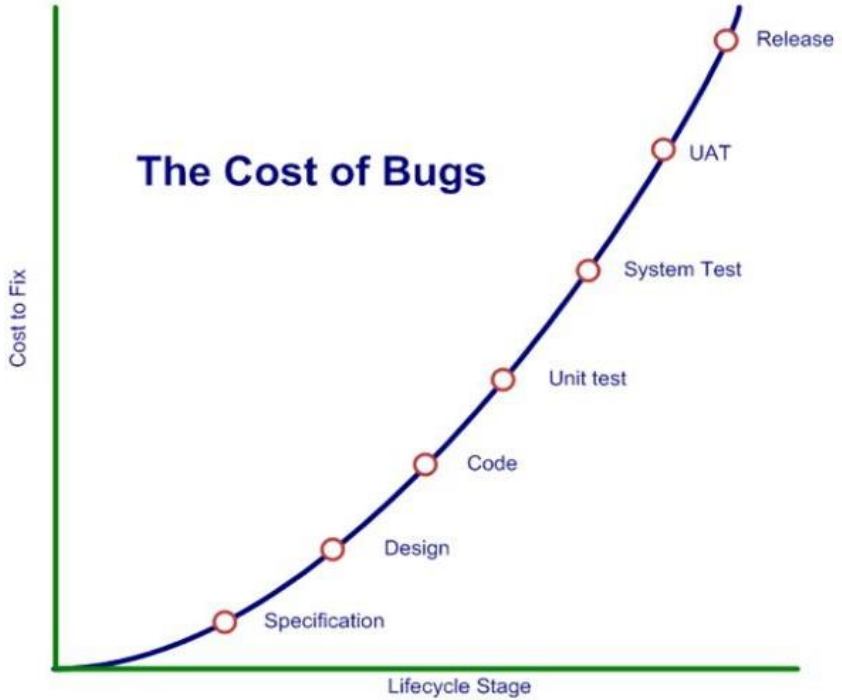
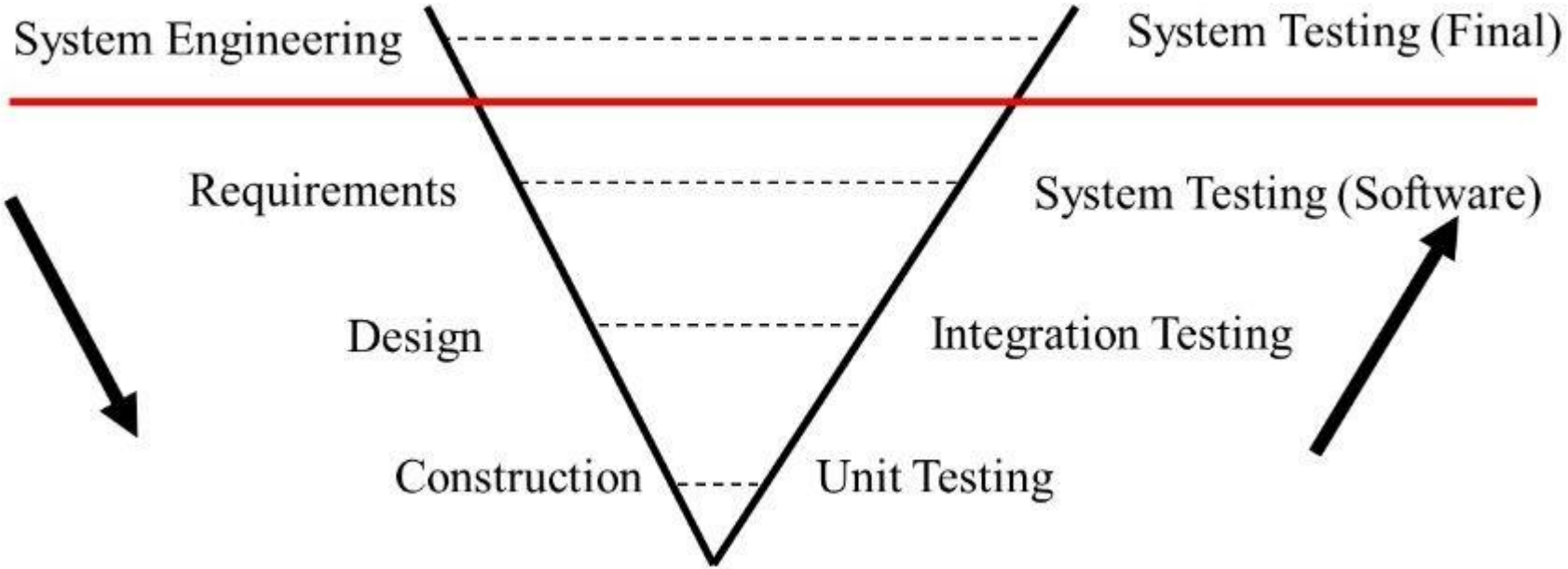
Junit testiranje

