

## Analiza **SOLID** principa

- **Single Responsibility Principle** – Program sadrži veliki broj klasa koje izvršavaju samo jedan zadatak. Svaka klasa sadrži jednostavne metode, izvršava samo određeni tip akcija svojstven za tu klasu. Možemo zaključiti da je **S** princip ispoštovan.
- **Open/Closed Principle** – Klase su dizajnirane tako da svaka klasa koja koristi neku drugu klasu ne treba biti modificirana pri uvođenju novih funkcionalnosti. Metode kontejnerskih klasa, one koje vrše dodavanje u kontejnersku klasu, ne predstavljaju problem i ne narušavaju **O** princip.
- **Liskov Substitution Principle** – Program posjeduje apstraktnu klasu “*Korisnik*”, koju nasljeđuju klase “*Student*”, “*Profesor*”, “*Demonstrator*” i “*Asistent*”. Ne možemo reći da je ispoštovan **L** princip, ali ne možemo reći ni da nije ispoštovan, jer u našem dizajnu nigdje ne koristimo osnovni objekat (klasu “*Korisnik*”) već nam ona služi samo za nasljeđivanje.
- **Interface Segregation Principle** – Program nije implementirao nijedan interfejs, tako da je razmatranje poštivanja **I** principa nemoguće.
- **Dependency Inversion Principle** – Program je ispoštovao **D** princip, jer bazna klasa (“*Korisnik*”) iz koje ostale klase (“*Profesor*”, “*Asistent*”, “*Demonstrator*”, “*Student*”) vrše nasljeđivanje je apstraktna.