Programiranje 2

Principi pisanja programa

Jelena Graovac

www.matf.bg.ac.rs/~jgraovac

Beograd, 20. februar, 2020.

Pregled

Pregled

Principi pisanja programa

- Razumljivost i čitljivost programa, iako nebitna za računar, od ogromne je važnosti za kvalitet i upotrebljivost programa
- Održavanje sistema često ne rade oni programeri koji su program napisali
- Razumljivost programa omogućava lakšu analizu njegove ispravnosti i složenosti
- Biće navedene preporuke iz:
  - teksta Linux Kernel Coding Style, autora Linusa Torvaldsa
  - knjige The Practice of Programming, autora Brajana Kernigena i Roba Pajka

Principi pisanja programa

- Timski rad i konvencije
  - Nije najvažnije koja konvencija se koristi nego koliko strogo se nje pridržava
  - Jedan isti programer treba da bude spreman da u različitim timovima i različitim projektima koristi različite konvencije
  - Sistemi za kontrolu verzija (SVN, Git, CVS, Mercurial, Bazaar)

Principi pisanja programa

- Vizualni elementi programa
  - Broj karaktera u redu (obično 80 karaktera u redu)
    - Ukoliko red programa ima više od 80 karaktera, to najčešće ukazuje na to da kôd treba reorganizovati
  - Broj naredbi u redu, zagrade i razmaci
  - Nazubljivanje teksta programa (Python - utiče i na značenje)
  - Od pomoći mogu da budu specijalizovani editori teksta ili editori uključeni u integrisana razvojna okruženja (engl. IDE, Integrated Development Environment)

Principi pisanja programa

- Imenovanje promenljivih i funkcija
  - Kamilja notacija (na primer, brojKlijenata)
  - Notacija sa podvlakom (na primer, broj_klijenata)
  - Mađarska notacija (na primer, uBrojKlijenata)
  - Preporuka za dužinu imena promenljivih
  - Pravila za lokalne promenljive (brojače u petlji)
  - Koristiti sugestivna imena, ne generička
  - Imena funkcija treba da budu bazirana na glagolima
  - Kod nekih promenljivih se koristi isti prefiks
  - Engleski jezik

Principi pisanja programa

- Pisanje izraza
  - Pisanje izraza u jednostavnom i intuitivno jasnom obliku
  - Umesto
    ```
    !(c < '0') && !(c > '9')
    ```
    bolje je
    ```
    '0' <= c && c <= '9'
    ('0' <= c) && (c <= '9')
    '0'<=c && c<='9'
    ```
  - Umesto
    ```
    prestupna = godina % 4 == 0 && godina % 100 != 0 || godina % 400 == 0;
    ```
    bolje je
    ```
    prestupna = ((godina%4 == 0) && (godina%100 != 0)) || (godina%400 == 0);

    prestupna = (godina % 4 == 0 && godina % 100 != 0) || (godina % 400 == 0);
    ```

Principi pisanja programa

- Pisanje izraza (nastavak)
  - Umesto
    ```
    c += (d = a < b ? printf("a") : prinf("b"));
    ```
    bolje je
    ```
    if (d = a < b)
        printf("a");
    else
        prinf("b");
    c += d;
    ```
  - Umesto
    ```
    str[i++] = str[i++] = ' ';
    ```
    bolje je
    ```
    str[i++] = ' ';
    str[i++] = ' ';
    ```
  - Nakon dodele a[a[1]]=2; koju vrednost ima element a[a[1]]? Da li nužno ima vrednost 2?

Principi pisanja programa

- Pisanje izraza (nastavak)
  - Umesto
    `c += (d = a < b ? printf("a") : prinf("b"));`
    bolje je

```
if (d = a < b)
    printf("a");
else
    prinf("b");
c += d;
```

  - Umesto
    `str[i++] = str[i++] = ' ';`
    bolje je
    `str[i++] = ' ';`
    `str[i++] = ' ';`
  - Nakon dodele a[a[1]]=2; koju vrednost ima element a[a[1]]? Da li nužno ima vrednost 2? Šta će se desiti ako je a[1] bila jednaka 1, a vrednost a[2] različita od 2?

