында әдіс бірдей орындалатын болса, онда ол әдісті қарапайым әдіс сияқты анықтау керек.

10.2 Әдістерді статикалық мұралану мысалы

Қарапайым геометриялық пішіндердің мұрагерлік кластар тізбегін құрамыз. Базалық класс ретінде өрісі жоқ, ауданды есептеу виртуальды әдісі және басып шығару «таза виртуальды әдісі» бар класты қарастырайық.

Статикалық мұрагерлік мысалы – объекттердің қарапайым құрылуы, өрістерге 0-ден 100-ге дейінгі аралықтағы кездейсоқ мәндерді меншіктеу және оларды экранға басып шығару.

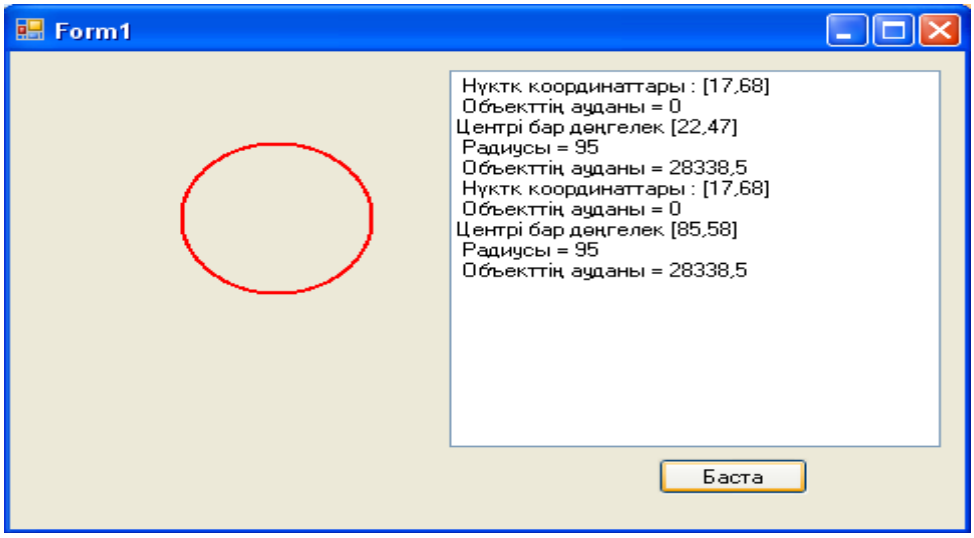
Form1.cs файлының коды:

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
namespace WindowsApplication1
{
 public partial class Form1 : Form
 {
 public int x, y, ra;
 public abstract class baseGeo
 {
  public virtual double ploc() { return 0; }
  public abstract string printO();
 }
 public class tka : baseGeo
 {
  protected int x, y;
  public string ss;
  public tka()
  {
  Random rnd = new Random();
  x = rnd.Next(100);
  y = rnd.Next(100);
  }
```

```
 public int gettkax() { return x; }
 public int gettkay() { return y; }
 public void settka(int xx, int yy)
 {
 x = xx;
 y = yy;
 }
 override public string printO()
 {
 return "[" + x.ToString() + "," + y.ToString() + "]";
 }
}
public class kryg : tka
{
 public kryg()
 {
 Random rnd = new Random(10);
 r = rnd.Next(100);
 }
 public void setkryg(int ax, int ay, int rr)
 {
 settka(ax, ay);
 r = rr;
 }
 public void getkryg(out int ax, out int ay, out int rr)
 {
 ax = x;
 ay = y;
 rr = r;
 }
 public int getkrygr() { return r; }
 override public double ploc()
 // базалық кластың әдісін жабамыз
 {
 return 3.14 * r * r;
 }
 override public string printO()
 // tka класының әдісін жабамыз
 {
 string ss = "";
 ss = "Центрі бар дөңгелек " + base.printO() + "\r\n";
 // tka класының басып шығару функциясы мұраланады
 ss = ss + " Радиусы = " + r.ToString() + "\r\n";
 return ss;
 }
 private int r;
}
public Form1()
{
 InitializeComponent();
}
private void button1_Click(object sender, EventArgs e)
{
```

