C++ kalba: konkrečiai ir suprantamai

Parengė: Darius Bakšys
IT vyr. mokytojas
Vilniaus Žirmūnų gimnazija
dbaksys@gmail.com

Turinys

- Įvadas
- Programos struktūra
- Pirmoji programa
- Kintamieji ir konstantos
- Valdymo sakiniai
- Ciklai

MOKYKLŲ TOBULINIMO CENTRAS

Turinys (tęsinys)

- Masyvai
- Skaitymas/rašymas į failą
- Funkcijos
- Struktūros
- Klasės

MOKYKLŲ TOBULINIMO CENTRAS

Trumpa istorija

- Pirmosios programavimo kalbos C++ versijos buvo sukurtos tos pačios "Bell Labs" kompanijos darbuotojo Brajeno Struastrupo. Tai įvyko 1986 m. Į naują kalbą buvo įtraukta objektinio programavimo galimybė (žiūrėti OOP) bei buvo ištaisytos ankstesnės versijos klaidos. Pirmieji šios kalbos vartotojai buvo "Bell Labs" kompanijos darbuotojai, o pirmasis komercinis transliatorius buvo parašytas 1993 m.

MOKYKLŲ TOBULINIMO CENTRAS

Trumpa istorija

- Pirmuoju transliatoriumi tapo preprocesorius Croft, transliuojantis C++ kodą į alternatyvų jam C kodą. Kaip tik nuo tada atsirado knygų apie C++ ir jos greitai išpopuliarėjo. Dabar ši kalba skaitoma kaip viena svarbiausių kuriant didelius ir sudėtingus projektus.

MOKYKLŲ TOBULINIMO CENTRAS

C/C++ kompiliatoriai

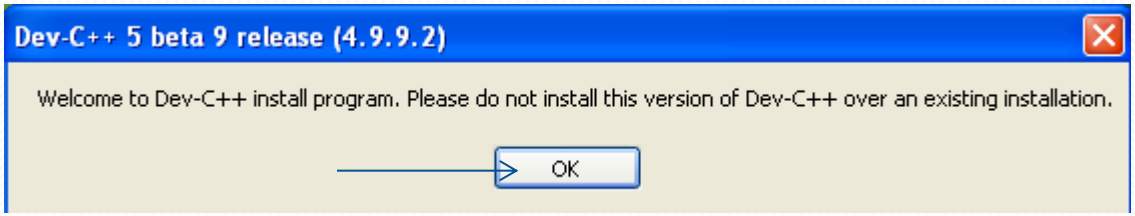
- Microsoft Visual C++
- Borland C++
- Dev C++
- Clang
- NetBeans
- CodeBlocks