NUnit, NSubstitute, OpenCover

Čemu testiranje?



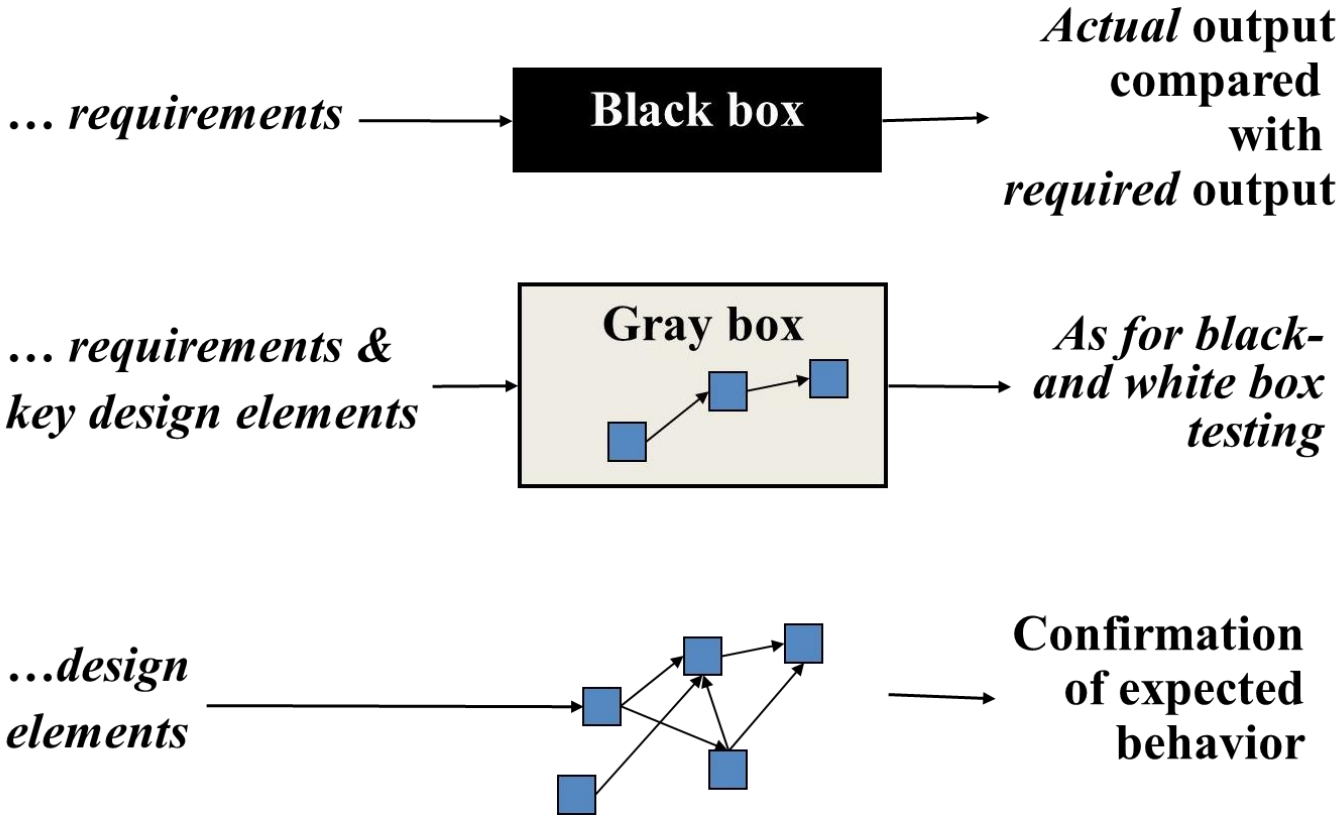
Strategije pri testiranju



Dobiti testiranja

- Testovi smanjuju broj bagova u novim funkcionalnostima
- Testovi smanjuju broj bagova u postojećim funkcionalnostima
- Testovi predstavljaju dobru dokumentaciju
- Testovi smanjuju cenu pravljenja izmena
- Testovi unapređuju arhitekturu rešenja
- Testovi olakšavaju refaktorisanje
- Testovi definišu funkcionalnosti
- Testovi predstavljaju odbranu od drugih programera
- Testovi vas teraju da razmišljate pre pisanja koda
- Testovi ubrzavaju razvoj

