

Московский Авиационный Институт
(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа 2 по курсу ООП: основы программирования на языке C#

2. АГРЕГАЦИЯ ПО ЗНАЧЕНИЮ И ВЛОЖЕНИЕМ

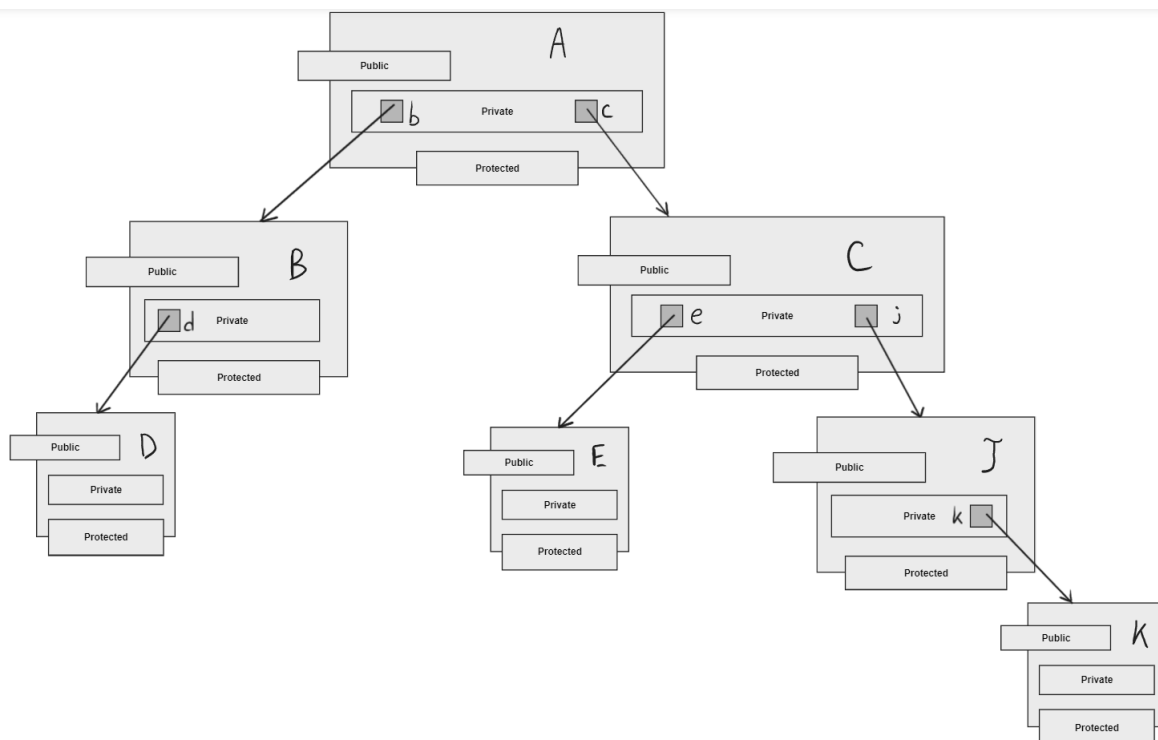
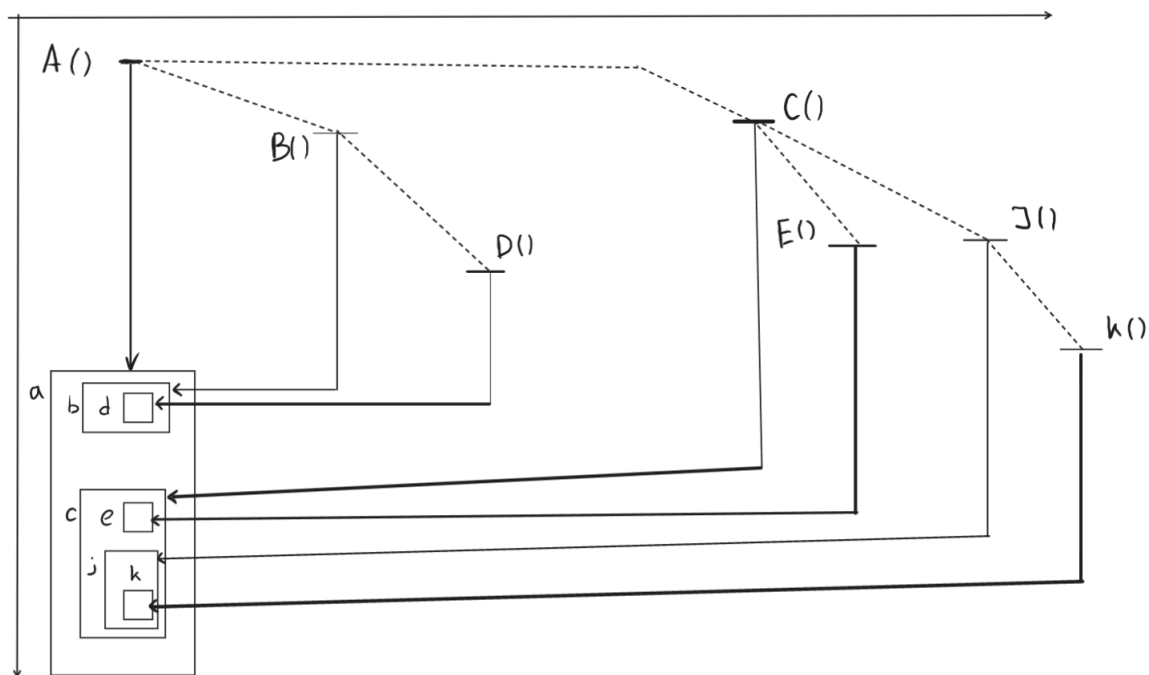
Работу выполнил:

М8О-205Б-21 Титеев Р.М. _____
(подпись) (вариант)

Руководитель: _____/Кузнецова С.В.
(подпись)

Дата: ____ октября 2022

Агрегация по значению



Текст программы

```

1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;

```

```

7 namespace lab_2_1
8 {
9     class A
10    {
11        private B b = new B();
12        private C c = new C();
13        public A() { }
14        public void mA()
15        {
16            Console.WriteLine("method of A");
17        }
18        public B bA
19        {
20            get { Console.Write("get b ->"); return b; }
21        }
22        public C cA
23        {
24            get { Console.Write("get c ->"); return c; }
25        }
26    }
27
28    class B
29    {
30        private D d = new D();
31        public B() { }
32        public void mB()
33        {
34            Console.WriteLine(" method of B");
35        }
36        public D dA
37        {
38            get { Console.Write("get d ->"); return d; }
39        }
40    }
41
42    class C
43    {
44        private J j = new J();
45        private E e = new E();
46        public C()
47        {
48            this.c_val = 0;
49        }
50        public void mC()
51        {
52            Console.WriteLine(" method of C");
53        }
54        public E eA

```

```

55     {
56         get { Console.Write("get e ->"); return e; }
57     }
58     public J jA
59     {
60         get { Console.Write("get j ->"); return j; }
61     }
62     public int c_val { set; get; }
63 }
64
65 class D
66 {
67     public D() { }
68     public void mD()
69     {
70         Console.WriteLine(" method of D");
71     }
72 }
73
74 class E
75 {
76     private D d = new D();
77     public E() { }
78     public void mE()
79     {
80         Console.WriteLine(" method of E");
81     }
82
83     public D dA
84     {
85         get { Console.Write("get d ->"); return d; }
86     }
87 }
88
89 class J
90 {
91     private K k = new K();
92     public J() { }
93     public void mJ()
94     {
95         Console.WriteLine(" method of J");
96     }
97
98     public K kA
99     {
100         get { Console.Write("get k ->"); return k; }
101     }
102 }