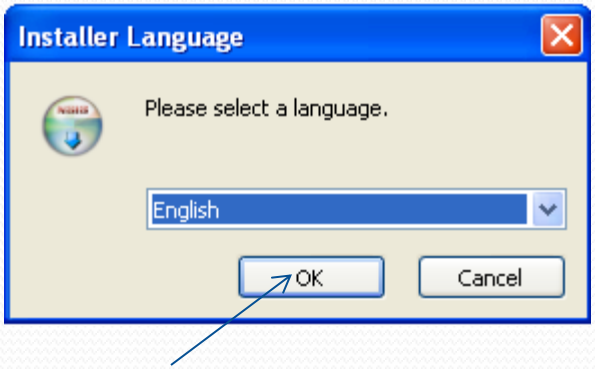
MOKYKLŲ TOBULINIMO CENTRAS

Tema. **DEV C++ aplinkos diegimas**

Programinės įrangos parsisiuntimas

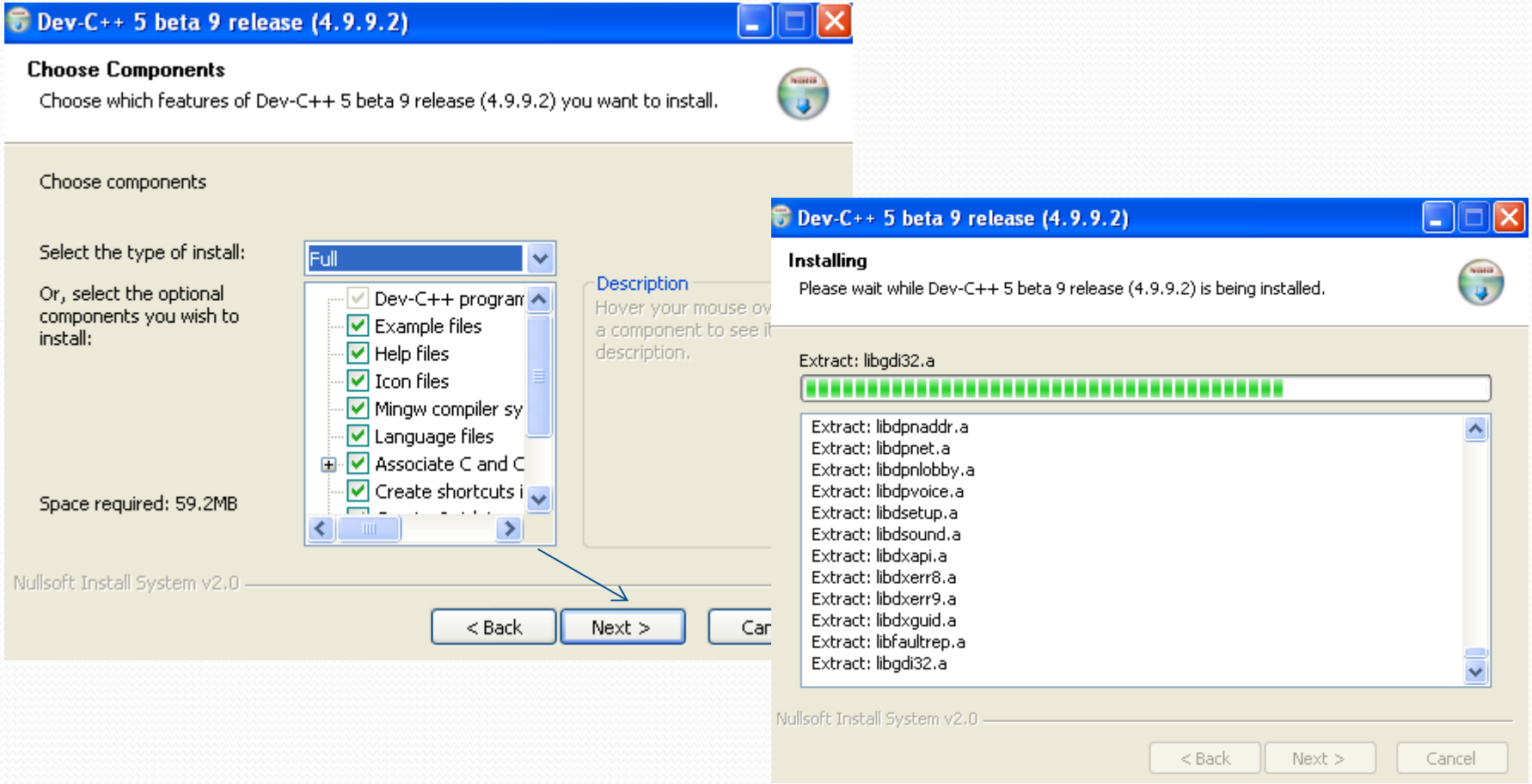
1. Parsisiunčiame DEV C++ spragtelėję nuorodą.
2. Vykdome diegimą:



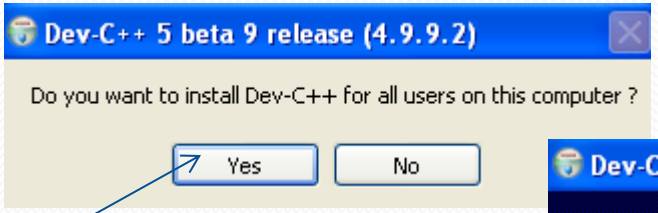


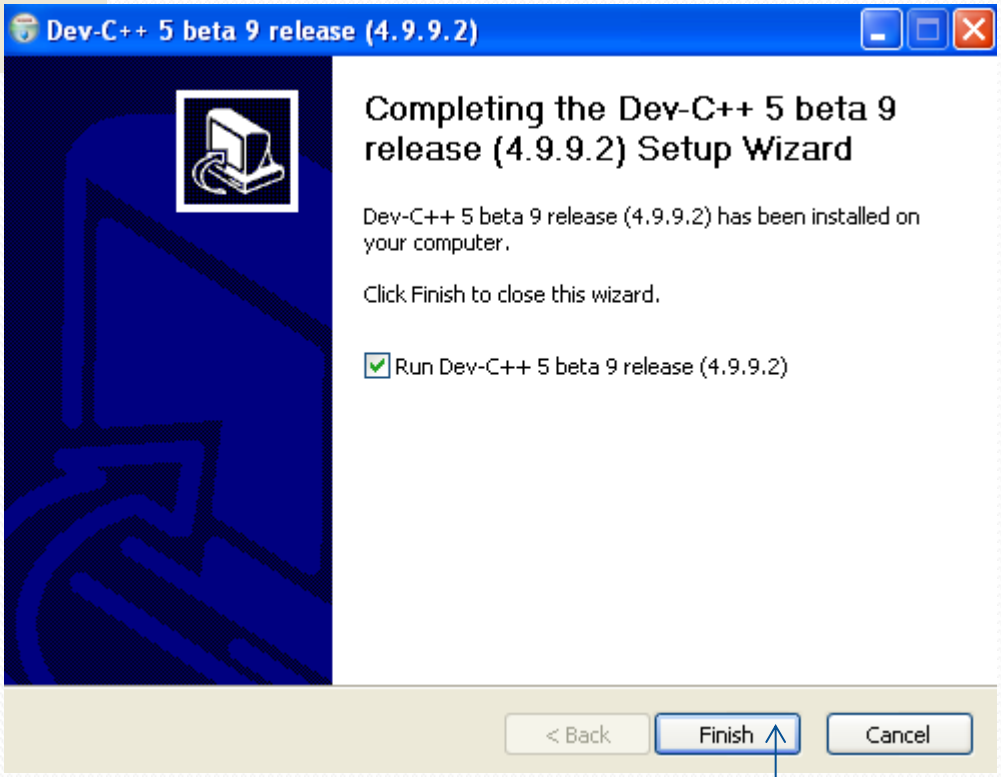
MOKYKLŲ TOBULINIMO CENTRAS

tęsinys

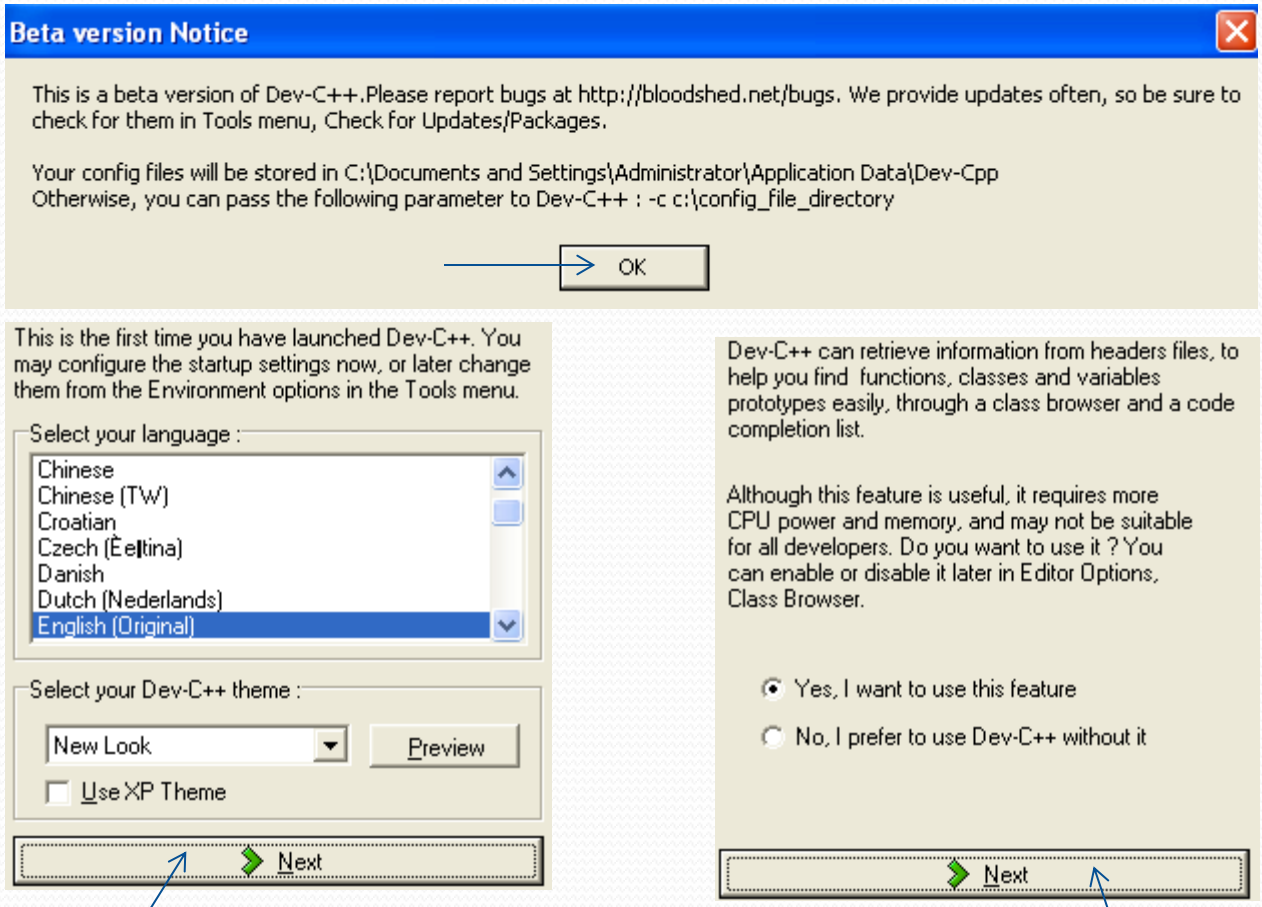


tęsinys


Dev-C++ 5 beta 9 release (4.9.9.2)
Do you want to install Dev-C++ for all users on this computer ?
Yes
No

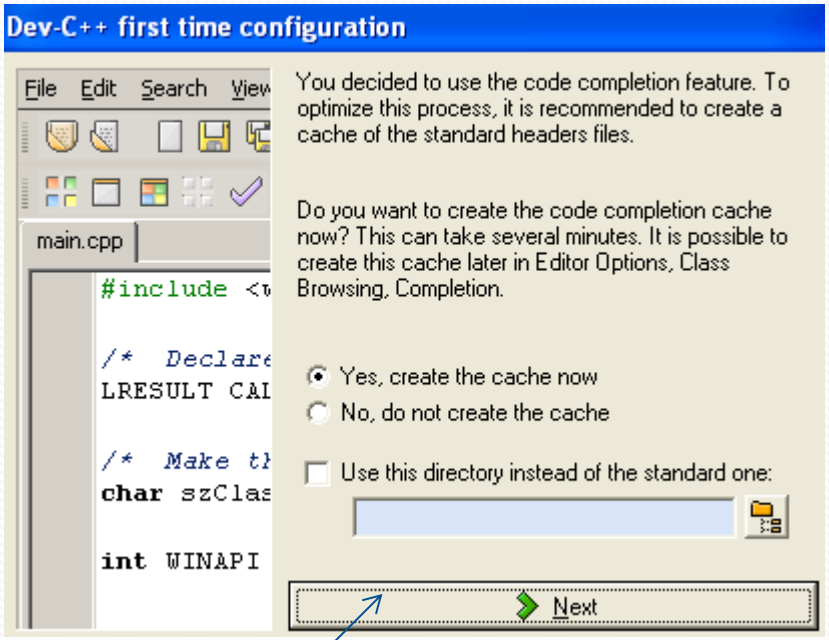

Dev-C++ 5 beta 9 release (4.9.9.2)
Completing the Dev-C++ 5 beta 9 release (4.9.9.2) Setup Wizard
Dev-C++ 5 beta 9 release (4.9.9.2) has been installed on your computer.
Click Finish to close this wizard.
Run Dev-C++ 5 beta 9 release (4.9.9.2)
< Back
Finish
Cancel