Tipovi testiranja

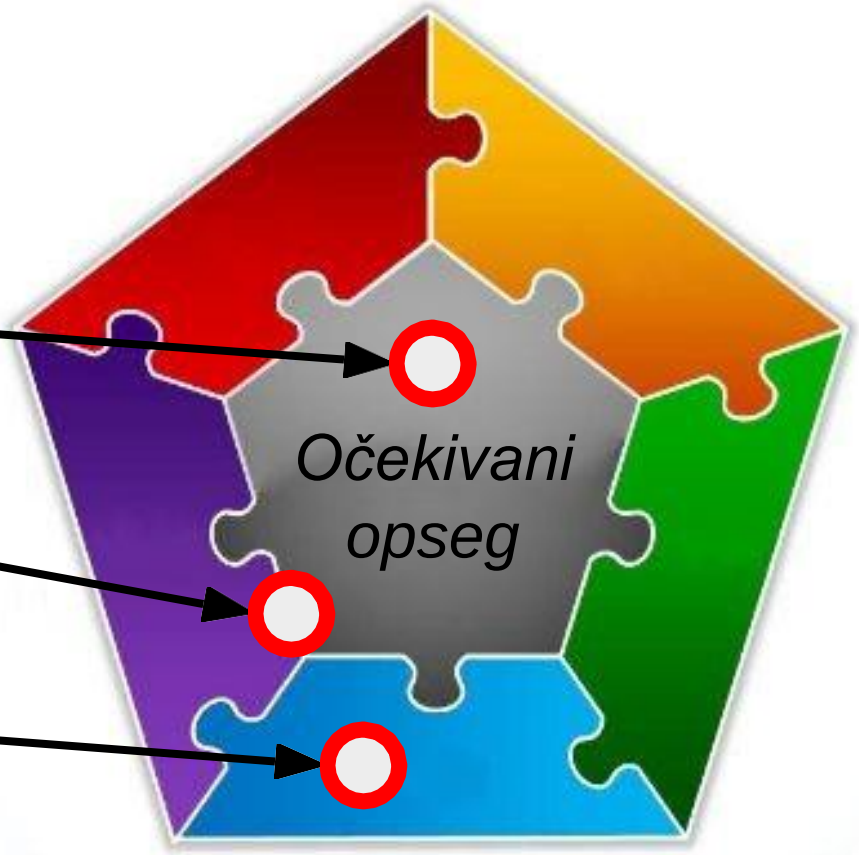


Lociranje problema

U opsegu

Granični slučaj