```

```

103     class K
104     {
105         public K() { }
106         public void mK()
107         {
108             Console.WriteLine(" method of K");
109         }
110     }
111     internal class Program
112     {
113         static void Main(string[] args)
114         {
115             A a = new A();
116             a.mA();
117             a.bA.mB();
118             a.cA.mC();
119
120             a.bA.dA.mD();
121             a.cA.jA.mJ();
122             a.cA.eA.mE();
123
124             a.cA.jA.kA.mK();
125
126             Console.WriteLine($"a.cA.c_val: {a.cA.c_val}");
127             a.cA.c_val = 15;
128             Console.WriteLine($"a.cA.c_val: {a.cA.c_val}");
129
130             Console.ReadKey();
131         }
132     }
133 }

```

Результат работы

```
D:\Projects\laboratory\3_sem x + -
method of A
get b -> method of B
get c -> method of C
get b ->get d -> method of D
get c ->get j -> method of J
get c ->get e -> method of E
get c ->get j ->get k -> method of K
get c ->a.cA.c_val: 0
get c ->get c ->a.cA.c_val: 15
```

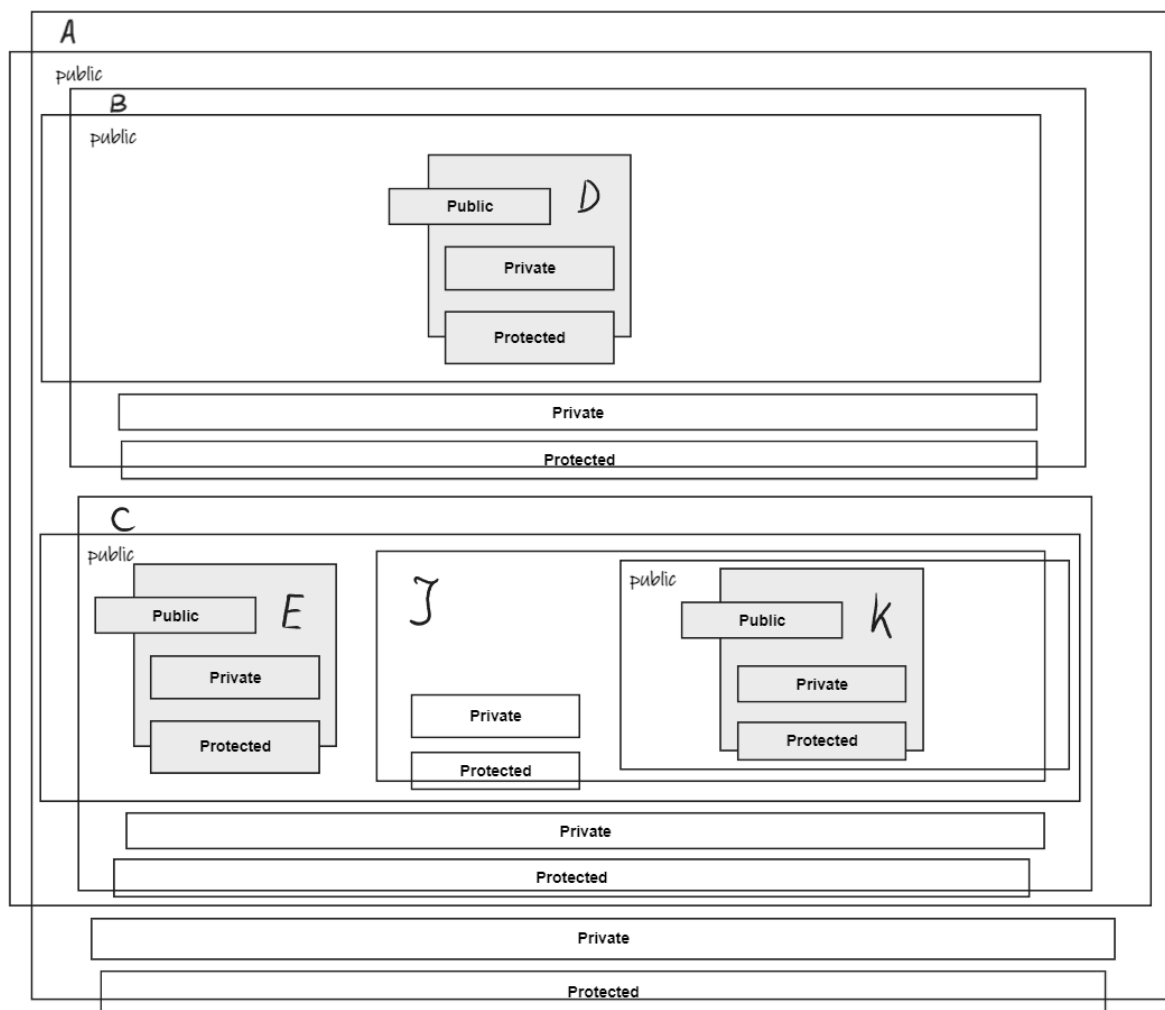
Пример использования

Используя данный метод можно описать устройство компьютера. Так, например, при инициализации самого компьютера создадутся все нужные компоненты (материнская плата, процессор, видеокарта и т.д.). Однако в самой программе мы можем отдельно создать компонент этого компьютера. Например можно создать отдельно процессор или любую другую часть, и при этом не запускать инициализацию самого компьютера.

Вывод

При агрегации по значению, все объекты создаваемого класса существуют внутри него самого, уничтожение их отдельно от объекта, частью которого они являются невозможно. Их уничтожение происходит при уничтожении самого верхнего в иерархии объекта. Например: *b*, *c* – части *a*, они создаются в конструкторе *a*, а их уничтожение происходит при вызове деструктора *a*.

Агрегация вложением



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_2_2
8 {
9     class A
10    {
11        public A() { }
12        public class B
13        {
14            public B() { }
15            public class D
16            {
17                public D() { }
