

A Praction

ctice () Compete



Rank

Leaderboard

Try the Next Challenge | Try a Random Challenge





Dashboard > C++ > Inheritance > Accessing Inherited Functions

Points: 475 Rank: 6579

Your Accessing Inherited Functions submission got 30.00 points.

Share

Twee

×

Accessing Inherited Functions ■



Problem Submissions Leaderboard Discussions

You are given three classes A, B and C. All three classes implement their own version of func.

In class A, func multiplies the value passed as a parameter by 2:

```
class A
    public:
        A(){
            callA = 0;
    private:
        int callA;
        void inc(){
            callA++;
    protected:
        void func(int & a)
            a = a * 2;
            inc();
    public:
        int getA(){
            return callA;
};
```

In class B, func multiplies the value passed as a parameter by 3:

```
class B
{
    public:
        B(){
            callB = 0;
      }
    private:
      int callB;
      void inc(){
            callB++;
      }
    protected:
      void func(int & a)
      {
            a = a * 3;
            inc();
      }
    public:
      int getB(){
            return callB;
```

```
};
```

In class C, func multiplies the value passed as a parameter by 5:

```
class C
  {
      public:
          C(){
               callC = 0;
          }
      private:
           int callC;
          void inc(){
              callC++;
          }
      protected:
          void func(int & a)
               a = a * 5;
               inc();
      public:
          int getC(){
               return callC;
  };
You are given a class D:
  class D
  {
          int val;
          public:
                   //Initially val is 1
                    D()
                    {
                           val = 1;
                    }
                    //Implement this function
                    void update_val(int new_val)
                    //For Checking Purpose
                    void check(int); //Do not delete this line.
  };
```

You need to modify the class *D* and implement the function **update_val** which sets *D*'s *val* to *new_val* by manipulating the value by only calling the *func* defined in classes *A*, *B* and *C*.

It is guaranteed that $\textit{new_val}$ has only 2,3 and 5 as its prime factors.

Input Format

Implement class D's function update_val. This function should update D's val only by calling A, B and C's func.

Constraints

 $1 \le new_val \le 10000$

Note: The new_val only has ${f 2,3}$ and ${f 5}$ as its prime factors.

Sample Input

 $new_val = 30$

Sample Output

A's func will be called once.

B's func will be called once.

C's func will be called once.

```
Explanation
```

```
Initially, val = 1.

A's func is called once:

val = val*2
val = 2

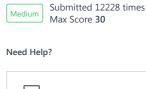
B's func is called once:

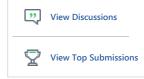
val = val*3
```

C's func is called once:

val = 6

```
val = val*5
val = 30
```





RATE THIS CHALLENGE



Download problem statement

Download sample test cases

∠ Suggest Edits

```
f ⊮ in
```

```
Current Buffer (saved locally, editable) \ \mathscr{V} \ \mathfrak{O}
                                                                                               C++14
                                                                                                                                    \Diamond
 1 ▼ #include<iostream>
 3 using namespace std;
 4
 5
    class A
 6 ▼ {
         public:
 7
 8 ▼
             A(){
                   callA = 0;
 9
              }
10
11
         private:
              int callA;
12
13 ▼
              void inc(){
14
                   callA++;
              }
15
16
          protected:
17
18
              void func(int & a)
19 ▼
20
                   a = a * 2;
```

```
21
                inc();
            }
22
23
        public:
24 ▼
            int getA(){
25
                return callA;
26
27
   };
28
29 class B
30 ▼ {
31
        public:
32 ▼
            B(){
33
                callB = 0;
            }
34
35
        private:
            int callB;
36
37 ▼
            void inc(){
38
                callB++;
39
            }
40
        protected:
            void func(int & a)
41
42 ▼
            {
                a = a * 3;
43
44
                inc();
            }
45
46
        public:
47 ▼
            int getB(){
                return callB;
48
49
50 };
51
52 class C
53 ₹ {
        public:
54
55 ₹
            C(){
56
                callC = 0;
57
            }
58
        private:
            int callC;
59
60 ▼
            void inc(){
61
                callC++;
62
            }
63
        protected:
            void func(int & a)
64
65 ▼
            {
66
                a = a * 5;
67
                inc();
            }
68
        public:
69
70 ▼
            int getC(){
                return callC;
71
72
73 };
74 class D: private A,B,C
75 ▼ {
76 private:
77
         int val;
78
    public:
79
        D(): val(1){}//Initially val is 1
80
81
         void update_val(int new_val)
82 ▼
             while(new_val % 2 == 0)
83
84 ▼
85
                 new_val /= 2;
86
                 A::func(val);
87
             }
88
             while(new_val % 3 == 0)
89 ▼
90
                 new val /= 3;
91
                 B::func(val);
92
             }
             while(new_val % 5 == 0)
93
94 ▼
             {
95
                 new_val /= 5;
96
                 C::func(val);
```

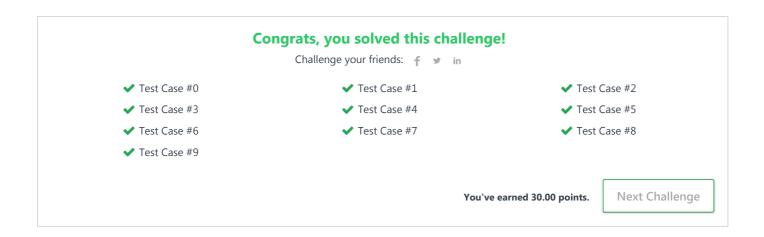
```
98
 99
           void check(int);
100
      };
101
102
103
     void D::check(int new_val)
104 ▼ {
105
           update_val(new_val);
      cout << "Value = " << val << endl << "A's func called " << getA() << " times " << endl << "B's func called "
<< getB() << " times" << endl << "C's func called " << getC() << " times" << endl;</pre>
106
107
108
109
110 int main()
111 ▼ {
           D d;
112
           int new_val;
113
114
           cin >> new_val;
           d.check(new_val);
115
116
117
      }
                                                                                                                                    Line: 100 Col: 3
```

Test against custom input

1080

Run Code

Submit Code



Contest Calendar|Blog|Scoring|Environment|FAQ|About Us|Support|Careers|Terms Of Service|Privacy Policy|Request a Feature