C programavimo kalba

1 paskaita

Įvadas, standartai, programos struktūra, duomenų tipai

doc. dr. Dalius Mažeika
e-paštas Dalius.Mazeika@sc.vgtu.lt
VGTU SC L317
www.vgtu.lt/usr/dma
e-stud.vgtu.lt

Kurso turinys

- Įvadas, standartai, programos struktūra
- Duomenų tipai
  - Kintamieji, konstantos, jų tipai, tipų keitimas;
- Išraiškos ir operatoriai
- Ciklo ir sąlygos operatoriai
- Funkcijos
- Nuorodos (reference) ir rodyklės (pointer)
- Masyvai
- Eilutės ir operacijos su jomis
- Struktūros
- Priešprocesoriaus direktyvos
- C bibliotekos
  - Eilučių, simbolių apdorojimas, I/O, matematinės funkcijos, laiko ir datos, bendro pobūdžio.

Literatūros sąrašas

1. J. Blonskis, V. Bukšnaitis, V. Jusas, R. Marcinkevičius, A. Misevičius. **C++ Builder. Mokomoji knyga.** Smaltija, 2005.

2. A. Vidžiūnas. **C++ ir C++ Builder pradmenys.** Smaltija, 2003.
3. Ž. Kancleris. Programavimo kalbos ir skaičiavimų receptai. I DALIS, C ir C++ įvadas. VU, 2004.

4. http://freecomputerbooks.com/
5. http://www.vgtu.lt/usr/dma/c.htm
6. http://sc-algis.vgtu.lt

Įvadas

Programavimas – tai problemos sprendimo užrašymas kompiuterinei sistemai suprantama forma, kurį ji gali įvykdyti.

Norint tinkamai parašyti programą:

- Reikia mokėti išspręsti problemą popieriuje "rankomis"
- Mokėti sudaryti sprendimo kelią (algoritmą), naudojant bendruosius operatorius
- Mokėti vieną iš programavimo kalbų, suprantamą kompiuteriui.

*Programming is **not** about **mathematics**, it is about **organisation*** !

(Rob Miles "Introduction to C Programming" 1995)

Reikalinga speciali programavimo kalba, nes:

- Įprastinė kalba per daug sudėtinga (sinonimai, antonimai, šauktukai ir t .t.)
- Trūksta konkretumo ir vienareikšmiškumo

Programavimo kalbos

Programavimo kalbos skirtstomos į:

- Žemo lygio programavimo kalbos (Assembler)
- Aukšto lygio programavimo kalbos (C/C++, Java, C#, Delfi, Fortran, Pascal, Basic ir t.t.)

Žemo lygio programavimo kalbose naudojamos atitinkamos instrukcijos procesoriui, todėl jų tekstų ilgiai dideli, tačiau programų atlikimo laikas pakankamai mažas (pvz. įrenginių tvarkyklės).

Aukšto lygio programavimo kalbos naudoja papildomą programinio kodo "balastą", kuris didina vykdomojo failo dydį bei mažina programos atlikimo greitį.

Programavimo kalba C

C programavimo kalbą 1972 m. sukūrė Dennis Ritchie (Bell Lab.) Ją panaudojo UNIX OS kūrimui. Jis rėmėsi tokiomis programavimo kalbomis:

- ALGOL (1960)
- CPL (1963)
- B (1970)

C kalbos tęsinys objektiškai orientuotam programavimui C++ (B.Stroustrup, AT&T Bell Lab. ~1980 m.), C# (Microsoft 1997)

Standartai:

- *Tradicinis C* (B.Kernighan, D.Ritchie. *The C Programming Language*. 1978)
- *Standard C (1989)* **ANSI C** (ANSI X3.159-1989) ISO/IEC 9899:1990
- *Standard C (1995)*
- *Standrad C (1999)* ISO/IEC 9899:1999

C kalbos privalumai

C – šiuolaikinė programavimo kalba.

C – efektyvi programavimo kalba. Ji leidžia geriausiai išnaudoti kompiuterinius resursus.