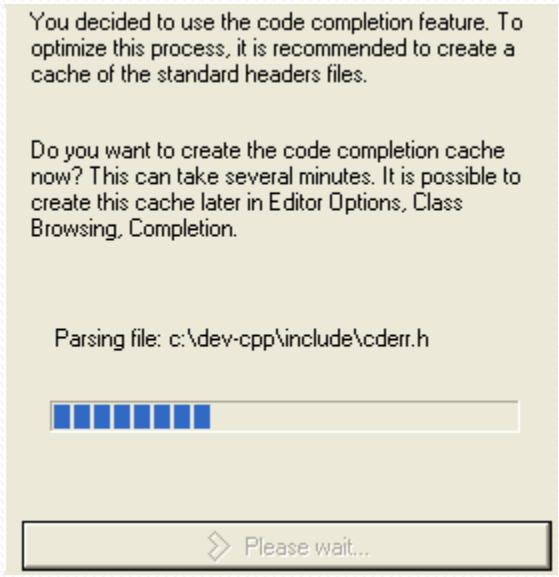
tęsinys

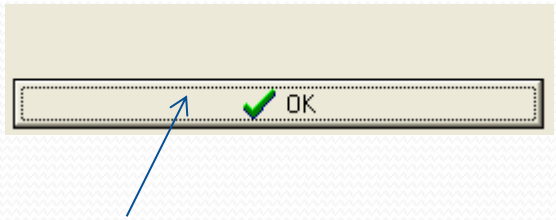

Beta version Notice
This is a beta version of Dev-C++.Please report bugs at http://bloodshed.net/bugs. We provide updates often, so be sure to check for them in Tools menu, Check for Updates/Packages.
Your config files will be stored in C:\Documents and Settings\Administrator\Application Data\Dev-Cpp
Otherwise, you can pass the following parameter to Dev-C++ : -c c:\config_file_directory
OK
This is the first time you have launched Dev-C++. You may configure the startup settings now, or later change them from the Environment options in the Tools menu.
Select your language :
Chinese
Chinese (TW)
Croatian
Czech (Čeština)
Danish
Dutch (Nederlands)
English (Original)
Select your Dev-C++ theme :
New Look
Preview
Use XP Theme
Next
Dev-C++ can retrieve information from headers files, to help you find functions, classes and variables prototypes easily, through a class browser and a code completion list.
Although this feature is useful, it requires more CPU power and memory, and may not be suitable for all developers. Do you want to use it ? You can enable or disable it later in Editor Options, Class Browser.
Yes, I want to use this feature
No, I prefer to use Dev-C++ without it
Next

