Testiranje

[od strane samog programera]

Sadržaj

1. Uvodna reč
2. Uticaj testiranja na kvalitet software-a
3. Najbolji pristupi testiranju
4. “Vreća trikova”
5. Tipične greške
6. Zaključak

1. Reč - dve

➔ Uloga u poboljšanju kvaliteta software-a

➔ Šta se tačno testira?

- Unit testing - klase, rutine, manje izolovane celine
- Component testing - nešto veće izolovane celine
- Integration testing - dve ili više klasa, paketa, komponenti, podsistema
- Regression testing - ponavljanje “starih” testova
- System testing - finalna testiranja; performanse, bezbednost
- Ostali - beta - testovi, testovi performansi i konfiguracije

➔ TESTIRANJE != DEBAGOVANJE

2. Zašto programeri ne vole da testiraju?

- ➔ Lov na svoju grešku
- ➔ Nema mu kraja
- ➔ Ne poboljšava kod (problem loše dijete)
- ➔ Šta želiš to i dobiješ

- ➔ Šta sa rezultatima?

3. FAQ

A. Pre ili posle?
   a. uvek može posle
   b. isto utrošeno vreme za pisanje testova, ali ne i za detekciju greške
   c. programiranje “s predumišljajem”

FAQ (2)

B. Optimista je neobavešteni pesimista?

a. Programeri misle da su pokrili 95% slučajeva, a realno je ~ 50 - 60%

C. Nisu čista posla

a. p : č = 5 : 1, a ne obrnuto!

4. Savršenstvo je ideal

- ➔ Nije moguće utvrditi da je kod savršeno ispravan
- ➔ Primer:

Ime		26^{20}
Adresa		26^{20}
Telefon		10^{10}
	sacuvaj u fajl	

$$26^{20} * 26^{20} * 10^{10} \sim 10^{66}$$

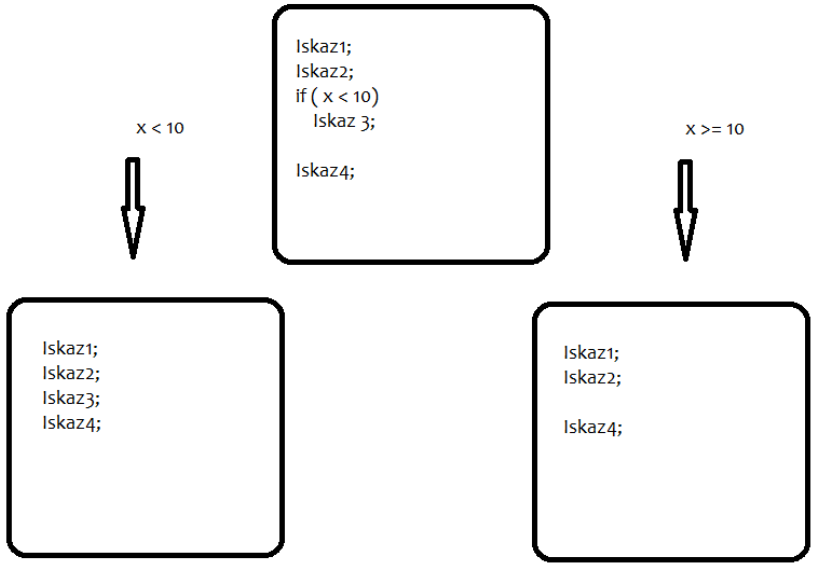
?

Ipak postoje tehnike...

A. Testovi u odnosu na strukturu koda

[od tačke A do tačke B na sve moguće načine && minimalan broj prolazaka]

- kompletan?



Ipak postoje tehnike… (2)

B. Testovi u odnosu na vrednosti promenljivih

➔ Promenljive se mogu naći u 3 stanja:

- Definisane [defined]
- Upotrebljene [used]
- Mrtve [killed]

➔ Rutine u odnosu na promenljive:

- Započete [entered]
- Završene [exited]

Ipak postoje tehnike… (3)

Sledeće situacije nisu dobre:

Defined-Defined
Defined-Exited
Defined-Killed
Entered-Killed
Entered-Used
Killed-Killed
Killed-Used
Used-Defined

Ipak postoje tehnike… (4)

- ➔Defined-Defined – dva uzastopna definisanja
- ➔Defined-Exited - definisana a neupotrebljena
- ➔Defined-Killed - definisana pa odmah poništena
- ➔Killed-Killed - dva puta poništena

Ipak postoje tehnike… (5)