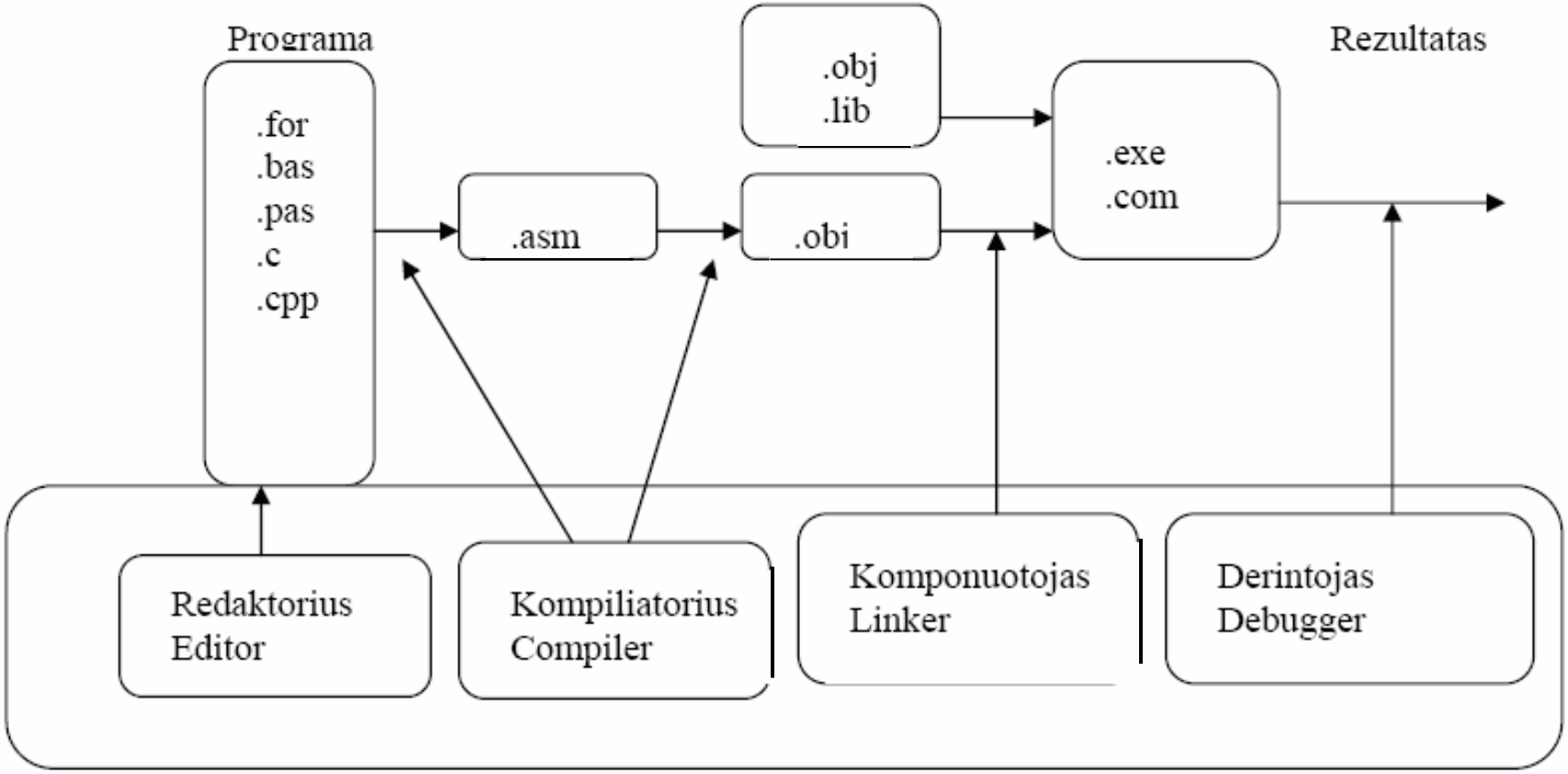
C kalba parašytos programos yra kompaktiškos ir greitai vykdomos.

C – mobilioji programavimo kalba. Tai reikškia, jei programa parašyta šia kalba, ji gali būti lengvai, su nedideliais pataisymais arba visai be jų, perkeliama į kitas skaičiavimo sistemas.