MOKYKLŲ TOBULINIMO CENTRAS

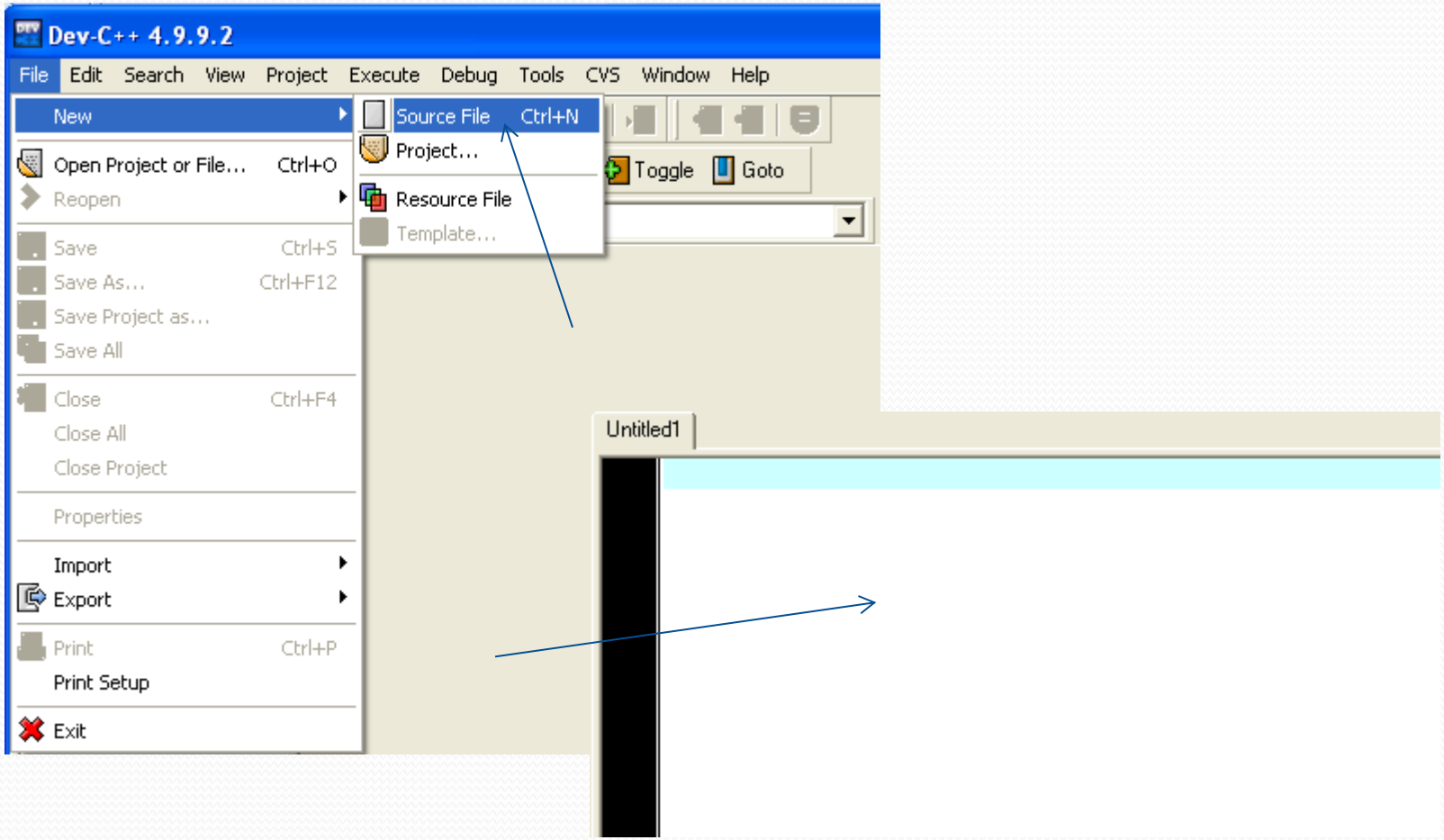
tęsinys



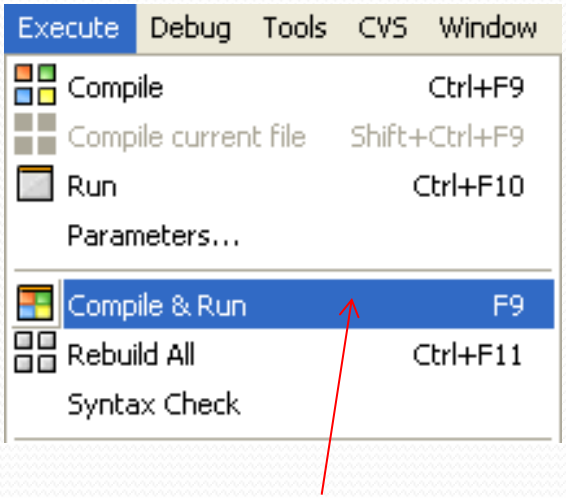


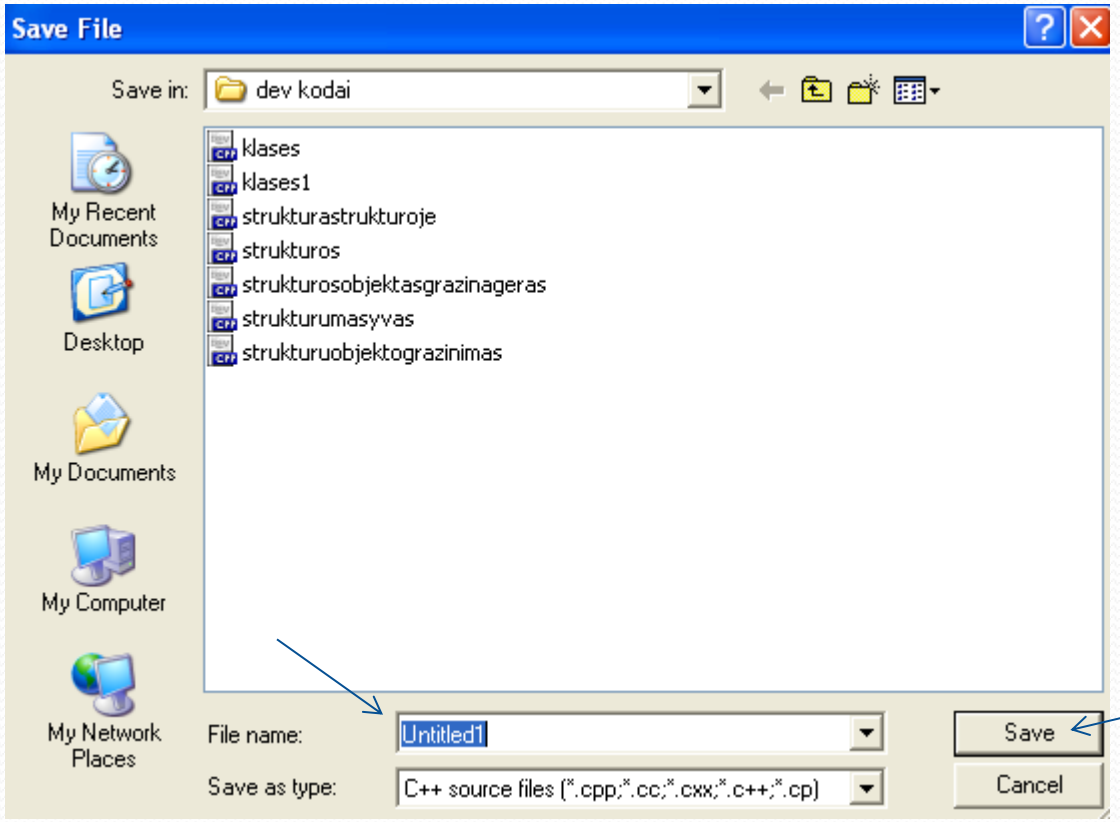