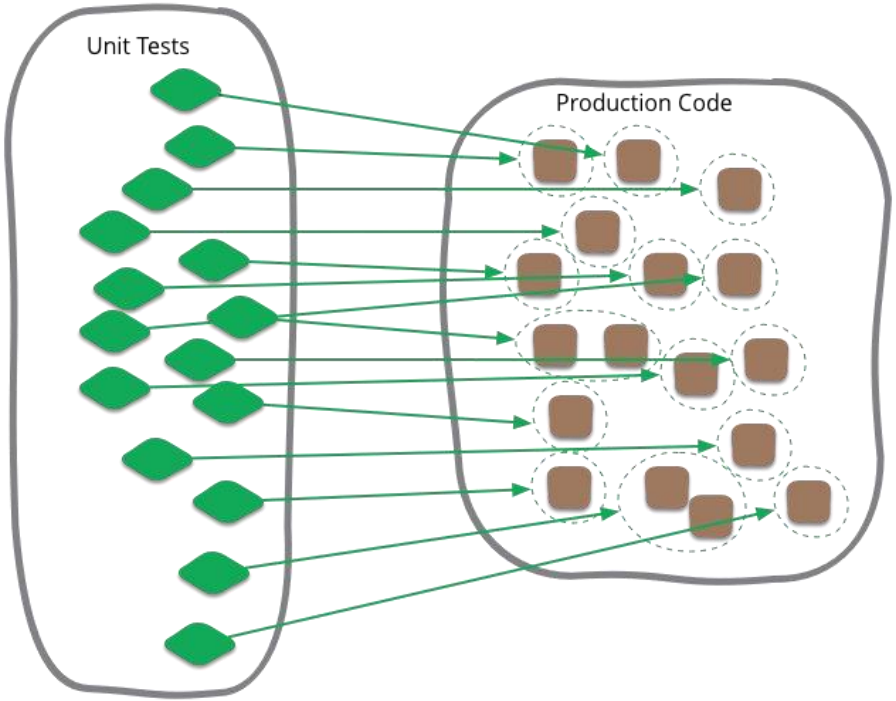
Van opsega



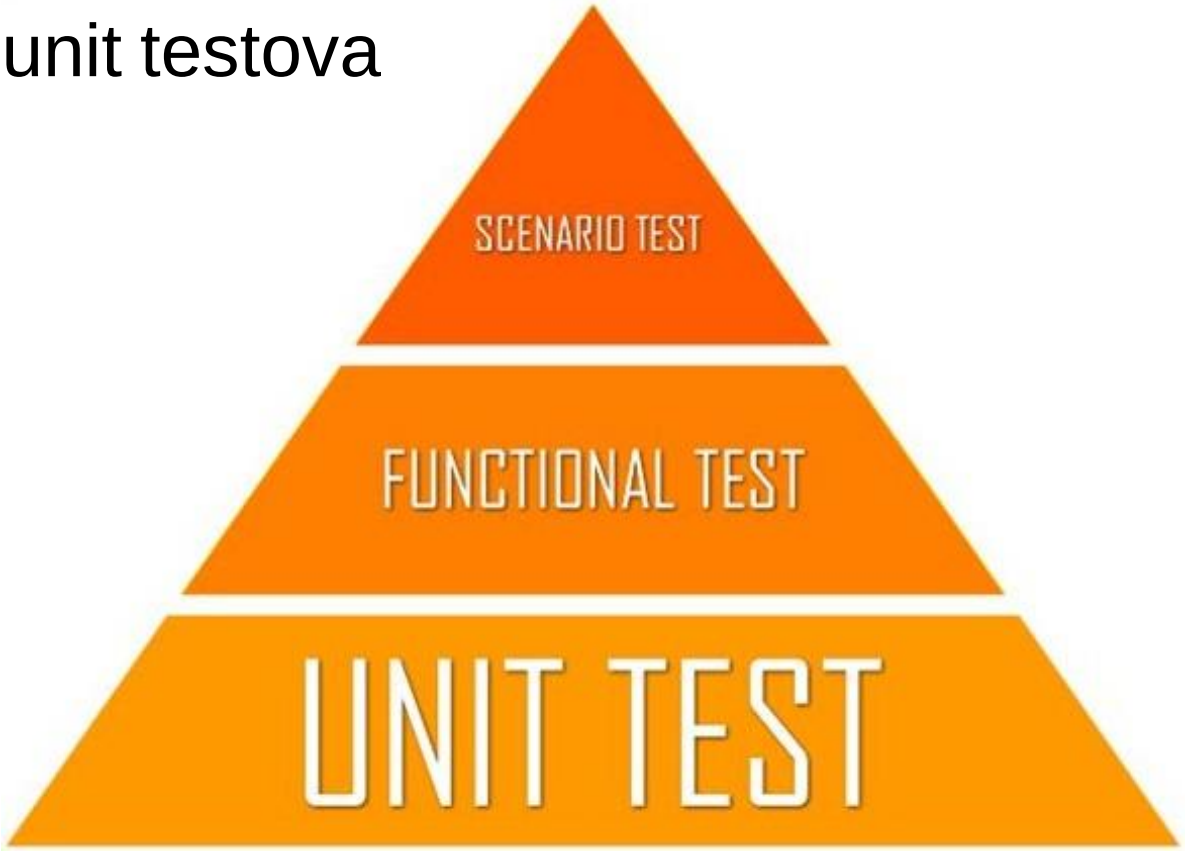
Šta je junit testiranje?