Principi pisanja programa

- Korišćenje idioma
  - Idiomi su ustaljene jezičke konstrukcije koje predstavljaju celinu (npr. for (i = 0; i < n; i++))
  - Umesto

```
i=0;
while (i <= n-1)
  a[i++] = 1.0;

for (i = 0; i < n; )
  a[i++] = 1.0;

for (i = n; --i >= 0; )
  a[i] = 1.0;
```

bolje je

```
for (i = 0; i < n; i++)
  a[i] = 1.0;
```

Principi pisanja programa

- Korišćenje idioma (nastavak)
  - for (;;)
  - while ((c = getchar()) != EOF)
- Korišćenje konstanti
  - ```
    #define MAKS_IME_ULICE 50
    char imeUlice[MAKS_IME_ULICE];
    ili još bolje
    const unsigned int MAKS_IME_ULICE = 50;
    ```
  - Povratne vrednosti funkcija
  - Konstante od značaja za čitav program
    ```

Principi pisanja programa

- Korišćenje konstanti (nastavak)
  - Kodovi karaktera
    - Umesto
      `if (65 <= c && c <= 90)`
      bolje je
      `if ('A' <= c && c <= 'Z')`
      a još je bolje
      `if (isupper(c))`
    - Bolje je pisati NULL i `'\0'` umesto 0
    - Umesto 2 ili 4 bolje je pisati sizeof(int)
    - Umesto sizeof(int) bolje je pisati sizeof(a[0]) ako je niz a tipa int[]

Principi pisanja programa

- Korišćenje makroa sa argumentima
  - Treba ih izbegavati
- Pisanje komentara
  - Komentari ne treba da objašnjavaju ono što je očigledno
    - return OK; /* vrati OK */ - nema smisla
    - k += 1.0; /* k se uvecava za 1.0 */ - nema smisla
    - k += 1.0; /* kamatna stopa veca je za 1.0 */ - ima smisla
  - Komentari treba da budu koncizni
  - Komentari treba da budu uskladeni sa kôdom
  - Komentarima treba objasniti ulogu datoteka i globalnih objekata
  - Loš kôd ne treba komentarisati, već ga popraviti

Principi pisanja programa

- Pisanje komentara (nastavak)
  - Komentari treba da budu laki za održavanje