C. Klase ekvivalencije grešaka
D. Nagađanje
E. Ispitivanje granica vrednosti
F. Klase loših podataka
   - ➔ premalo/previše podataka
   - ➔ pogrešnog tipa
   - ➔ pogrešna veličina

Korisno - *convenient hand checks*

➔Nije zgoreg prištedeti na vremenu

[$20,000 I $90,783.82 su u istoj klasi]

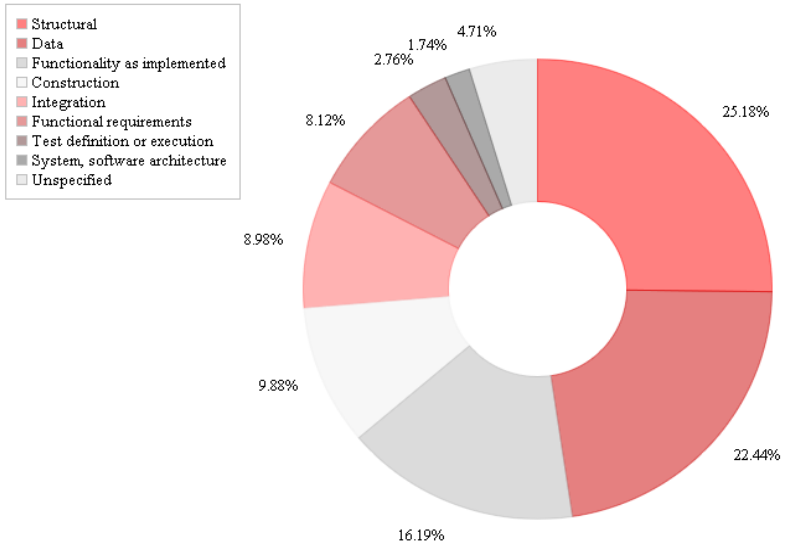
Malo statistike

- ➔ Da li su greške ravnomerno distribuirane po kodu?
  - 80% grešaka u 20% klasa i rutina
  - 85% grešaka se može ispraviti samo korekcijom jedne rutine
  - 95% prouzrokovano od strane programera, 2% od strane sistemskog software-a, 2% ostali alati i 1% hardware
  - 36% greške u kucanju

Malo statistike (2)

- ➔ 85% grešaka se rešava za manje od nekliko sati
- ➔ 15% od nekoliko sati do nekoliko dana
- ➔ 1% više

Malo statistike (3)



Čuvajte se testova

- ➔ Problem često “čuči” baš tu (pisani u letu)
- ➔ Dobra praksa:
  - Pisati ih jednako pažljivo
  - Planirati testove kad i ostale komponente
  - Ponavljanje starih testova
  - Nadograđivanje starih testova
  - Zgodni za “hand-checks”

Checklist

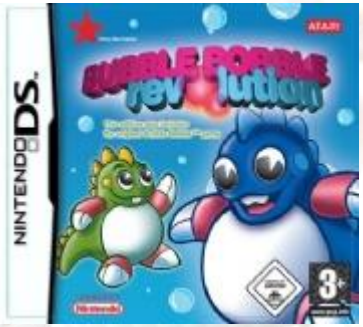
- ❑ Da li se svaki zahtev vezan za klasu ili rutinu testira?
- ❑ Da li se svaki element dizajna vezan za klasu ili rutinu testira?
- ❑ Da li je svaka linija koda testirana makar jednom? Da li je to baš najmanji broj testova neophodnih za testiranje tog koda?

Checklist (2)

- ❑ Da li su sve “putanje” kroz kod ispitane makar jednom?
- ❑ Da li su uklonjeni svi pogrešni parovi stanja i rutina?
- ❑ Da li su uzete u obzir sve najčešće greške zasnovane na ranijim iskustvima?
- ❑ Da li su sve vrednosti u svojim okvirima?

Checklist (3)

- Da li je ispitano ponašanje programa kada su vrednosti iskočile iz svojih granica?
- Da li su testovima obuhvaćeni slučajevi kad je unet pogrešan tip podataka (npr, negativan broj zaposlenih)?
- Da li je ispitana usaglašenost sa starim verzijama OS-a, odnosno sa interfejsima sa starijom verzijom?



Google

Hvala na pažnji

Literatura:

1. Code Complete, Developer Testing (22)
2. http://www.pcworld.com/article/166778/bizarre_bugs.html
3. http://listverse.com/2012/12/24/10-seriously-epic-computer-software-bugs/