- Junit testiranje predstavlja razvojni proces u kojem se najmanje jedinice aplikacije, koje se mogu testirati, pojedinačno i nezavisno detaljno proveravaju u zavisnosti od željenog ponašanja.
- Junit testovi proveravaju ponašanje jedinice, koja se testira, samo u odnosu na slučajeve od interesa. Na taj način, programeri se ohrabruju da prave izmene u kodu, bez bojazi kako će se te izmene odraziti na rad ostalih jedinica ili na rad celog programa.
- Kada se jednom utvrde svi junit testovi u programu, tako da rade na efikasan i pravilan način (bez grešaka), može se preći na proveravanje većih komponenti.

Junit testiranje i izolovani delovi aplikacije



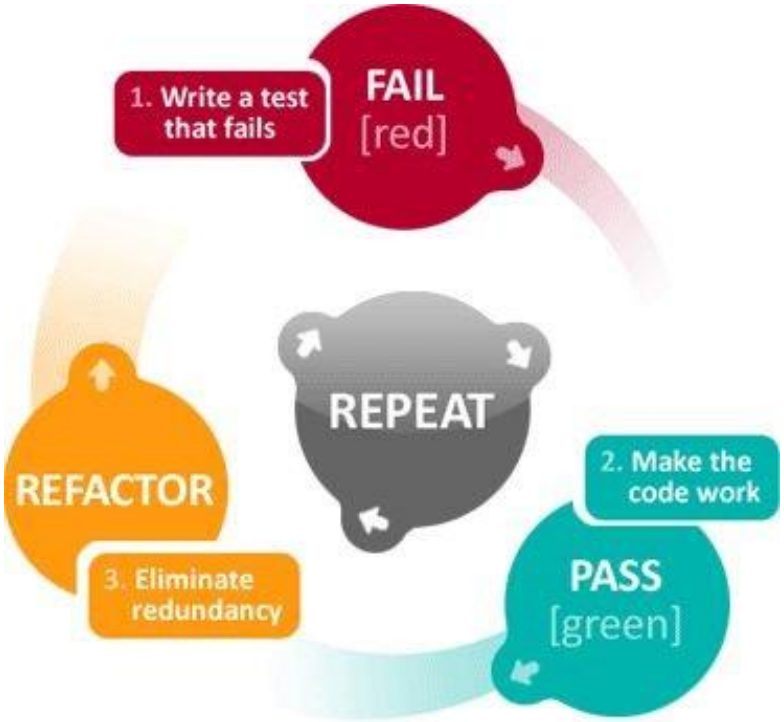
Važnost junit testova