C – galinga ir lanksti programavimo kalba. Didelė dalis galingos UNIX operacinės sistemos parašyta C kalba. C kalba parašytos programos naudojamos matematiniams-fizikiniams ir techniniams uždaviniams spręsti, taip pat naudojama ir kompiuterinei grafikai kurti (*Cg* programavimo kalba NVidia grafiniams procesoriams).

C – turi galimybę panaudoti eilę valdančiųjų konstrukcijų, kurios paprastai asocijuojasi su asembleriu.

Vykdomosios programos generavimas



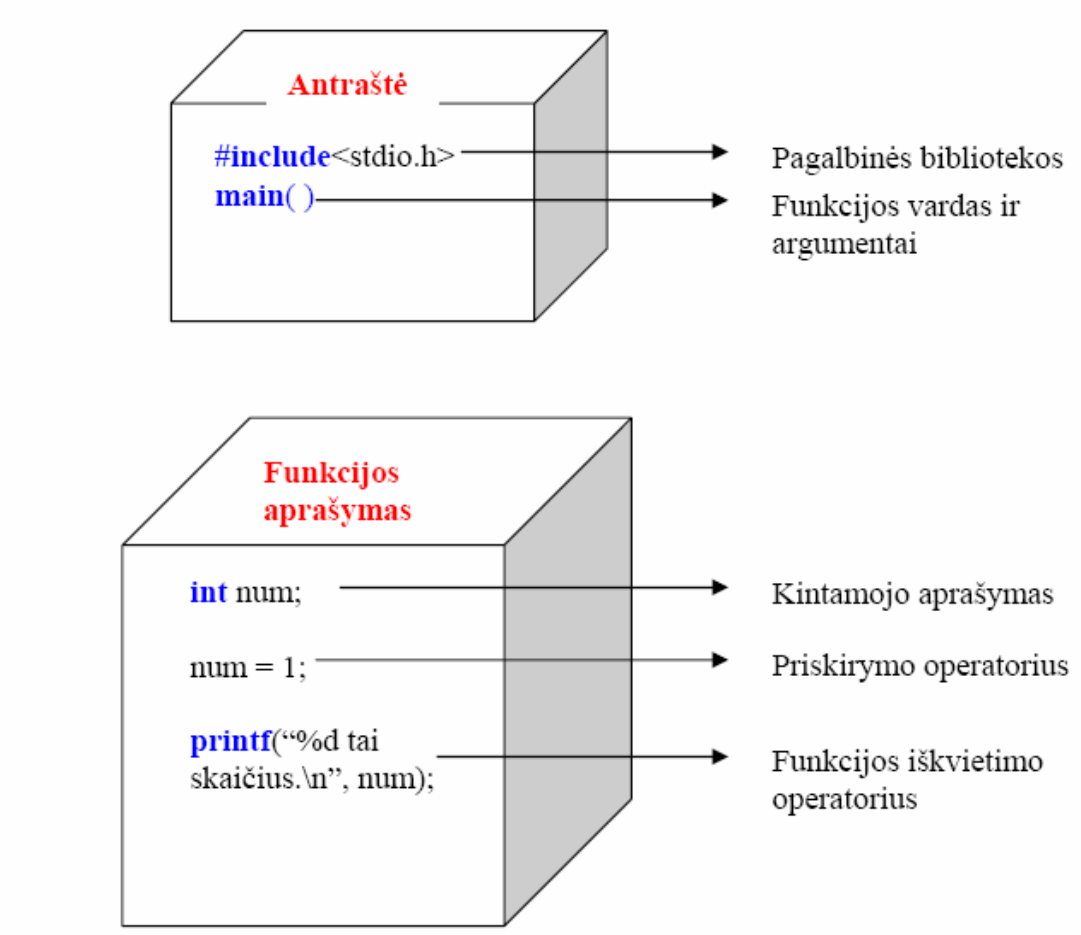
C programos struktūra

C kalbos programą sudaro:

1. Kompiliatoriaus instrukcijos – priešprocesoriaus direktyvos;
2. Struktūriniai blokai, kuriose saugomi duomenys bei atliekami veiksmai su duomenimis;

```
#include <stdio.h>
  int main ( )
{  int num;
        num = 1;
   printf (" Sveikas pasauli.... \n");
   printf (" %d - tai skaičius. \n", num);
  return 0;
}
```

