Polimorfizam - Primjeri ispisa

1. Detaljno objasniti rezultat izvrsavanja narednih programa:

a.)

```
#include <iostream>

class A {
  public:
  virtual void f() { std::cout << "A"; }
};

class B : public A {
  private:
  void f() { std::cout << "B"; }
};

void g(A& a) { a.f(); }

int main() {
  B b;
  g(b);
}
```

Program ce ispisati B jer klasa B radi redefiniciju metoda f, a metod f je virtuelni metod klase a koju klasa b naslijedjuje.

b.)

```
#include <iostream>

struct A {
  A() { foo(); }
  virtual ~A() { foo(); }
  virtual void foo() { std::cout << "1"; }
  void bar() { foo(); }
};

struct B : public A {
  virtual void foo() { std::cout << "2"; }
};
```

```
int main() {
  B b;
  b.bar();
}
```

Program ce prvo ispisati broj 1 zbog konstuktora super klase(strukture) A, zatim ce ispisati broj 2 jer jer je metod A::foo() oznacen kao virtual, a B implementira metod foo() i na kraju ce ispisati broj 1 zbog ~A() koji poziva svoju virtuelnu metodu foo() koja je implementirana za strukturu A.

c.)

```
#include <iostream>

struct X {
  virtual void f() const { std::cout << "X"; }
};

struct Y : public X {
  void f() const { std::cout << "Y"; }
};

void print(const X& x) { x.f(); }

int main() {
  X arr[1];
  Y y1;
  arr[0] = y1;
  print(y1);
  print(arr[0]);
}
```

Program ce prvo ispisati Y jer struktura Y implementira virutelni metod f strukture X od koje naslijedjuje, zatim ce ispisati X jer je arr tipa X i elementi u nizu su tipa X pa se poziva X::f().

d.)

```
#include <iostream>

struct A {
  virtual void foo(int a = 1) { std::cout << "A" << a;
}
```

```
};

struct B : A {
  virtual void foo(int a = 2) { std::cout << "B" << a;
}
};

int main() {
  A* b = new B;
  b->foo();
}
```