Naujo dokumento atvėrimas



Kodo kompiliavimas





MOKYKLŲ TOBULINIMO CENTRAS

Tema. **Aplinkos CodeBlocks diegimas**

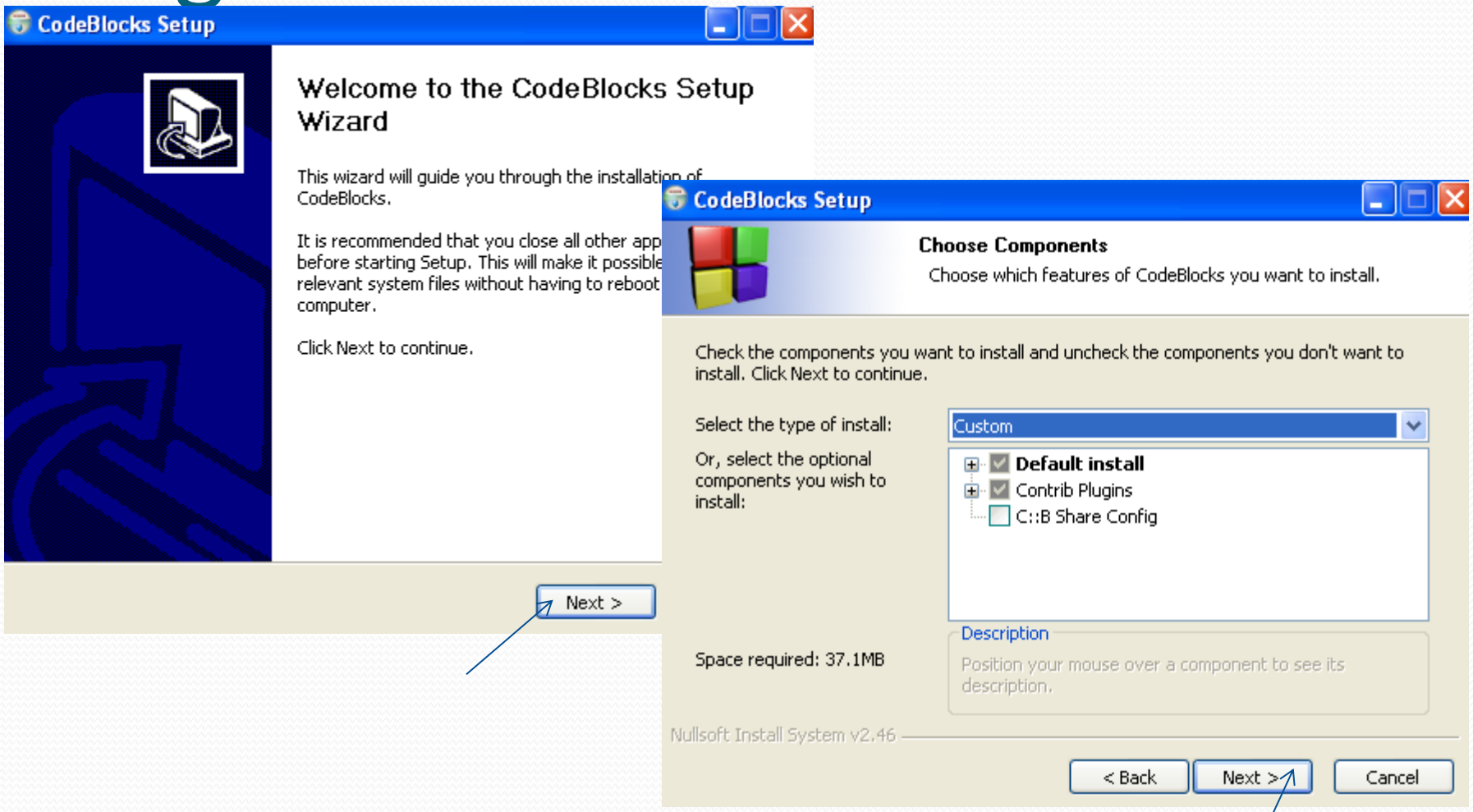
MOKYKLŲ TOBULINIMO CENTRAS

