1. Spuneți dacă programul de mai jos este corect. În caz afirmativ, spuneți ce afișează, în caz negativ propuneți o (singură) modificare prin care programul devine corect.

```
#include<iostream>
using namespace std;
class A
{
    static int x;
    const int y;
public:
    A(int i, int j) : x(i), y(j) {}
    static int f(int z, int v) { return x + z + v; }
};

int main ()
{
    A ob(1, 2);
    cout << ob.f(3, 2);
return 0;
}
```

2. Spuneți dacă programul de mai jos este corect. În caz afirmativ, spuneți ce afișează, în caz negativ propuneți o (singură) modificare prin care programul devine corect.

```
#include <iostream>
using namespace std;
class B
{ protected: static int x; int i;
  public: B() {x++; i=1; }
    ~B() {x--;}
    static int get_x() {return x;}
    int get_i() {return i;}
};
int B::x;
class D: public B {
    public: D() {x++;}
    ~D() {x--;}};
    int f(B *q) { return (q->get_x()) + 10; }
int main() {
    B *p = new B[10];
    cout<<f(p);
    delete[] p;
    p=new D;
    cout<<f(p);
    delete p;
    cout<<D::get_x();
return 0;}
```

3. Spuneți dacă programul de mai jos este corect. În caz afirmativ, spuneți ce afișează, în caz negativ propuneți o (singură) modificare prin care programul devine corect.

```
#include<iostream>
using namespace std;
class B {
protected:
    int x;
public:
    B(int i = 1) { x=i; }
    virtual B f(B ob) {return x+ob.x+1; }
    void afisare() {cout<<x;}
};
class D: public B {
public:
    D(int i = -2) : B(i) {}
    B f(B ob) {return x+ob.x-1;}
};
int main() {
    B *p1 = new D;
    B *p2 = new B;
    B *p3 = new B(p1->f(*p2));
    p3->afisare();
return 0;
}
```

4. Descrieti pe scurt constructorii de copiere.

5. Descrieti pe scurt comportamentul functiilor virtuale in constructori si in destructori.

6. Spuneți dacă programul de mai jos este corect. În caz afirmativ spuneți ce afișează, în caz negativ spuneți de ce nu compileaza.

```
#include<iostream>
using namespace std;
class A {
public:
    virtual int f() const { cout<<"A::f()\n"; return 1; }
    virtual void f(string) const {cout<<"A::f(string)\n"; }
    virtual void g() const {cout<<"A::g()\n";}
};
class B: public A {
public:
    void g() const {cout<<"B::g()\n";}
};
class C: public A {
public:
    int f() const {cout<<"C::f()\n"; return 2;}
};
int main() {
```

```
    string s("CTI");
    B ob1;
    int x = ob1.f();
    ob1.f(s);
    C ob2;
    x = ob2.f();
return 0;
}
```

7. Spuneți dacă programul de mai jos este corect. În caz afirmativ, spuneți ce afișează, în caz negativ propuneți o (singură) modificare prin care programul devine corect.

```
#include<iostream>
using namespace std;
struct X {
int i;
public: X(int ii=5)
    {i = ii; cout<<i<<"";}
const int tipareste(int j)
    {cout<<i<<"";<<return i+j;}};
int main(){
    X O(7);
    O.tipareste(5);
    X&O2=O;
    O2.tipareste(6);
    X* p =&O;
    cout<<p->tipareste(7);
return 0;}
```

8. Spuneți dacă programul de mai jos este corect. În caz afirmativ, spuneți ce afișează, în caz negativ propuneți o (singură) modificare prin care programul devine corect.

```
#include<iostream>
using namespace std;
class B{ protected: int x;
public:
    B(int i=0):x(i){}
    operator int(){return x;}
};
class D: public B{
public:
    D(int i=0):B(i){}
    operator float(){return x;}
};
int main(){D ob(12);
    try{throw ob;}
    catch(float a){cout<<"A";}
    catch(B b){cout<<"B";}
    catch(D d){cout<<"C";}
```

```
    catch(...){cout<<"D";}
    return 0;}
```

9. Spuneți dacă programul de mai jos este corect. În caz afirmativ, spuneți ce afișează, în caz negativ propuneți o (singură) modificare prin care programul devine corect.

```
#include<iostream>
using namespace std;
class A
{   int x;
public:
    A(int i=0){x=i;}
    A operator+(const A& a}{return x+a.x;}
    template<class T>ostream& operator<<(ostream&);
};
template<classT>
ostream&A::operator<<(ostream& o){o<<x;return o;}
int main(){
    A a1(1),a2(2);
    cout<<a1+a2;
    return 0;}
```

10. Spuneți dacă programul de mai jos este corect. În caz afirmativ, spuneți ce afișează, în caz negativ propuneți o (singură) modificare prin care programul devine corect.

```
#include<iostream>
#include<vector>
using namespace std;
class Test{int i;
public:
    Test(int x=0):i(x){cout<<"C ";};
    Test(const Test& x)(i=x.i;cout<<"CC";}
    ~Test(){cout<<"D"}};
int main(){
    vector<Test>v;
    v.push_back(1);
    Test ob(3);
    v.push_back(ob);
    Test& ob2 = ob;
    v.push_back(ob2);
return 0;}
```

11. Spuneți dacă programul de mai jos este corect. În caz afirmativ, spuneți ce afișează, în caz negativ propuneți o (singură) modificare prin care programul devine corect.

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

```
class cls{int i;
public: cls(){cout<<"2";}}A;
class cls2: public cls
{public: cls2(){cout<<"1";}}B;
class cls3:public cls
{public:cls3(){cout<<"4";}}C;
class cls4; public cls3, public cls2
{cls c;
public: cls4(){cout<<"3";}};
int main()
{cls4 ob;
return 0;
}
```

12. Spuneți dacă programul de mai jos este corect. În caz afirmativ, spuneți ce afișează, în caz negativ propuneți o (singură) modificare prin care programul devine corect.

```
class B { int i;
public:
    B() {i = 1;}
    virtual int get_i() { return i; }
};
class D: virtual public B {int j;
public:
    D() {j = 2;}
    int get_i() { return B::get_i() + j; }
};
class D2: virtual public B {int j2;
public:
    D2() {j2 = 9;}
    int get_i() { return B::get_i() + j2; }
};
class MM: public D, public D2 {int x;
public:
    MM() {x = D::get_i()+D2::get_i();}
    int get_i() { return x; }
};
int main () {
    B *o = new MM();
    cout << o->get_i() << "\n";
    MM *p = dynamic_cast<MM*>(o);
    if(p) cout << p->get_i() << "\n";
    D *p2 = dynamic_cast<D*>(o);
    if(p2) cout << p2->get_i() << "\n";
    return 0;
}
```

13. Spuneți dacă programul de mai jos este corect. În caz afirmativ, spuneți ce afișează, în caz negativ propuneți o (singură) modificare prin care programul devine corect.

```
#include <iostream>
using namespace std;
class cls {
    int i;
public:
    cls(int j):i(j){}
    int f(int* x) { x[1] = 10; return *x;}};
int main()
{
    cls a(3);
    int x[10];
    for(int i=0;i<10;i++) x[i] = 10-i;
    cout << a.f(x)[2] << " " << x[3];
    return 0;
}
```

14. Descrieti, pe scurt, proprietatile metodelor statice.

15. Exemplificati, pe scurt, conceptul de iterator.