Domaći zadatak 09 – Branislav Manojlović kp11
KI301-DZ09

Tekst zadatka

1. Odabrati jedan algoritam i napisati pseudokod,
2. Opisati prednost napisanog pseudokoda,
3. Izvršiti proveru pseudokoda shodno opisanim pravilima za pisanje pseudokoda na predavanju,
4. Ukoliko su uočene neke nepravilnosti u pisanju pseudokoda, napisati ih i objasniti kako je moguće unaprediti pseudokod.

Rešenje:

1) Algoritam za koji pravim pseudokod vrši izračunavanje jednačine: x=(3x-2)/x, u Java programu.

Pseudokod algoritma:

korak 1: START programme,

korak 2: OPEN Scanner (aktiviram klasu Scanner koja omogućava unos korisničkih podataka putem tastature),

korak 3: INITIALIZE variable "ulaz", TYPE double (inicijalizujem promenljivu ulaz tipa double) AND add VALUE (i dodeljujem joj unetu vrednost),

korak 4: CALL method "jednacina()",

korak 5: FORWARD value of "ulaz" to method "jednacina()",

korak 6: ACCEPT value forwarded from "jednacina(ulaz)" and INTIALIZE variable "x",

korak 7: INITIALIZE variable "rez" with VALUE = 0,

korak 8: IF (x == 0),

korak 9: THROW: NullPointerException,

korak 10: WRITE: x ne moze imati vrednost 0!!!,

korak 11: ELSE rez = (3*x – 2) / x,

korak 12: RETURN: rez,

korak 13: ACCEPT rez,

korak 14: WRITE rez,

korak 15: CLOSE Scanner

korak 16: END programme.

Izvorni kod programa za izračunavanje jednačine: x=(3x-2)/x

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package zadatak12;
import java.util.Scanner;
/**
 *
 * @author Branislav
 */
public class JednacinaDomaci {
    static Scanner sc = new Scanner(System.in);
    ///metoda koja resava zadatu jednacinu x=(3x-2)/x
    public static double jednacina(double x){
        double rez=0;
        if(x==0) {
            throw new NullPointerException("X ne moze imati vrednost 0!!!");
        }else{
            rez = (3*x-2)/x ;
        }
        return rez;
    }

    public static void main(String[] args) {
        double ulaz;
        System.out.println("Unesite broj x: ");
        ulaz = sc.nextDouble();
        System.out.println("Rezultat je: " + jednacina(ulaz));
        sc.close();
    }
}
```

2) Prednost napisanog pseudokoda, ogleda se u tome što je program napisan u pseudokodu prikazan u vidu liste instrukcija koje trebaju da se izvrše po određenom redu. Ovaj način predstavljanja programa je jednostavan za pregled i razumevanje i licima koja nisu programeri a to je veoma važno kada je u pitanju biznis aspekt razvoja aplikacije. Time se u razvoj aplikacije može uključiti veći broj aktera i mogu se na jednostavniji i brži način razjasniti i definisati funkcionalnosti aplikacije.

3) Provera napisanog pseudokoda se vrši po sledećim pravilima:
   - Prvo zapišemo listu osnovnih koraka,
   - Zatim odlučujemo da li je osnovne korake potrebno razbiti na manje korake,
   - Na kraju zapisujemo sve korake redom jedan za drugim u skladu sa radnim tokom (work-flow) aplikacije. Pri pisanju strogo izbegavamo sintaksu program. jezika.

4) Pseudokod je moguće unaprediti tako što se glavni algoritam aplikacije optimizuje u smislu da se iz njega izdvoje podprogrami koji mogu sami za sebe da budu opisani posebnim algoritmom, pa samim tim i pseudokodom.

Primer izgleda pseudokoda nakon izvršene optimizacije je prikazan u nastavku.

Pseudokod Glavnog algoritma posle optimizacije:

korak 1: START programme,

korak 2: OPEN Scanner (aktiviram klasu Scanner koja omogućava unos korisničkih podataka putem tastature),

korak 3: INITIALIZE variable "ulaz", TYPE double (inicijalizujem promenljivu ulaz tipa double) AND add VALUE (i dodeljujem joj unetu vrednost),

korak 4: CALL method "jednacina()",

korak 5: FORWARD value of "ulaz" to method "jednacina()",

korak 6: ACCEPT rez,

korak 7: WRITE rez,

korak 8: CLOSE Scanner

korak 9: END programme.

Pseudokod Podalgoritma (podprograma – *metode jednacina()*) posle optimizacije:

korak 1: START programme,

korak 2: ACCEPT value forwarded from "jednacina(ulaz)" and INTIALIZE variable "x",

korak 3: INITIALIZE variable "rez" with VALUE = 0,

korak 4: IF (x == 0),

korak 5: THROW: NullPointerException,

korak 6: WRITE: x ne moze imati vrednost 0!!!,

korak 7: ELSE rez = (3*x – 2) / x,

korak 8: RETURN: rez,

korak 9: END programme.

MSc Branislav Manojlović
Novi Sad 12.06.2018.