18            }
19        }
20    }
21 }

```

```

18         public void mD()
19         {
20             Console.WriteLine(" method of D");
21         }
22     }
23     public void mB()
24     {
25         Console.WriteLine(" method of B");
26     }
27     public D dA
28     {
29         get { Console.Write("get d ->"); return d; }
30     }
31     private D d = new D();
32 }
33
34 public class C
35 {
36     public C()
37     {
38         this.c_val = 0;
39     }
40     public class E
41     {
42         public E() { }
43         public void mE()
44         {
45             Console.WriteLine(" method of E");
46         }
47     }
48     public class J
49     {
50         public J() { }
51         public class K
52         {
53             public K() { }
54             public void mK()
55             {
56                 Console.WriteLine(" method of K");
57             }
58         }
59         public void mJ()
60         {
61             Console.WriteLine(" method of J");
62         }
63         public K kA
64         {
65             get { Console.Write("get k ->"); return k; }

```



```

66         }
67         private K k = new K();
68     }
69     public void mC()
70     {
71         Console.WriteLine(" method of C");
72     }
73     public E eA
74     {
75         get { Console.Write("get e ->"); return e; }
76     }
77     public J jA
78     {
79         get { Console.Write("get j ->"); return j; }
80     }
81     public int c_val { set; get; }
82     private J j = new J();
83     private E e = new E();
84 }
85
86 public void mA()
87 {
88     Console.WriteLine("method of A");
89 }
90 public B bA
91 {
92     get { Console.Write("get b ->"); return b; }
93 }
94 public C cA
95 {
96     get { Console.Write("get c ->"); return c; }
97 }
98 private B b = new B();
99 private C c = new C();
100 }
101
102 internal class Program
103 {
104     static void Main(string[] args)
105     {
106         A a = new A();
107
108         Console.WriteLine($"a.cA.c_val: {a.cA.c_val}");
109         a.cA.c_val = 15;
110         Console.WriteLine($"a.cA.c_val: {a.cA.c_val}");
111
112         a.mA();
113     }

