Dvanajste vaje iz računalništva I

Jurij Buh

12.1.1999

1. Arhitektura programske opreme

Arhitektura programske opreme pomeni samo strukturo programa, v kateri so vkljucčene posamezne funkcije. Tu moramo tocčno premisliti koliko funkcij bo uporabljal glavni program, kakšno vlogo bodo imele posamezne funkcije, kakšne parametre potrebujejo za izvajanje, kateri so vhodni in izhodni parametri ter na kakšen način so prenešeni – ali po vrednosti, ali po referenci.

Za načrtovanje programa se uporabljata dva sistema: Od zgoraj navzdol oz. *top down*, ter od spodaj navzgor *bottom up*. Prvi pomeni, da začnemo načrtovamje v glavnem programu (main), kjer prvo definiramo funkcije, ki bodo zastopane v programu, tako da se bo vsaka stvar odvijala v svoji funkciji, nato pa preidemo na načrtovanje posameznih funkcij, ter njihovih podfunkcij. Pri *bottom up* sistemu pa se načrtovanje začne pri samem jedru problema, ter nato po potrebi dodajamo še ostale funkcije, ko imamo vse, jih še samo povežemo v glavnem programu.

2. Načrtovanje programa za izračun determinante matrike

2.1 Grobi opis funkcij, ki jih bomo potrebovali

Za izracun determinante je potrebno najprej prebrati matriko, jo izpisati ter nato izračunati njeno determinanto. Ker hočemo da se vsaka stvar izvrši v svoji funkciji pač potrebujemo tri funkcije. In sicer:

1. `BeriMatriko`: Funkcija ki prebere matriko;
2. `IzpisiMatriko`: Funkcija ki jo izpiše;
3. `DeterminantaMatrike`: Funkcija ki izračuna determinanto;

Ker so glavne funkcije definirane, moramo dolociti le še vhodne in izhodne parametre za posamezno funkcijo.

2.1.1 Parametri funkcije `BeriMatriko`

Ker v to funkcijo ne pride nič iz programa, je brez vhodnih parametrov, vrniti pa mora dimenzijo matrike, da se ustvari potreben prostor v pomnilniku, ter vrednost matrike (ij-te elemente). Pišemo :

```
BeriMatriko(in: -; out: dimenzija, matrika)
```

Pri tej funkciji nastopi težava, ker nam mora vrniti dve vrednosti, in sicer dimenzijo matrike ter vrednost matrike. Zato bomo rabili še dve podfunkciji in sicer:

`BeriVrednost:` funkcija, ki dobi kot vhodni parameter niz z opisom količine, ki naj jo prebere, nato zahteva željeno količo, jo prebere ter vrne količino. Vrnjena količina bo uporabljena kot dimenzija.

```
BeriVrednost(in: *opis kolicine; out: vrednost = dim)
```

`BeriVrstico:` Ta funkcija ima za vhodni podatek dimenzijo matrike, ter nato po i-tih stolpcih bere vredost j-te vrstice, vrne pa vrstico matrike ($0 \leq i, j < dim$).

```
BeriVrstico(in: dim; out: vrstica matrike)
```

2.1.2 Parametri funkcije `IzpisiMatriko`

Funkcija `IzpisiMatriko`, rabi za izpis matrike dimenzijo ter vrednost matrike. Ker v program nič ne vrne je brez izhodnih parametrov.Pišemo:

```
IzpisiMatriko(in: dimenzija, matrika; out: -
```

2.1.3 Parametri funkcije `DeterminantaMatrike`

Ta funkcija potrebuje za izračun determinante dimenzijo matrike, ter vrednost, vrniti pa nam mora vrednost determinante, ki se mora tudi izpisati.

Podrobnejši opis na naslednjih vajah!

2.2 Nekaj o parametrih

Parametri so vhodni in izhodi podatki. Vhodni podatki so tisti, ki v funkcijo vstopajo, izhodni pa so tisti, ki jih funkcija vrne v program. Parametri so lahko prenešeni po vrednosti ali po referenci:

Po vrednosti: Sprememba parametrov se ne odraža navzven. Vsi vhodni parametri so načeloma prenešeni po vrednosti.

Po referenci: Sprememba parametrov se odraža navzven. V c-ju poteka prenos izhodnih parametrov po referenci, zato v funkciji pred najavo parametra pišemo `*`.

Zavedati se moramo, da prenos polj, matrik ali vektorjev *vedno* poteka po referenci, zato pred najavo *ne* pišemo `*`.

Obstaja še dodatna možnost, da iz funkcije kaj dobimo in sicer kot rezultat s pomočjo `return`. Vendar nam lahko funkcija vrne samo eno preprosto vrednost (npr. `int`, `char`, `float` ipd.). Pred samo funkcijo, je potrebno tudi najaviti tip rezultata funkcije. Torej če rezultat ni, pišemo `void`.

2.3 Zapis glav funkcij

Ker sedaj vemo kakšni so vhodni in izhodni parametri že lahko zapišemo glave funkcij

1. Glava glavnega programa:

   ```
   int main(void)
   ```

   Kot opazimo je pred main oznaka za cela števila `int`, čeprav nam program nič ne vrne:

   ```
   main(in:  -; out:-)
   ```

   `int` pišemo le zaradi dogovora.

2. Funkcija za branje matrike

   ```
   void BeriMatriko(int *dimenzija, int matrika[max_vel][max_vel])
   ```

   Obe vrednosti sta prenešeni po referenci, vendar pred matriko ni potrebno postavljati `*`, ker se matrike vedno prenašjo po referenci.

3. Funkcija za branje vrstice:

   ```
   void BeriVrstico(int dimenzija, int vrstica[max_vel])
   ```

   Tudi tu je vrstica prenešena po referenci, ker je to vektor.

4. Funkcija za branje ene vrednosti:

   ```
   int BeriVrednost(char *opis)
   ```

   `int` pred funkcijo je zato, ker smo se odločili, da nam funkcija vrne vrednost kot rezultat. Zvezdica pred opis pa pomeni, da je opis niz, ki pa je prenešen po vrednosti.

5. Funkcija za izpis matrike:

   ```
   void IzpisiMatriko (int dimenzija, int matrika[max_vel][max_vel])
   ```