Parsisiuntimo nuorodos

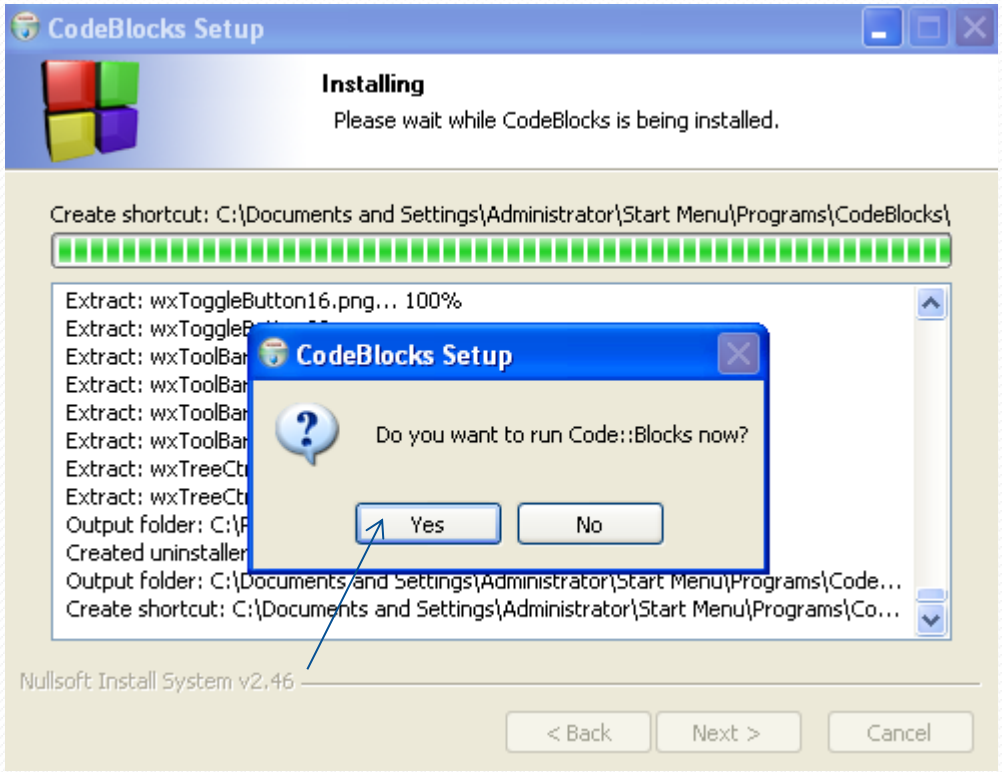
- Windows 2000/ XP, Vista/7 vartotojams
- **Linux 32-bit** vartotojams
- **Linux 64-bit** vartotojams