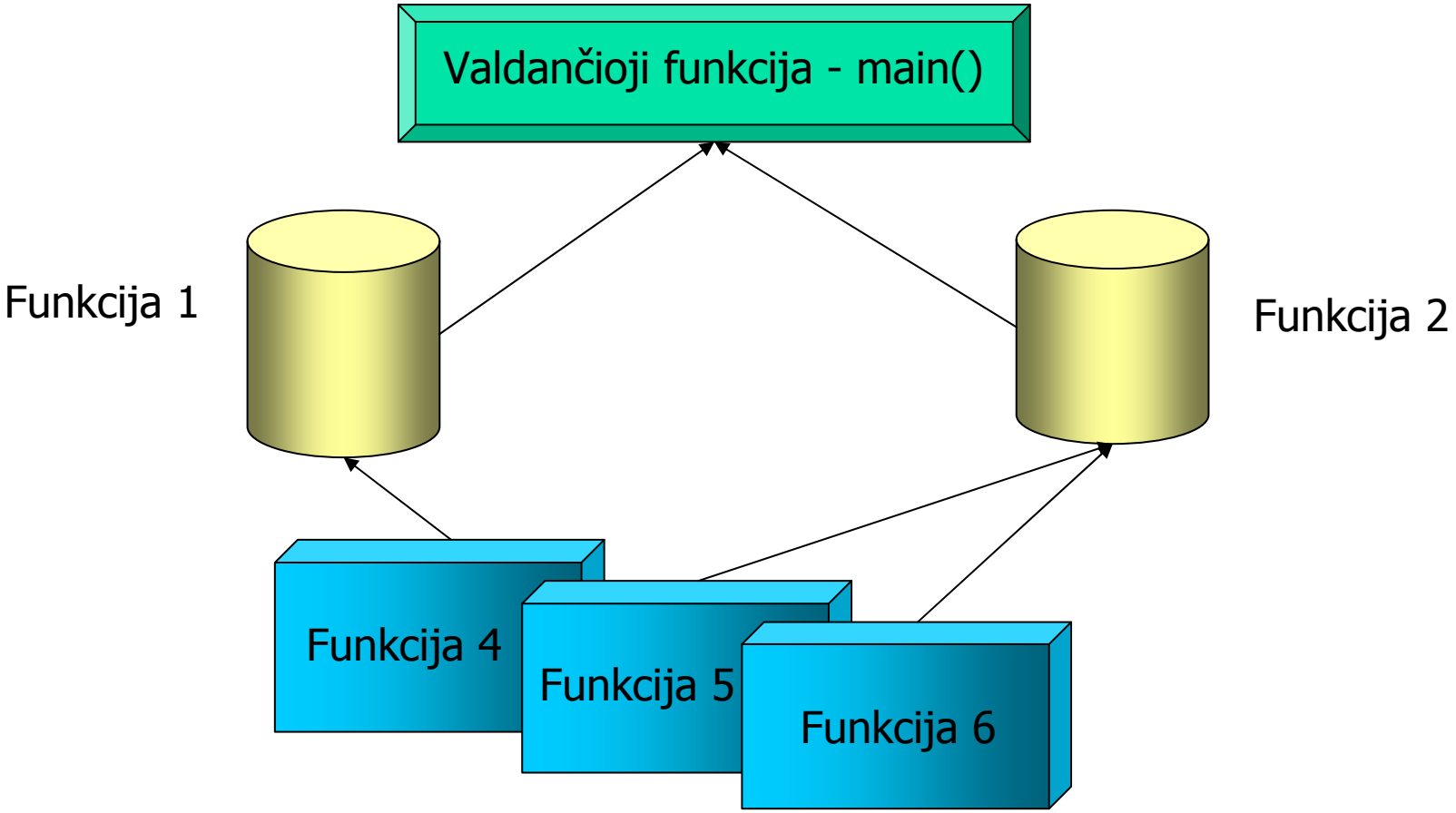
Programos struktūra



Funkcijos struktūros paprasčiausia schema

Sudėtingesnė programos struktūra

C programą paprastai sudaro *vienas ar daugiau* programinio kodo blokų, kuriuose talpinama tam tikra programos dalis - funkcijos.



Programos struktūra

Funkcijos – tai C programos struktūros pagrindas.

