

Московский Авиационный Институт
(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа 4 по курсу ООП: основы программирования на языке C#

4. НАСЛЕДОВАНИЕ: РАСШИРЕНИЕ, СПЕЦИФИКАЦИЯ, СПЕЦИАЛИЗАЦИЯ, КОНСТРУИРОВАНИЕ, КОМБИНИРОВАНИЕ

Работу выполнил:

М8О-205Б-21 Титеев Р.М.

(подпись)

(вариант)

Руководитель: _____/Кузнецова С.В.

(подпись)

Дата: ____ октября 2022

Расширение, Спецификация, Специализация, Конструирование

Текст программы

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace lab_4_1
8 {
9     public interface A {
10         void mA();
11         int fA();
12     }
13
14     // Specialization + Extension
15     public class B : A {
16         public B() {
17             this.v_1 = 10;
18             this.v_2 = 20;
19         }
20         protected int v_1 { set; get;}
21         public int v_2 { set; get;}
22         public virtual int f() {
23             Console.WriteLine("class B f() ");
24             return 0;
25         }
26         public void mA() {
27             this.v_1 = this.v_1 * this.v_1;
28         }
29         public int fA() {
30             return this.v_2 + 1;
31         }
32     }
33     public class D : B {
34         public D(){
35             this.v_3 = 33;
36         }
37         public int v_3 {set; get;}
38         public override int f() {
39             Console.WriteLine("class D f() ");
40             return base.f() + 10;
41         }
42         public int fD() {
43             return this.v_1 * this.v_3;
44         }
45     }
```

```

45     }
46
47     // Specification
48     public abstract class C : A {
49         public void mC(int a){
50             this.v_1 = a;
51         }
52         public abstract int fC();
53         public void mA() { this.v_1 = this.v_1 * 10; }
54         public int fA() { return this.v_1; }
55         private int v_1 = 22;
56     }
57     public class E : C {
58         public E() { }
59         public override int fC(){
60             Console.WriteLine("fC()");
61             return 0;
62         }
63     }
64     public class J : C {
65         public J() { }
66         public override int fC() {
67             Console.WriteLine("fC()");
68             return 1 + this.fA();
69         }
70     }
71
72     // Construction
73     public class F{
74         public F() {
75             this.v_1 = 1;
76         }
77         protected int v_1 { set; get;}
78         public int f() {
79             return v_1;
80         }
81     }
82     public class K : F {
83         public K() {
84             this.v_2 = 2;
85         }
86         private int v_2 { set; get;}
87         public int f() {
88             return v_2 + v_1;
89         }
90     }
91
92

```

```

93     internal class Program
94     {
95         static void Main(string[] args)
96         {
97
98
99             A a = new B();
100            B b = new B();
101            Console.WriteLine("        Specialization + Extension");
102            Console.WriteLine($"class B b.f() {b.f()}");
103            Console.WriteLine($"class B v_2 {b.v_2}");
104            a.mA();
105            Console.WriteLine($"interface A fA() {a.fA()}");
106
107            b = new D();
108            a = b;
109
110            Console.WriteLine($"class D b.f() {b.f()}");
111            Console.WriteLine($"class D v_2 {b.v_2}");
112            Console.WriteLine($"class D v_3 {((D)b).v_3}");
113            Console.WriteLine($"class D fD() {((D)b).fD()}");
114
115
116            Console.WriteLine("        Specification");
117            C c = null;
118            c = new E();
119            Console.WriteLine("class C c.fC() {0}", c.fC());
120
121            c = new J();
122            Console.WriteLine("class C c.fC() {0}", c.fC());
123
124
125            Console.WriteLine("        Construction");
126            F f = new F();
127            Console.WriteLine($"class F f.f() {f.f()}");
128            f = new K();
129            Console.WriteLine($"class K f.f() {((K)f).f()}");
130
131            Console.ReadKey();
132        }
133    }
134 }

```

Результат работы

```
D:\Projects\laboratory\3_sem  x + -
Специализация + расширение
class B f()
class B b.f() 0
class B v_2 20
interface A fA() 21
class D f()
class B f()
class D b.f() 10
class D v_2 20
class D v_3 33
class D fd() 330
    Спецификация
fc()
class c c.fc() 0
fc()
class c c.fc() 23
    Конструирование
class F f.f() 1
class K f.f() 3
```

Комбинирование

Текст программы

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace lab_4_2
8 {
9     public interface A{
10         void mA();
11         int fA();
12     }
13
14     public interface B{
15         int fB();
16         void mB();
17     }
18
19     public class C{
20         public C() { this.v_1 = 33; }
21         public int v_1 { set; get; }
22         public int f()
23         {
24             Console.WriteLine("class C f() ");
25             return 1;
26         }
27     }
28
29     public class D : C, A, B{
30         public D() { this.v_2 = 1; this.v_3 = 2; }
31         protected int v_2 { set; get; }
32         public int v_3 { set; get; }
33
34         public void mA() { this.v_2 = this.v_3 + this.v_1; }
35         public int fA() { return this.v_3 * 10; }
36
37         public int fB() { return this.v_3 * (10-this.v_1); }
38         public void mB() { this.v_2 = this.v_1*this.v_3+100; }
39     }
40     internal class Program{
41         static void Main(string[] args)
42         {
43             A a = null;
44             a = new D();
```

```

45         a.mA();
46         Console.WriteLine("      Combination");
47         Console.WriteLine($"a.fa() = {a.fA()}");
48         Console.WriteLine($"((D)a).f() =  {(D)a).f()}");
49
50         C c = new C();
51         c.f();
52         c = new D();
53
54         Console.ReadKey();
55
56     }
57 }
58 }

```

Результат работы

```

D:\Projects\laboratory\3_sem  x  +  v
Комбинирование
a.fa() = 20
class C f()
((A)b).f() = 1
class C f()

```

Вывод

При специализации дочерний класс является более конкретным, частным или специализированным случаем родительского класса.

Расширение дает возможность добавить новые функциональные возможности к родительскому классу, но не меняет наследуемое поведение.

При спецификации родительский класс описывает поведение, которое реализуется в дочернем классе, но оставлено нереализованным в родительском.

Во время конструирования дочерний класс использует методы, предопределяемые родительским классом.

Комбинирование позволяет объединить черты нескольких классов(интерфейсов) в одном (в данном случае трех).