

Examen scris

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

```
#include<iostream.h>
class A
{
    protected: int x;
    public: A(int i):x(i){ }
           int get_x() { return x; } };
class B: public A
{
    public: B(int i):A(i) {}
           B operator+(const B& b) {return x+b.x; } };
int main()
{
    B a(23), b(-15);
    cout<<a+b;
    return 0;
}
```

- II. Descrieți pe scurt cum este implementat mecanismul de control al tipului în timpul execuției – RTTI.

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

```
#include<iostream.h>
class A
{
    int x;
    public: A(int i=25) { x=i; }
           friend int& f(const A&); };
int& f(const A& c) { return c.x; }
int main()
{
    A a(15);
    cout<<f(a);
    return 0;
}
```

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

```
#include<iostream.h>
class A
{
    int *v,dim;
    public: A(int i) { dim=i; v=new int[i];
                for(int j=1; j<dim; j++) v[j]=j; }
                int size() { return dim; }
                friend int& operator[](A, int) };
int& operator[](A a, int i) { return *(a.v+a.size()-i); }
int main()
{ A a(10);
  a[3]=7;
  for (int i=0; i<a.size(); i++) cout<<a[i];
  return 0;
}
```



- V. Descrieți pe scurt diferența dintre o clasă și un obiect.



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

```
#include<iostream.h>
class A
{
    int x;
    public: A(int i):x(i){}
                int get_x() const { return x; } };
class B: public A
{
    int *y;
    public: B(int i):A(i){ y=new int[i];
                for(int j=0; j<i; j++) y[j]=1; }
                B(B&);
                int& operator[](int i) { return y[i]; } };
B::B(B& a)
{ y=new int[a.get_x()];
  for(int i=0;i<a.get_x();i++) y[i]=a[i];
}
ostream& operator<<(ostream& o, B a)
{ for(int i=0;i<a.get_x();i++) o<<a[i];
  return o;
}
int main()
{ B b(5);
  cout<<b;
  return 0;
}
```



Σ

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

```
#include<iostream.h>
class A
{
    static int x;
    public: A(int i=1) { x=i; }
           int f(A a) { return x+a.x; }
           static int g() { return f(2)/2; } };

int A::x=7;
int main()
{
    cout<<A::g();
    return 0;
}
```



- VIII. Descrieți pe scurt crearea dinamică de obiecte.



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

```
#include<iostream.h>
class A
{
    protected: int x;
    public: A(int i=12) { x=i; }
           int get_x() { return x; } };

class B: public A
{
    public: B(int i):A(i) {}
           B(const A& a) {return a.x; } };

int main()
{
    A a(9);
    B b(a);
    cout<<b.get_x();
    return 0;
}
```



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

```
#include<iostream.h>
class A
{
    protected: int x;
    public: A(int i=-16) { x=i; }
           virtual A f(A a) { return x+a.x; }
           void afisare(){ cout<<x; } };
class B: public A
{
    public: B(int i=3):A(i) {}
           B f(B b) { return x+b.x+1; } };
int main()
{ A *p1=new B, *p2=new A, *p3=new A(p1->f(*p2));
  p3->afisare();
  return 0;
}
```



- XI. Descrieți pe scurt proprietățile unui câmp static al unei clase.

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

```
#include<iostream.h>
class A
{
    public: int x;
           A(int i=0) { x=i; }
           A operator+(const A& a) { return A(x+a.x); } };
ostream& operator<<(ostream& o, A a) { o<<a.x; return o; }
template <class T>
class B
{
    T y;
    public: B() {}
           B(T i) { y=i; }
           B operator+(B ob) { return B(ob.y+1); }
           void afisare(){ cout<<y; } };
int main()
{ B<int> b1(-15); B<A> b2(1);
  (b1+b2).afisare();
  return 0;
}
```



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

```
#include<iostream.h>
class A
{ public: int x;
    A(int i=13) { x=i; } };
class B: virtual public A
{ public: B(int i=15) { x=i; } };
class C: virtual public A
{ public: C(int i=17) { x=i; } };
class D: public A
{ public: D(int i=19) { x=i; } };
class E: public B, public C, public D
{ public: int y;
    E(int i,int j):D(i),B(j){ y=x+i+j; }
    E(E& e) { y=-e.y; } };

int main()
{ E e1(-9,3), e2=e1;
  cout<<e2.y;
  return 0;
}
```

- XIV. Descrieți pe scurt funcțiile șablon și dați exemplu de 3 situații în care un apel de funcție NU generează o versiune a funcției dintr-un șablon disponibil pentru funcția respectivă.

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

```
#include<iostream.h>
class A
{    const int x, y;
    public: A(int i, int j):x(i), y(j) { }
    int f() { return x+y; } };

int main()
{ const A a(5,6);
  cout<<a.f();
  return 0;
}
```

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

```
#include<iostream.h>
class A
{
    public: int x;
           A(int i=0) { x=i; }
           virtual A minus() { return(1-x); } };
class B: public A
{
    public: B(int i=0) { x=i; }
           void afisare() { cout<<x; } };
int main()
{
    A *p1=new B(27);
    *p1=p1->minus();
    p1->afisare();
    return 0;
}
```

XVII. Descrieți pe scurt diferența dintre polimorfismul de compilare și cel de execuție.

XVIII. Spuneți dacă programul de mai jos este corect. În caz afirmativ, spuneți ce afișează pentru o valoare întreagă citită egală cu 23, în caz negativ spuneți de ce nu este corect.

```
#include <iostream.h>
int f(int y)
{ if (y%2) throw y+2.5;
  return y/2;
}
int main()
{ int x;
  try
  {
    cout<<"Da-mi un numar: ";
    cin>>x;
    if (x%3) x=f(x);
    else throw x;
    cout<<"Numarul "<<x<<" e bun!"<<endl;
  }
  catch (int i)
  { cout<<"Numarul "<<i<<" nu e bun!"<<endl;
  }
  return 0;
}
```