Neuobičajene kontrolne strukture

Matematički fakultet, Beograd
Seminarski rad iz kursa Razvoj softvera 2

–

Matijević Luka 1106/14

Prezentacija obuhvata:

- Na koji način višestruki povratak iz rutine može doprineti čitljivostu koda
- Savete za upotrebu rekurzije
- Debata vezana za goto naredbe
- Ukupna perspektiva

Smernice za korišćenje return naredbe

- Koristiti return kada poboljsava citljivost koda
- Koristiti klauzule čuvare
- Minimizovati broj return-ova u svakoj rutini

Smernice za korišćenje return naredbe

- Koristiti return kada poboljšava čitljivost koda
  - U određenim rutinama, jednom kada saznamo odgovor, zelimo da se odmah vratimo u pozivajucu rutinu.
- Koristiti klauzule čuvare
- Minimizovati broj return-ova u svakoj rutini

Smernice za korišćenje return naredbe

- Koristiti return kada poboljsava citljivost koda
- Koristiti klauzule čuvare
  - Korisiti klauzule čuvare za pojednostavljenje obrade grešaka. Umesto ugnježdenih if naredbi, bolje je posle svake provere uslova staviti return ili exit
- Minimizovati broj return-ova u svakoj rutini

Smernice za korišćenje return naredbe

- Koristiti return kada poboljsava citljivost koda
- Koristiti klauzule čuvare
- Minimizovati broj return-ova u svakoj rutini
  - Tesko je razumeti rutinu ako, čitajući njen kraj, nismo svesni mogućnosti da se ona vratila u rutinu pozivaoca ranije u kodu. Iz ovog razloga treba oprezno koristiti return, samo kada će poboljšati čitljivost koda.

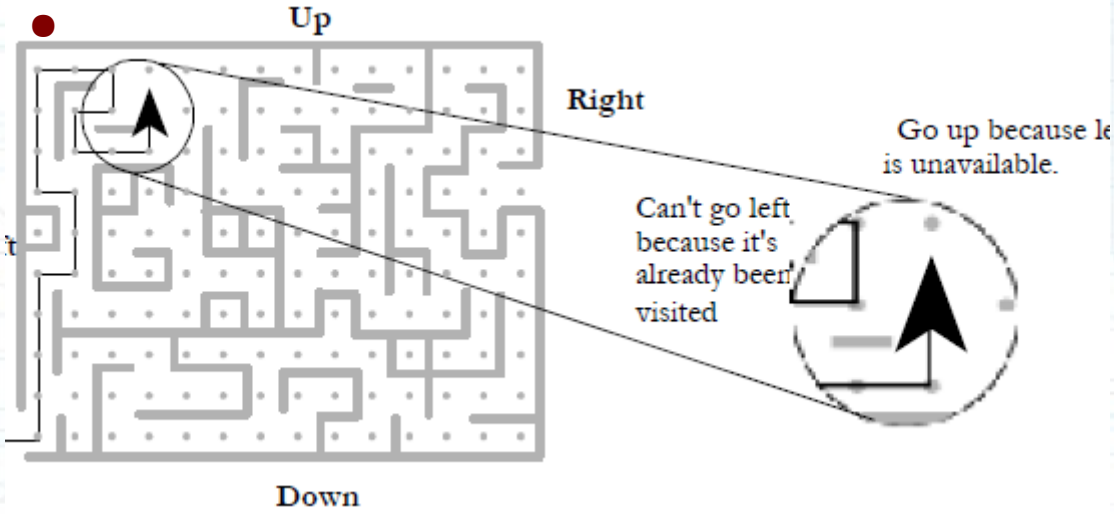
Rekurzija

- Kod rekurzije, rutina rešava manji deo problema, deli problem na manje delove, a zatim poziva samu sebe da reši manje delove
- Uglavnom se koristi kada je mali deo problema lako rešiti, a veći deo lako podeliti na manje
- Generalno, rekuryija dovodi do manjeg koda, sporijeg izvršavanja I velikog zauzimanja steka

Rekurzija

- Primer rekurzije

  - Traženje izlaza iz lavirinta



Saveti za korišćenje rekurzije

- Uveriti se da postoji izlaz iz rekurzije
- Koristiti sigurnosne brojače da bi se izbegla beskonačna rekurzija
- Ograničiti rekurziju na jednu rutinu
- Paziti na stek
- Ne koristiti rekurziju za faktorijel ili Fibonačijev niz

Saveti za korišćenje rekurzije

- Uveriti se da postoji izlaz iz rekurzije
  - Proveriti da li rutina ima nerekurzivnu putanju. To uglavnom znači da rutina ima test kojim prekida dalju rekurziju ako ona nije potrebna
- Koristiti sigurnosne brojače da bi se izbegla beskonačna rekurzija
- Ograničiti rekurziju na jednu rutinu
- Paziti na stek
- Ne koristiti rekurziju za faktorijel ili Fibonačijev niz

Saveti za korišćenje rekurzije

- Uveriti se da postoji izlaz iz rekurzije
- Koristiti sigurnosne brojače da bi se izbegla beskonačna rekurzija
  - Ako koristimo rekurziju u situacijama gde ne možemo napraviti jednostavan test za izlaz iz rekurzije, koristiti sigurnosne brojače da bi se izbegla beskonačna rekurzija
- Ograničiti rekurziju na jednu rutinu
- Paziti na stek
- Ne koristiti rekurziju za faktorijel ili Fibonačijev niz