Funkcijų pagalba galime sudaryti aiškią programos struktūrą, apibrėžti ryšius tarp loginių programos blokų, todėl C programavimo kalba vadinama **struktūrine programavimo kalba**.

Paprastai, failą sudaro viena arba kelios funkcijos.

Nepriklausomai nuo funkcijų skaičiaus programoje, **visada privalo būti** pagrindinė (valdančioji) funkcija pavadinimu **main**( ).

Komentarai

Komentarai – tai programos dalis, kurią kompiliatorius ignoruoja.

Komentarams žymėti naudojami tokie simboliai:

// - komentaras prasideda nuo šio simbolio ir tęsiasi iki eilutės galo
/* - komentaro pradžia, */ - kometaro pabaiga

```
#include <stdio.h>
int main ( )
{   float ilgis, plotis, plotas;  // kintamųjų aprašymas
 /* kintamųjų reikšmių priskyrimas įvedant jas klaviatūra    */
     scanf ( "%f", &ilgis) ;
     scanf ( "%f", &plotis ) ;
     plotas = ilgis * plotis ;

/* kintamųjų reikšmių išvedimas  */
     printf ( “Stačiakampio plotas: %f metrų.\n", plotas) ;
     return 0;

}
```

Identifikatoriai

Darbui su duomenimis naudojami *kintamieji* ir *konstantos*.

Identifikatorius – tai kintamojo, konstantos ar funkcijos vardas.

Identifikatoriaus apribojimai:

- neturi viršyti *256 simbolių*.
- negali sutapti su vienu iš C/C++ raktiniais žodžiais t.y. :

asm, auto, bool, break, case, catch, char, class, const, const_cast, continue, default, delete, do, double, dynamic_cast, else, enum, explicit, export, extern, false, float, for, friend, goto, if, inline, int, long, mutable, namespace, new, operator, private,protected, public, register, reinterpret_cast, return, short, signed, sizeof, static, static_cast, struct, switch, template, this, throw, true, try, typedef, typeid, typename, union, unsigned, using, virtual, void, volatile, wchar_t, while

Identifikatoriai

Identifikatoriams naudojami simboliai:

- Didžiosios bei mažosios raidės,
- Skaičiai,
- _ (apatinis brūkšnys)

Identifikatoriaus pirmasis simbolis privalo būti raidė arba _

Teisingi vardai	**Klaidingi vardai**
Wiggly	$Z^**
cat1	1cat
Hot_Tub	Hot-Tub
_kcaB	don’t

Duomenų tipai

Raktiniai žodžiai: **int, char, float, double, long, short, unsigned**

Tipas	Dydis, B	Reikšmė
bool	1	true arba false
unsigned short int	2	0 … 65533
short int	2	-32768 … 32767
unsigned long int	4	0 … 4294967295
long int	4	-2147483648 … 2147483647
int (16 bit)	2	-32768 … 32767
int (32 bit)	4	-2147483648 … 2147483647
unsigned int (16 bit)	2	0 …65535
unsigned int (32 bit)	4	0 …4294697295
char	1	0 …256
float	4	3.4e-38 …3.4e+38 (7 skaičiai)
double	8	2.2e-308 … 1.8e+308 (15 skaitmenų)
void	2 ..4	

Pavyzdys

```
#include <stdio.h>
void main ( )
{
   int a;
   int b;
   int c, d;
   int  f = 1, h = 2;
   short int abc;
   float  tax, rate;
   float g; h;              // blogai
   bool check;
     check = false;
     g = 1.5e+21;
     rate = -2.8e-5;
     tax = rate * g;
}
```

Konstantos

Konstantos – tai duomenys, kurių reikšmė negali kisti programos vykdymo metu.

Konstantų tipai: *literalinės ir tipinės*.

Literalinės – tai simboliai, sveiki ir trupmeniniai skaičiai, eilutės.

Sveikų skaičių konstantos: dešimtainės, aštuntainės, šešioliktainės.

Tipinės konstantos apibrėžiamos raktiniu žodžiu const.

Pavyzdys

```
char string[ ] = "Mano batai buvo du";
const double pi =3.1415;
const int Radius = 3;
double plotas = 0;
plotas = pi * radius * radius;
```