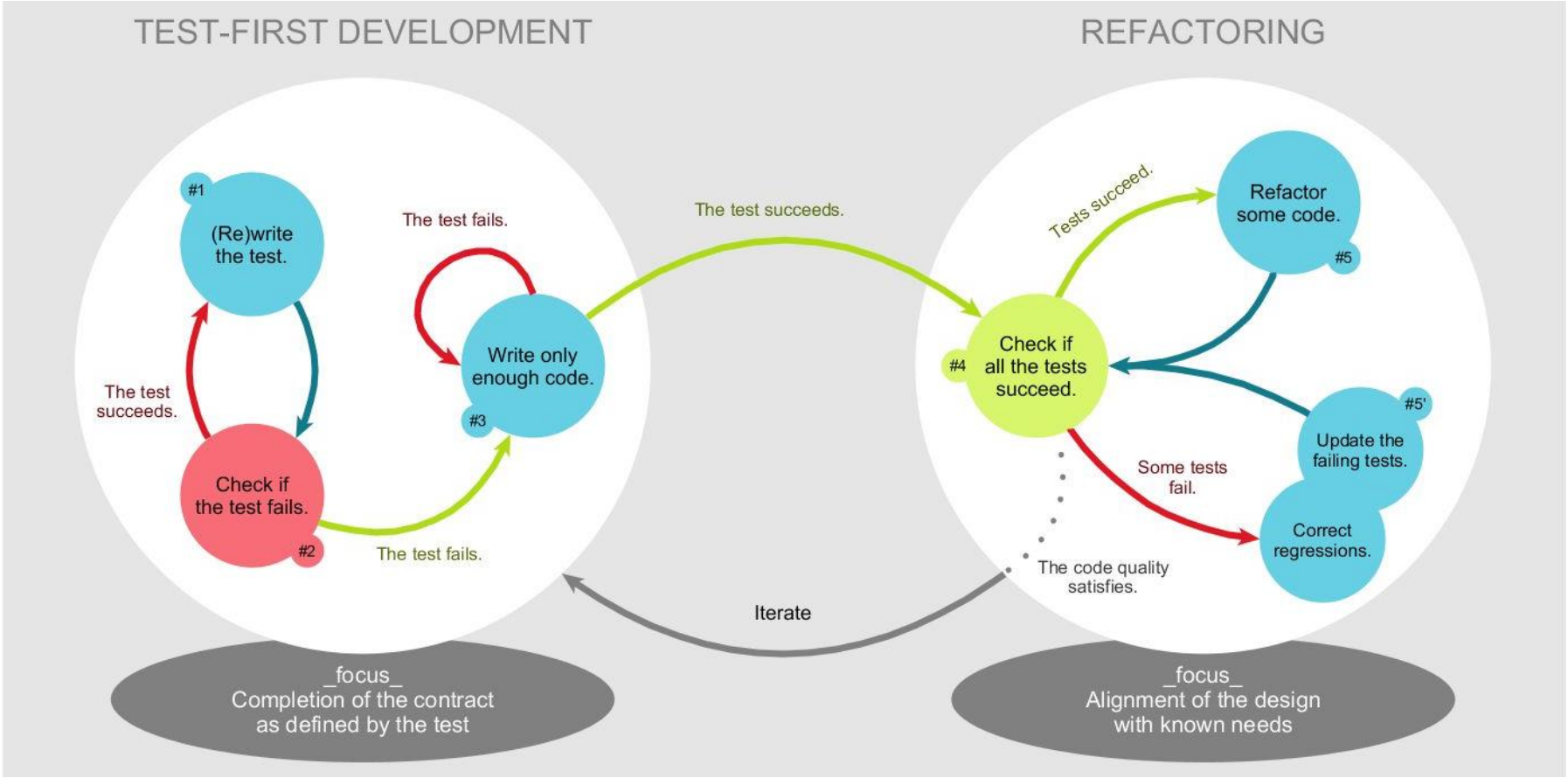


Junit testiranje u praksi

- Junite testove piše programer koji implementira funkcionalnost.
- Testove treba pisati tokom razvoja koda. Tako se smanjuje verovatnoća pojave bagova rano u razvoju. I, na taj način, se čuva dragoceno vreme gledano na duge staze.
- Razvojni inženjer više puta pokreće testove da bi proverio ponašanje određene jedinice koda, u zavisnosti od različitih ulaznih vrednosti.
- Pisanje junit testova može biti vremenski zahtevno i dosta naporno.
- Zahteva strpljenje i istrajnost od strane razvojnog tima.
- Pri pisanju junit testova, moramo biti svesni da možda nećemo uspeti da pokrijemo svaki scenario koji se može odigrati prilikom rada programa u stvarnom okruženju.