Saveti za korišćenje rekurzije

- Uveriti se da postoji izlaz iz rekurzije
- Koristiti sigurnosne brojače da bi se izbegla beskonačna rekurzija
- Ograničiti rekurziju na jednu rutinu
  - Ukloniti ciklične rekurzije ili ih ograničiti sigurnosnim brojačem
- Paziti na stek
- Ne koristiti rekurziju za faktorijel ili Fibonačijev niz

Saveti za korišćenje rekurzije

- Uveriti se da postoji izlaz iz rekurzije
- Koristiti sigurnosne brojače da bi se izbegla beskonačna rekurzija
- Ograničiti rekurziju na jednu rutinu
- Paziti na stek
  - Paziti na alokaciju lokalnih promenljivih u rekurzivnim rutinama, pogotovo za velike objekte. Poželjnije je koristiti heap nego stack
- Ne koristiti rekurziju za faktorijel ili Fibonačijev niz

Goto

- Argumenti protiv goto
- Argumenti za goto
- Goto debata
- Obrada grešak I goto

Argumenti protiv goto

- Kod koji sadrži goto naredbe teži je za formatiranje
- Korišćenje goto naredbi otežava kompajlersku optimizaciju
- Dovodi do kršenja principa po kome bi kod trebao da tece sa vrha na dole

Argumenti za goto

- Dobro postavljen goto može eliminisati potrebu za dupliranjem koda
- Koristan je u rutinama koje alociraju resurse, vrše neku operaciju sa njima, a zatim dealociraju resurse.
- U nekim slučajevima, može proizvesti kraći I brži kod

Goto debata

- Glavna crta većine diskusija je plitak pristup problemu
- Knjige uglavnom prezentuju trivijalne primere u kojima izbacuju goto naredbu, ali to nisu realne situacije u kojima programeri koriste goto
- Uglavnom se ne pominju slučajevi u kojima programer, iako svestan alternativa, odluči da koristi goto radi čitljivijeg koda

Obrada grešaka I goto

- Pisanje veoma interaktivnog koda dovodi do situacije da moramo posvetiti dosta pažnje obradi gresaka, kao I čišćenju resursa ukoliko dođe do greske.
- Ove rutine su tipični primeri kada programeri koriste goto
- Slični primeri su rutine koje treba da alociraju I počiste resurse, kao što su konekcija sa bazom, memorija ili privremeni fajlovi

Strategije za izbacivanje goto naredbi

- Napisati sa ugnježdenim if naredbama
- Napisati sa statusnom promenljivom
- Napisati sa try-finally

Strategije za izbacivanje goto naredbi

- Napisati sa ugnježdenim if naredbama
  - If naredbe se moraju napisati tako da se svaka izvršava samo ako prethodni test uspe.
  - Glavna mana ovog pristupa je što postoji velika ugnježdenost
  - Distanca između koda za obradu greške I koda u kome se desila greška je prilično velika
- Napisati sa statusnom promenljivom
- Napisati sa try-finally

Strategije za izbacivanje goto naredbi

- Napisati sa ugnježdenim if naredbama
- Napisati sa statusnom promenljivom
  - Kreiramo promenljivu koja indikuje da li je rutina u stanju greške
  - Prednost je što ne dovodi do duboko indentovanog koda
  - Problem je što statusne promenljive nisu tako česta praksa, kao što bi trebale da budu
  - Njihovo korišćenje treba u potpunosti dokumentovati
- Napisati sa try-finally

Strategije za izbacivanje goto naredbi

- Napisati sa ugnježdenim if naredbama
- Napisati sa statusnom promenljivom
- Napisati sa try-finally
  - Treba okružiti kod sa try naredbom, a zatim smestiti kod za čišćenje resursa u finally sekciju
  - Prednost pristupa je da se postiže dobra čitljivost bez korišćenja goto naredbi
  - Mana pristupa je što ne podržavaju svi jezici ovaj pristup

Poređenje pristupa

- Goto otklanja duboko indentovanje I bespotrebne testove, ali sadrži goto naredbe
- Pristup sa ugnježdenim if naredbama otklanja potrebu za goto naredbama, ali rezultira u veoma indentovanom kodu
- Statusne promenljive izbegavaju goto I indentovanje, ali uvode dodatne testove
- Pristup try-finally otklanja sve prethodne probleme, ali nije podržan u svim jezicima

Osnovne smernice za korišćenje goto naredbe

- Koristiti goto za simulaciju kontrolnih sruktura koje nisu podržane u jezicima
- Ne koristiti goto kada postoji ugrađen ekvivalent
- Izmeriti koliko se poboljšanje dobilo korišćenjem goto naredbe. Dokumentovati
- Ograničiti sebe najednu goto naredbu po rutini, osim ako ne pravimo kontrolne strukture

Nastavak...

- Paziti da goto naredbe uvek pokazuju na napred, a nikada pozadi, osim ako se ne programiraju kontrolne strukture
- Postarati se da sve goto labele budu iskorišćene
- Osigurati se da goto naredbe ne stvaraju nedostižan kod

Hvala na pažnji!