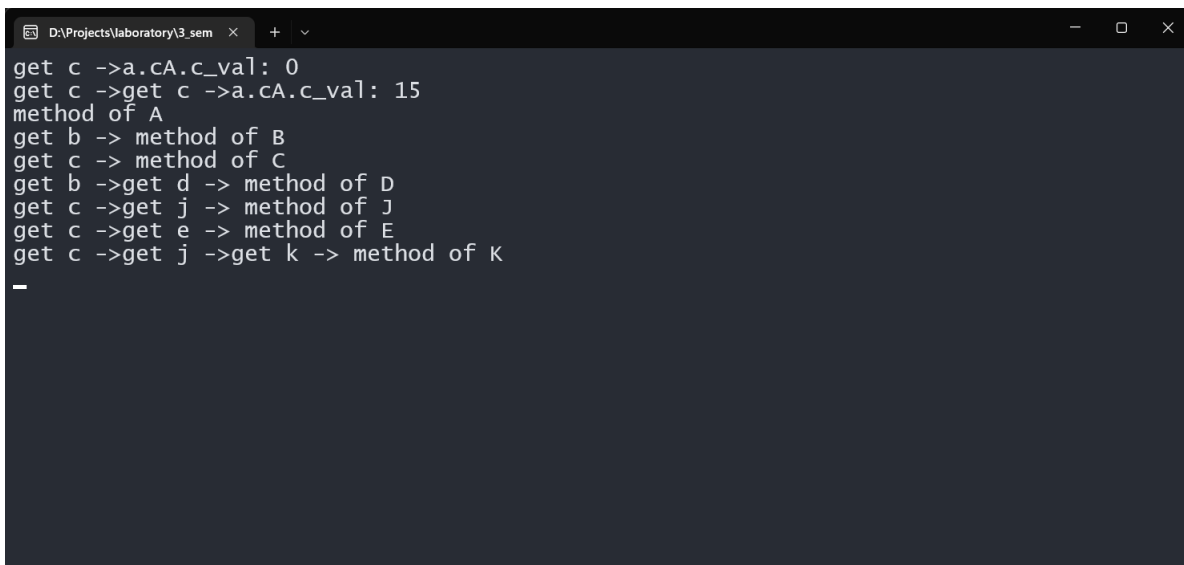
```

```

114         a.bA.mB();
115         a.cA.mC();
116
117         a.bA.dA.mD();
118         a.cA.jA.mJ();
119         a.cA.eA.mE();
120
121         a.cA.jA.kA.mK();
122
123         Console.ReadKey();
124
125
126     }
127 }
128 }

```

Результат работы



```

D:\Projects\laboratory\3_sem  x  +  -  x
get c ->a.cA.c_val: 0
get c ->get c ->a.cA.c_val: 15
method of A
get b -> method of B
get c -> method of C
get b ->get d -> method of D
get c ->get j -> method of J
get c ->get e -> method of E
get c ->get j ->get k -> method of K
-

```

Пример использования

Данный метод организации классов может быть использован для описание какого-нибудь игрового персонажа со сложной структурой тела. Так, например, главным в этой иерархии будет сам персонаж, а части его тела будут создаваться внутри этого класса. Таким образом мы можем изменять характеристики персонажа через главный класс, и при этом части тела нельзя будет создать отдельно в программе, неописав самого персонажа.

Вывод

При агрегации вложением, объявление классов происходит внутри классов, стоящих выше их по иерархии. Все объекты создаваемого класса существуют внутри него самого, уничтожение их отдельно от объекта, частью которого они являются невозможно. Их уничтожение происходит при уничтожение самого верхнего в иерархии объекта. Например: *b*, *c* – части *a*, они

создаются в конструкторе a , а их уничтожение происходит при вызове деструктора a .