```
  int i;
  double pl;
  string s;
  Random rnd = new Random(20);
  x = rnd.Next(100);
  y = rnd.Next(100);
  //ra = rnd.Next(100);
  // әдістерді статикалық мұралану мысалы
  tka a = new tka();
  a.settka(x, y);
  s = "Нүкте координаттары : " + a.printO() + "\r\n";
  textBox1.AppendText(s);
  s = " Объекттің ауданы = " + a.ploc().ToString() + "\r\n";
  textBox1.AppendText(s);
  kryg c = new kryg();
  x = c.gettkax();
  y = c.gettkay();
  ra = c.getkrygr();
  //c.setkryg(x, y, ra);
  textBox1.AppendText(c.printO());
  s = " Объекттің ауданы = " + c.ploc().ToString() + "\r\n";
  textBox1.AppendText(s);
  Invalidate();
 }
 private void Form1_Paint_1(object sender, PaintEventArgs e)
 {
  Pen myPen = new Pen(Color.Red, 2);
  Graphics g = e.Graphics;
  g.DrawEllipse(myPen, x, y, ra, ra);
 }
 }
}
```

Бағдарлама жұмысы 10.1-суретінде көрсетілген.



10.1-сурет – Әдістердің статикалық мұралануы

Әдістердің статикалық мұралануы бойынша келтірілген мысалда әдістердің қалқалау механизмі қолданылады, бірақ әдістер арасындағы байланыстар қосымшаның компиляциясы кезінде нақтылы анықталады.

10.3 Әдістердің динамикалық мұралануының мысалы

Әдістердің динамикалық мұралануының мысалы ретінде алдында қарастырылған мұрагерлік кластар тізбегін қолданамыз, бірақ қосымшаның жұмысы кезінде 5 құрылған объектті стекке орналастырамыз.

Form1.cs файлының коды:

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
namespace WindowsApplication1
{
 public partial class Form1 : Form
 {
 public int x, y, ra;
 public int[,] masi = new int[6,3];
 public Stack vstek = new Stack();
 public abstract class baseGeo
 {
 public virtual double ploc() { return 0; }
 public abstract string printO();
 }
 public class tka : baseGeo
 {
 protected int x, y;
 public tka()
 {
 Random rnd = new Random();
 x = (int)Math.Round(rnd.NextDouble() * 100);
 y = (int)Math.Round(rnd.NextDouble() * 100);
 }
 public int gettkax() { return x; }
 public int gettkay() { return y; }
 public void settka(int xx, int yy)
 {
 x = xx;
 y = yy;
 }
 override public string printO()
 {
 return "Нүкте [" + x.ToString() + "," + y.ToString() + "]";
 }
 }
```

