Univerzitet u Sarajevu
Elektrotehnički fakultet u Sarajevu
Odsjek za računarstvo i informatiku

SOLID Principi
Imunizacija '21

Objektno orijentisana analiza i dizajn

Naziv grupe: Schpritzer
Članovi: Muhamed Borovac
Eldar Čivgin
Dženan Nuhić
Benjamin Pašić

Single Responsibility Principle - Princip pojedinačne odgovornosti

Ovaj princip glasi: *Klasa bi trebala imati samo jedan razlog za promjenu.*
U našem dijagramu klasa, ovaj princip je ispunjen jer sve klase upravljaju isključivo nad svojim atributima.
Na primjer, klasa *CovidTest* sadrži osnovne informacije o jednom testu, poput tip testa, datum testiranja i sl.. Ova klasa ima metode koje vrše operacije nad istim atributima.
Također, klasa *CovidKarton* ima atribute koji su liste klasa *CovidTest*, *Bolest*, *Vakcinacija*. Na ovaj način je S princip ispunjen jer u suprotnom bi klasa *CovidKarton* vršila operacija nad atributima koji nisu njena odgovornost.

Open Closed Principle - Otvoreno zatvoreni princip

Ovaj princip glasi: *Entiteti softvera (klase, moduli, funkcije) trebali bi biti otvoreni za nadogradnju, ali zatvoreni za modifikacije.*
Ovaj princip je ispunjen jer u našem dijagramu klasa, klase su povezane vezama agregacije i kompozicije, jer u klasama kao atribute često imamo objekte drugih klasa tako da promjena u jednoj klasi neće značiti promjenu u ostalim klasama. Na primjer, ako bismo u klasi *Korisnik* dodali novu vrstu zanimanja, ne bismo imali razloga za mijenjanje klase *Korisnik*.

Liskov Substitution Principle - Liskov princip zamjene

Ovaj princip glasi: *Podtipovi moraju biti zamjenjivi njihovim osnovnim tipovima.*
Naš dijagram klasa posjeduje više apstraktnih klasa. Jedna od njih je klasa Zahtjev *iz koje su izvedene* ZahtjevZaTestiranje *i* ZahtjevZaVakcinaciju.
Klasa *StrucnaOsoba* kao atribut ima kolekciju apstraktne klase *Zahtjev*, te metode za dodavanje i brisanje zahtjeva. Kada dodajemo neki zahtjev, umjesto apstraktne klase Zahtjev, možemo proslijediti i klase naslijeđene iz nje (*ZahtjevZaTestiranje* i *ZahtjevZaVakcinaciju*).
Sa svime navedenim, smatramo da je ovaj princip ispunjen.

Interface Segregation Principle - Princip izoliranja interfejsa

Ovaj princip glasi: *Klijenti ne treba da ovise o metodama koje neće upotrebljavati.*
Ovaj princip je zadovoljen iz razloga što u našem sistemu ne postoji nijedan interfejs. Također, u našem sistemu nije prisutna nijedna "debela" klasa.

Dependency Inversion Principle - Princip inverzije ovisnosti

Ovaj princip glasi:
a) Moduli visokog nivoa ne bi trebali ovisiti od modula niskog nivoa, oba bi trebalo da ovise od apstrakcija,
b) Moduli ne bi trebali ovisiti od detalja. Detalji bi trebali biti ovisni od apstrakcija.
Na osnovu samih zahtjeva ovog principa, odnosno da ne treba ovisiti od konkretnih klasa, te prilikom naslijeđivanja treba razmatrati slučajeve da je osnovna klasa apstraktna, mi smo ovo primijenili u klasama *StrucnaOsoba*, *CovidKarton* gdje kao atribute koristimo apstraktne klase.