Seminar 7

Pentru fiecare dintre secventele de cod de mai jos, spuneti daca secventa compileaza sau nu. In caz afirmativ, spuneti ce va afisa (in cazul in care standardul C++ spune ca e comportament nedefinit, puteti specifica acest lucru). In caz negativ, sugerati o modificare, prin editarea a cel mult o linie de cod (modificarea unei linii de cod, adaugarea unei linii de cod sau stergerea unei linii de cod), care sa faca secventa sa compileze si spuneti ce afiseaza noua secventa de cod.

```cpp
// 1
#include <iostream>
using namespace std;
class A {
public:
    void foo () {cout << "A::foo" << endl;}
};

class B: public A {
public:
    void foo () {cout << "B::foo" << endl;}
};

void bar (A& a) {
    B *pb = dynamic_cast<B*>(&a);
    pb->foo();
}

int main () {
    B b;
    bar(b);
    return 0;
}
```

```
main.cpp:15:13: error: 'A' is not polymorphic
B *pb = dynamic_cast<B*>(&a);
          ^                    ~~
```

```cpp
// 2
#include <iostream>
using namespace std;

class C {
    const int i;
public:
    C (int j = 2022) { this->i = j; }
    operator int () { return this ->i; }
};

int main () {
    C c1(5), c2;
    cout << c1 << c2 << endl;
    return 0;
}
```

```
main.cpp:8:5: error: constructor for 'C' must explicitly initialize the const member
    'i'
C (int j = 2022) { this->i = j; }
^
```

```cpp
// 3
#include <iostream>
using namespace std;

class C {
public:
    C () {cout << "C"; }
    C (const C& c) { cout << "cpy-C"; }
    ~C () { cout << "~C";}
    static C getObject () {
        return C();
    }
};

void foo (C c) {}

int main () {
    foo(C::getObject());
    return 0;
}
```

```
C~C or Ccpy-C~Ccpy-C~C~C
```

```cpp
// 4
#include <iostream>
using namespace std;

class C {
    int x;
public:
    C (int y = 0) : x(y) {}
    const C operator+ (const C& c) {
        return C(this->x + c.x);
    }
    friend ostream& operator << (ostream& out, const C& c) {
        return out << c.x;
    }
};

int main () {
    C c1(2022), c2(05), c3(16);
    cout << c1 + c2 + c3;
    return 0;
}
```

```
main.cpp:18:21: error: invalid operands to binary expression ('const C' and 'C')
    cout << c1 + c2 + c3;
            ~~~~~~~ ^ ~~
```

```cpp
// 5
#include <iostream>
using namespace std;

class B {
    B () { cout << "B"; }
public:
    B (int x) { cout << "B(int)"; }
};

class D : B {
public:
    D () { cout << "D"; }
};

int main () {
    D d;
    return 0;
}
```

```
main.cpp:12:5: error: base class 'B' has private default constructor
    D () { cout << "D"; }
    ^
```

```
// 6
#include <iostream>
using namespace std;

class B {
public:
    virtual void foo () { cout << "B::foo"; }
    ~B () {cout << "~B"; }
};

class D : public B {
public:
    void foo () { cout << "D::foo"; }
    ~D () { cout << "~D"; }
};

int main () {
    B *b = new D();
    b->foo();
    delete b;
    return 0;
}
```

```
D::foo~B
```

```
// 7
#include <iostream>
using namespace std;

class A {
public:
    ~A () {cout << "~A";}
};
class B {
public:
    ~B () {cout << "~B";}
};
class C: virtual public A, public B {
public:
    ~C () {cout << "~C";}
};
class D: virtual public A, public B {
public:
    ~D () {cout << "~D";}
};
class E: public C, public D {
public:
    ~E () {cout << "~E";}
};

int main () {
    E e;
    return 0;
}
```

```
~E~D~B~C~B~A
```

```
// 8
#include <iostream>
using namespace std;

class C {
    int *i;
public:
    C (int& x) : i(&x) { }
    void set (int x) { *(this->i) = x;}
};

int main () {
    int i = 2022;
    C c(i);
    c.set(5);
    cout << i;
    return 0;
}
```

```
5
```

```cpp
// 9
#include <iostream>
using namespace std;

template <class T> void foo (T a, T b) {
    T aux  = a;
    a = b;
    b = aux;
}

template <> void foo (int a, int b) {
    cout << "swap<int>" << endl;
    int aux = a;
    a = b;
    b = aux;
}

int main () {
    int i = 2022, j = 5;
    foo(i, j);
    cout << i << " " << j << endl;
    return 0;
}
```

```
swap<int>
2022 5
```

```cpp
// 10
#include <iostream>
using namespace std;

class C {
public:
    C (int i) { cout << "C" << i;}
};

int main () {
    C *p = new C[100];
    return 0;
}
```

```
main.cpp:11:16: error: no matching constructor for initialization of 'C[100]'
    C *p = new C[100];
               ^
```

```cpp
// 11
#include <iostream>
using namespace std;

class B {
public:
    virtual void bar () {cout << "B::bar";}
};
class D: B {
    friend void foo (D d) {
        B *b = &d;
        b->bar();
    }
public:
    void bar () {cout << "D::bar";}
};

int main () {
    D ob;
    foo(ob);
    return 0;
}
```

```
D::bar
```

```cpp
// 12
#include <iostream>
using namespace std;

class B {
protected:
    static int count;
public:
    B() {count++;}
    static void display () {
        cout << count;
    }
};

class D: public B {
public:
    void triple () {
        this->count *= 3;
    }
};

int B::count = 0;

int main () {
    D vd[] = {D(), D(), D(), D(), D()};
    vd[2].triple();
    B::display();
    return 0;
}
```

```
15
```