Razvoj vođen testovima (*Test Driven Development*)




TEST-FIRST DEVELOPMENT
REFACTORING
#1
(Re)write the test.
The test fails.
The test succeeds.
Write only enough code.
#3
The test succeeds.
Check if the test fails.
#2
The test fails.
Tests succeed.
Refactor some code.
#5
#4
Check if all the tests succeed.
#5'
Update the failing tests.
Some tests fail.
Correct regressions.
The code quality satisfies.
Iterate
focus
Completion of the contract as defined by the test
focus
Alignment of the design with known needs

Šta je NUnit?

- NUnit je framework za junit testiranje, koji se može koristiti za sve .NET jezike.
- Inicialno je portovan iz JUnit framework, ali je potom iznova napisan i proširen dodatnim funkcionalnostima.
- NUnit je otvorenog koda (*open source*) i dostupan pod MIT licencom.
- Alternativa u .NET svetu: MS Unit
- URL: http://www.nunit.org/ & https://github.com/nunit

NUnit - uopšteno

- Testovi su označeni anotacijama (*Attributes*)
- Test kod sadrži tvrdnje (*Assertions*)
- Podržava konfiguracione dokumente
- Testovi se mogu organizovati u više biblioteka (*Assemblies*)

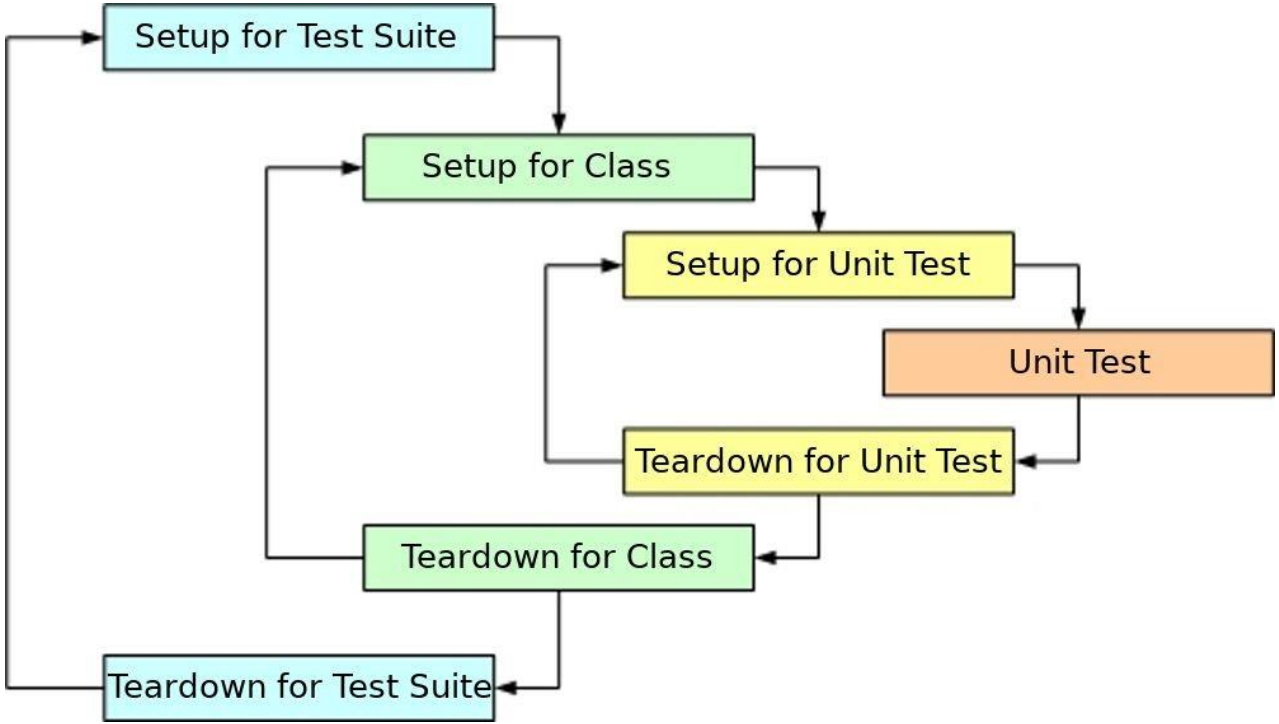
NUnit - anotacije

- **[TestFixture]** - označava klasu koje sadrži testove i, opciono, metode za postavku i čišćenje testova (*Setup & Teardown*).
- **[Test]** - označava određenu metodu, unutar test klase, da sadrži logiku samog testa.
- **[TestFixtureSetUp]** - označava metodu koja postavlja podešavanja na nivou klase. Izvršava se samo jednom, pre svih testova.
- **[TestFixtureTearDown]** - označava metodu koja vrši čišćenje na nivou klase. Izvršava se samo jednom, nakon što se izvrše svi testovi.

NUnit - anotacije

- **[SetUp]** - označava metodu koja postavlja podešavanja na nivou testa. Izvršava se jednom, pre svakog testa.
- **[TearDown]** - označava metodu koja vrši čišćenje na nivou testa. Izvršava se jednom, nakon svakog testa.
- **[ExpectedException(typeof(Exception))]** - označava da se očekuje da će doći do bacanja izuzetka. Test prolazi ukoliko se baci izuzetak.