```
/**************************************
* Funkcija area racuna povrsinu trougla *
**************************************/
```

  - Komentari mogu da uključuju standardne fraze
    - TODO
    - FIXME
    - HACK
    - BUG
    - XXX

Principi pisanja programa

- Modularnost
  - Podela programa na module treba da omogući:
    - Razumljivost
    - Upotrebljivost
  - Modularnost i podela na funkcije
  - Modularnost i podela na datoteke

Principi pisanja programa

- Modularnost i podela na datoteke (nastavak)
  - Datoteke zaglavlja obično imaju sledeću strukturu:
    - definicije tipova;
    - definicije konstanti;
    - deklaracije globalnih promenljivih (uz navodenje kvalifikatora extern);
    - deklaracije funkcija (uz navodenje kvalifikatora extern)
  - a izvorne datoteke sledeću strukturu:
    - uključivanje sistemskih datoteka zaglavlja;
    - uključivanje lokalnih datoteka zaglavlja;
    - definicije tipova;
    - definicije konstanti;
    - deklaracije/definicije globalnih promenljivih;
    - definicije funkcija.

Pregled

Obnavljanje

- Navedite barem jedan alat za kontrolu verzija.

Obnavljanje

- Navedite barem jedan alat za kontrolu verzija.
- Koje ime bi, u kamiljoj notaciji, imala promenljiva `int broj_cvorova`?

Obnavljanje

- Navedite barem jedan alat za kontrolu verzija.
- Koje ime bi, u kamiljoj notaciji, imala promenljiva `int broj_cvorova`?
- Koje ime bi, u mađarskoj notaciji, nosile promenljive `float* X` i `int* broj_cvorova`?

Obnavljanje

- Koliko se preporučuje da najviše ima karaktera u jednoj liniji programa i koji su razlozi za to?

Obnavljanje

- Koliko se preporučuje da najviše ima karaktera u jednoj liniji programa i koji su razlozi za to?
- Ukoliko neka linija našeg programa ima 300 karaktera, šta nam to sugeriše?

Obnavljanje

- Koliko se preporučuje da najviše ima karaktera u jednoj liniji programa i koji su razlozi za to?
- Ukoliko neka linija našeg programa ima 300 karaktera, šta nam to sugeriše?
- Kada je prihvatljivo da jedna linija programa ima više naredbi?

Obnavljanje

- Koliko se preporučuje da najviše ima karaktera u jednoj liniji programa i koji su razlozi za to?
- Ukoliko neka linija našeg programa ima 300 karaktera, šta nam to sugeriše?
- Kada je prihvatljivo da jedna linija programa ima više naredbi?
- U kojim situacijama se obično neka linija programa ostavlja praznom?

Obnavljanje

- Koliko se preporučuje da najviše ima karaktera u jednoj liniji programa i koji su razlozi za to?
- Ukoliko neka linija našeg programa ima 300 karaktera, šta nam to sugeriše?
- Kada je prihvatljivo da jedna linija programa ima više naredbi?
- U kojim situacijama se obično neka linija programa ostavlja praznom?
- Zašto za nazubljivanje teksta programa nije preporučljivo koristiti tabulator?

Obnavljanje

- Koliko se preporučuje da najviše ima karaktera u jednoj liniji programa i koji su razlozi za to?
- Ukoliko neka linija našeg programa ima 300 karaktera, šta nam to sugeriše?
- Kada je prihvatljivo da jedna linija programa ima više naredbi?
- U kojim situacijama se obično neka linija programa ostavlja praznom?
- Zašto za nazubljivanje teksta programa nije preporučljivo koristiti tabulator? Umesto tabulatora, koliko se obično koristi blanko karaktera?

Obnavljanje

- Napisati sledeće naredbe na prihvatljiviji način:
  if (!(c == 'y' || c == 'Y')) return;

Obnavljanje

- Napisati sledeće naredbe na prihvatljiviji način:
  if (!(c == 'y' || c == 'Y')) return;
  length = (length < BUFSIZE) ? length : BUFSIZE;

Obnavljanje

- Napisati sledeće naredbe na prihvatljiviji način:
  if (!(c == 'y' || c == 'Y')) return;
  length = (length < BUFSIZE) ? length : BUFSIZE;
  flag = flag ? 0 : 1;

Obnavljanje

- Napisati sledeće naredbe na prihvatljiviji način:
  if (!(c == 'y' || c == 'Y')) return;
  length = (length < BUFSIZE) ? length : BUFSIZE;
  flag = flag ? 0 : 1;
- Šta su to „magične konstante“ i da li one popravljaju ili kvare kvalitet programa?

Obnavljanje

- Napisati sledeće naredbe na prihvatljiviji način:
  if (!(c == 'y' || c == 'Y')) return;
  length = (length < BUFSIZE) ? length : BUFSIZE;
  flag = flag ? 0 : 1;
- Šta su to „magične konstante“ i da li one popravljaju ili kvare kvalitet programa?
- Kako se izbegava korišćenje „magičnih konstanti“ u programu?

Obnavljanje

- Napisati sledeće naredbe na prihvatljiviji način:
  if (!(c == 'y' || c == 'Y')) return;
  length = (length < BUFSIZE) ? length : BUFSIZE;
  flag = flag ? 0 : 1;
- Šta su to „magične konstante“ i da li one popravljaju ili kvare kvalitet programa?
- Kako se izbegava korišćenje „magičnih konstanti“ u programu?
- Šta je, kako bi kôd bio lakši za održavanje, bolje koristiti umesto deklaracije char ImeKorisnika[50]?

Obnavljanje

- Navedite barem jedan alat za automatsko generisanje tehničke dokumentacije.

Obnavljanje

- Navedite barem jedan alat za automatsko generisanje tehničke dokumentacije.
- Kojim se komentarom/markerom obično označava mesto u kôdu na kojem treba dodati kôd za neki podzadatak?

Obnavljanje

- Navedite barem jedan alat za automatsko generisanje tehničke dokumentacije.
- Kojim se komentarom/markerom obično označava mesto u kôdu na kojem treba dodati kôd za neki podzadatak?
- Koji se marker u okviru komentara u kôdu obično koristi za označavanje potencijalnih propusta i/ili grešaka koje naknadno treba ispraviti?

Obnavljanje

- Navedite barem jedan alat za automatsko generisanje tehničke dokumentacije.
- Kojim se komentarom/markerom obično označava mesto u kôdu na kojem treba dodati kôd za neki podzadatak?
- Koji se marker u okviru komentara u kôdu obično koristi za označavanje potencijalnih propusta i/ili grešaka koje naknadno treba ispraviti?
- Koji je preporučeni obim jedne datoteke programa?

Pregled

Literatura

- Slajdovi su pripremljeni na osnovu knjige
  Predrag Janičić, Filip Marić: Programiranje 2
- Za pripremu ispita, slajdovi nisu dovoljni, neophodno je koristiti knjigu!