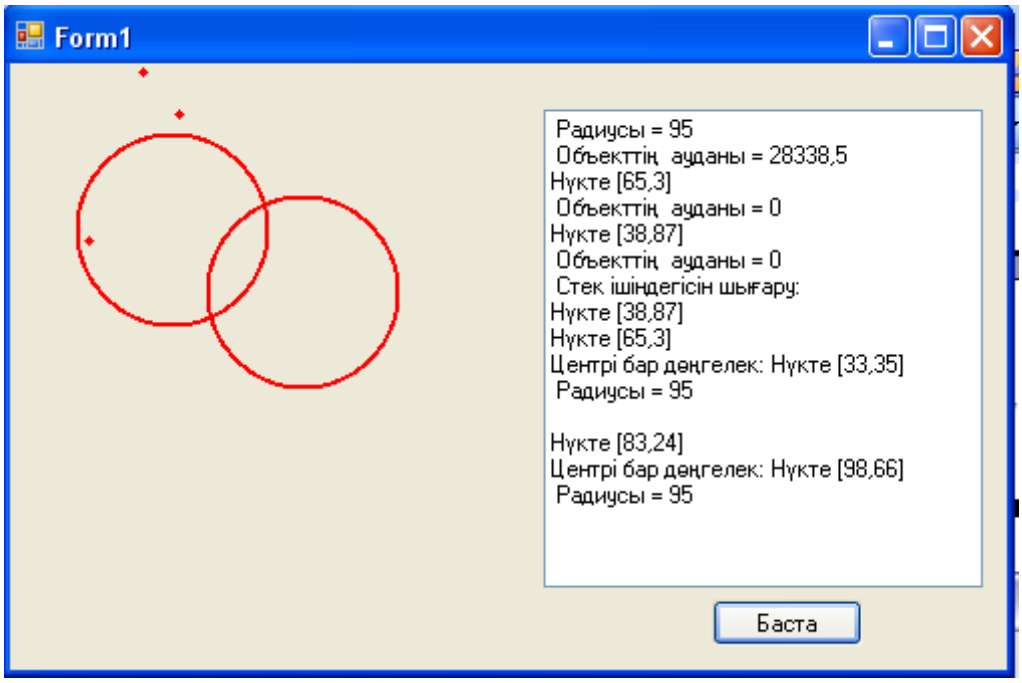
```
public class kryg : tka
{
public kryg()
{
Random rnd = new Random(10);
r = (int)Math.Round(rnd.NextDouble() * 100);
}
public void setkryg(int ax, int ay, int rr)
{
settka(ax, ay);
r = rr;
}
public void getkryg(out int ax, out int ay, out int rr)
{
ax = x;
ay = y;
rr = r;
}
public int getkrygr() { return r; }
override public double ploc()
 // базалық кластың әдісін жабамыз
{
return 3.14 * r * r;
}
override public string print0()
 // tka класының әдісін жабамыз
{
string ss = "";
ss = "Центрі бар дөңгелек: " + base.print0() + "\r\n";
// tka класының басып шығару функциясы мұраланады

ss = ss + " Радиусы = " + r.ToString() + "\r\n";
return ss;
}
private int r;
}
static void vkl(Stack vst, baseGeo n)
{
vst.Push(n);
}
public Form1()
{
InitializeComponent();
}
private void button1_Click(object sender, EventArgs e)
{
int i,k;
double pl;
string s;
Stack vstek = new Stack();
Random rnd = new Random(20);
// әдістерді  динамикалық мұралану мысалы
for (i = 1; i <= 5; i++)
```

```
{
x = (int)Math.Round(rnd.NextDouble() * 100);
y = (int)Math.Round(rnd.NextDouble() * 100);
k = (int)Math.Round(rnd.NextDouble() * 100);
if (k % 2 == 0) k = 0; else k = 1;
if (k == 0)
{
tka a = new tka();
a.settka(x, y);
s = a.printO() + "\r\n";
textBox1.AppendText(s);
s = " Объекттің ауданы = " + a.ploc().ToString() + "\r\n";
textBox1.AppendText(s);
masi[i, 0] = a.gettkax();
masi[i, 1] = a.gettkay();
masi[i, 2] = 0;
vkl(vstek, a);
}
else
{
kryg c = new kryg();
x = c.gettkax();
y = c.gettkay();
ra = c.getkrygr();
textBox1.AppendText(c.printO());
s = " Объекттің ауданы = " + c.ploc().ToString() + "\r\n";
textBox1.AppendText(s);
c.getkryg(out masi[i,0], out masi[i,1], out masi[i,2]);
vkl(vstek, c);
}
}
s = " Стек ішіндегісін шығару: \r\n";
foreach (baseGeo T in vstek)
s = s + T.printO() + "\r\n";
textBox1.AppendText(s);
Invalidate();
}
private void Form1_Paint(object sender, PaintEventArgs e)
{
Pen myPen = new Pen(Color.Red, 2);
Graphics g = e.Graphics;
for (int i = 1; i <= 5; i++)
{
if (masi[i,2]==0)
{
x = masi[i, 0]; y = masi[i, 1];
g.DrawEllipse(myPen, x, y, 2, 2);
}
else
{
x = masi[i, 0]; y = masi[i, 1]; ra = masi[i, 2];
g.DrawEllipse(myPen, x, y, ra, ra);
}
```

```
}
}
}
}
```

Бағдарлама жұмысы 10.2 суретінде көрсетілген.



10.2-сурет – Әдістердің динамикалық мұралануының мысалы

Стек элементтерінің ішіндегісін экранға шығару полиморфизм мысалы болады (әдістердің динамикалық мұралануының). «Статикалық» тұрғыдан стек элементін экранға шығару қарапайым геометриялық пішіндердің элементін экранға шығару болып есептеледі. Бірақ қосымшаның жұмыс істеу процесінде стекке нүкте мен шеңбер кластарының объекттері орналастырылады (әдістердің динамикалық мұралануының мәнісі осында).

10.4 Өзін-өзі тексеру сұрақтары

1 C# тілінде барлық кластардың иерархиялық тізбегінде қандай класс базалық класс болып табылады?

2 Туында класс конструкторын шақырған кезде базалық немесе туында кластардың қандай объектісі ерте (барлығынан бұрын) құрылады?

3 Мұрагерлік кластар тізбегінің артықшылықтары неде?

4 C# тілінде статикалық мұрагерлік нені білдіреді?

5 C# тілінде динамикалық мұрагерлік нені білдіреді?

6 C# тіліндегі полиморфизм ұғымы?

7 Полиморфизм механизмі қалай жүзеге асырылады?

8 Бағдарламаның жұмысы барасында объект анықталған болса, онда оның әдісі қалай аталады?

9 C# тілінде абстрактылы базалық класс деп қандай класс аталады?

10 Виртуальдық әдістер кестесі дегеніміз не?