- **[Ignore(“Not ready for primetime”)]** - koristi se ukoliko označeni test još nije spreman ili iz nekog razloga ne želimo da ga pokrećemo.

NUnit - izvršavanje testova



NUnit - tvrdnje (*Asserts*)

- Provere određenih tvrdnji (*assert*) su jedna od najvažnijih stvari u junit testiranju.
- Tvrdnje nam govore šta se očekivalo da se dogodi, ali ipak nije.
- Dobre provere tvrdnji nam pomažu pri praćenju bagova i lakšem razumevanju samih testova.

```
Assert.AreEqual(..);
Assert.AreNotEqual(..);
Assert.AreSame(..);
Assert.AreNotSame(..);
…
```

Struktura test projekta

- Aplikacija koja se razvija treba da ima test projekte izdvojene u poseban direktorijum pod imenom „Test“
- Jedan projekat treba da ima projekat koji ga testira, pri tome ime test projekta je: <ime-projekta>Test
- Jedan direktorijum u projektu treba da ime odgovarajući direktorijum u test projektu, pri tome ime test direktorijuma je: <ime-direktorijum>Test
- Jedan imenski opseg (*namespace*) treba da odgovarajući opseg u test projektu, pri tome ime opsega je: <ime-opsega>Test
- Jedna klasa u treba da ima klasu koja je testira, pri tome ime test klase je: <ime-klase>Test
- Jedna funkcija treba da ima funkciju koja je testira, pri tome ime test funkcije je: <ime-funkcije>Test

Zadatak

Implementirati interfejs Osoba koji sadrži:

```
string JMBG
string Ime
string Prezime
DateTime DatumRodjenja
Pol PolOsobe(Pol je tipa enumeracije)
```

Konstruktor treba da primi prilikom inicijalizacije JMBG, Ime I Prezime, a datum rođenja i pol osobe treba da se odredi na osnovu JMBG-a te osobe.

Zadatak

Uslovi:

- Prilikom inicijalizacije objekta klase Osoba, JMBG, ime i prezime ne smeju biti null, ime i prezime ne sme biti prazan string, JMBG mora sadržati 13 cifara.
- Prilikom određivanja datuma rodjenja, treba proveriti da li je prosleđen JMBG validan.

Napisati test slučajeve koji pokrivaju slučajeve:

- Konstruktor prima dobre parametre
- Konstruktor prima granične paramete
- Konstruktor prima loše parametre