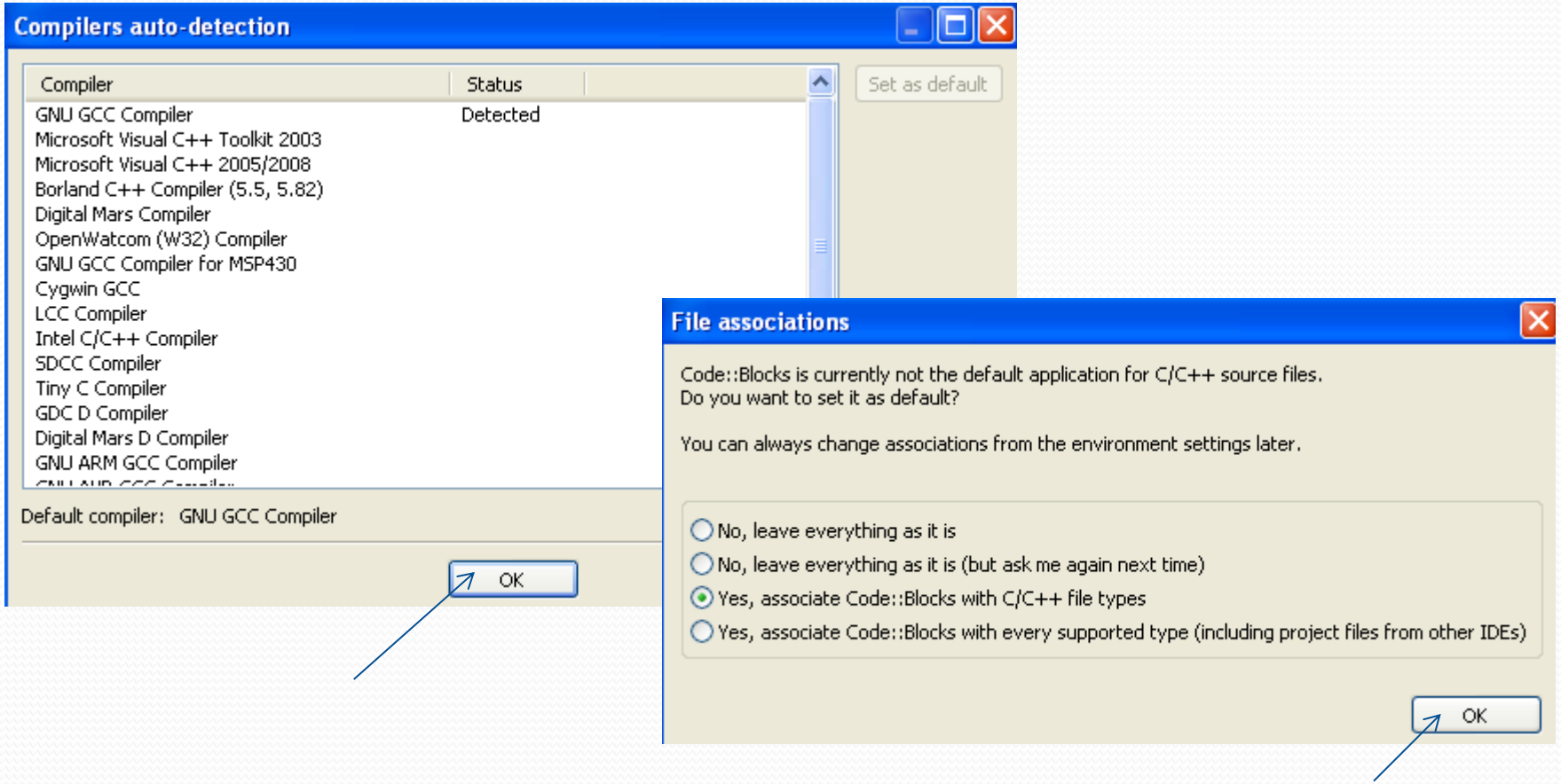
MOKYKLŲ TOBULINIMO CENTRAS

Diegimas

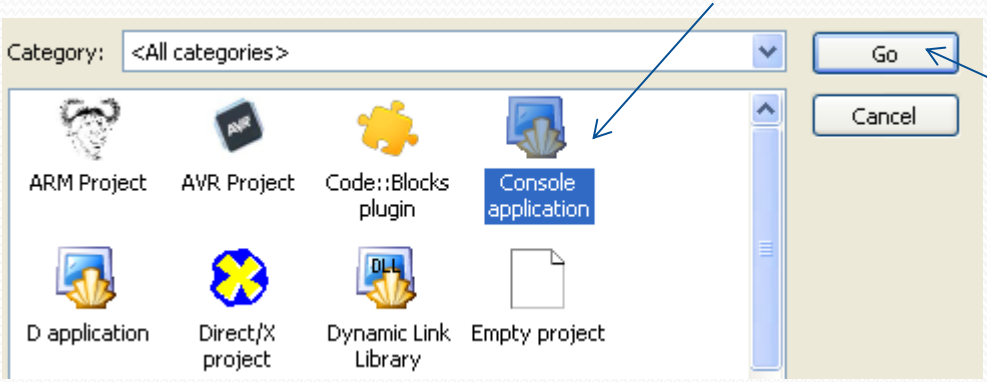


tęsinys

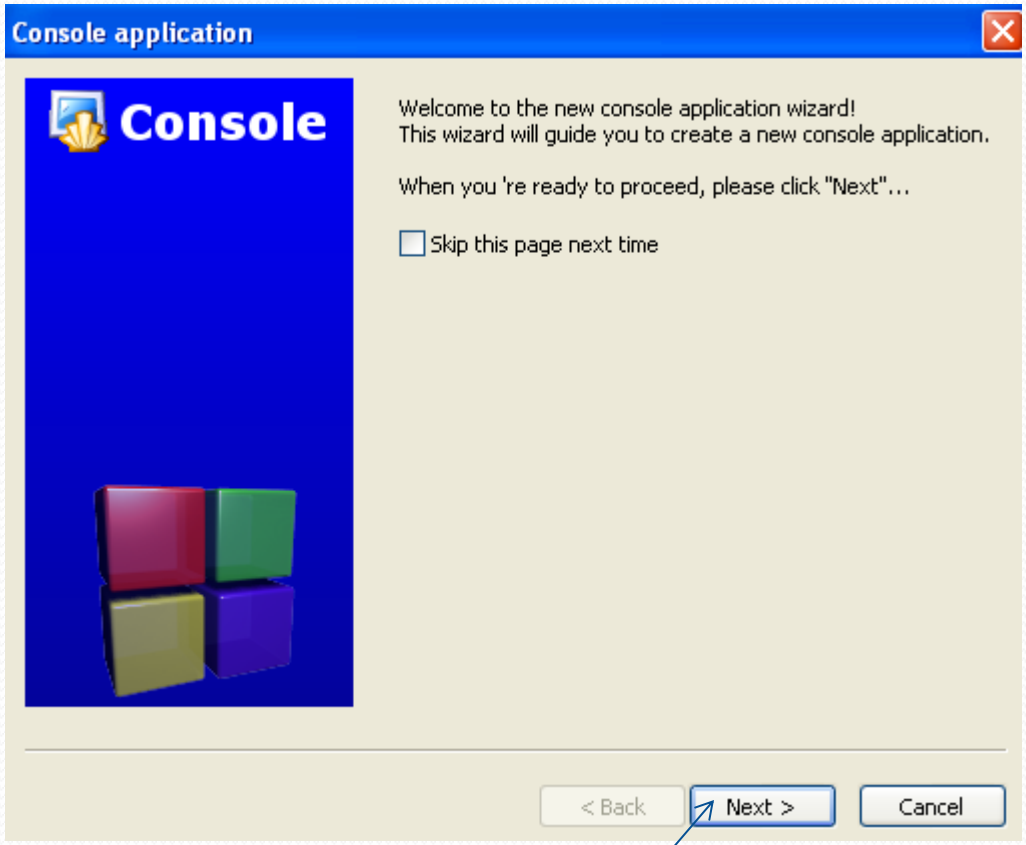

CodeBlocks Setup
Installing
Please wait while CodeBlocks is being installed.
Create shortcut: C:\Documents and Settings\Administrator\Start Menu\Programs\CodeBlocks\
Extract: wxToggleButton16.png... 100%
Extract: wxToolBar
Extract: wxToolBar
Extract: wxToolBar
Extract: wxToolBar
Extract: wxTreeCt
Extract: wxTreeCt
Output folder: C:\P
Created uninstaller
Output folder: C:\Documents and Settings\Administrator\Start Menu\Programs\Code...
Create shortcut: C:\Documents and Settings\Administrator\Start Menu\Programs\Co...
CodeBlocks Setup
Do you want to run Code::Blocks now?
Yes
No
Nullsoft Install System v2.46
< Back
Next >
Cancel

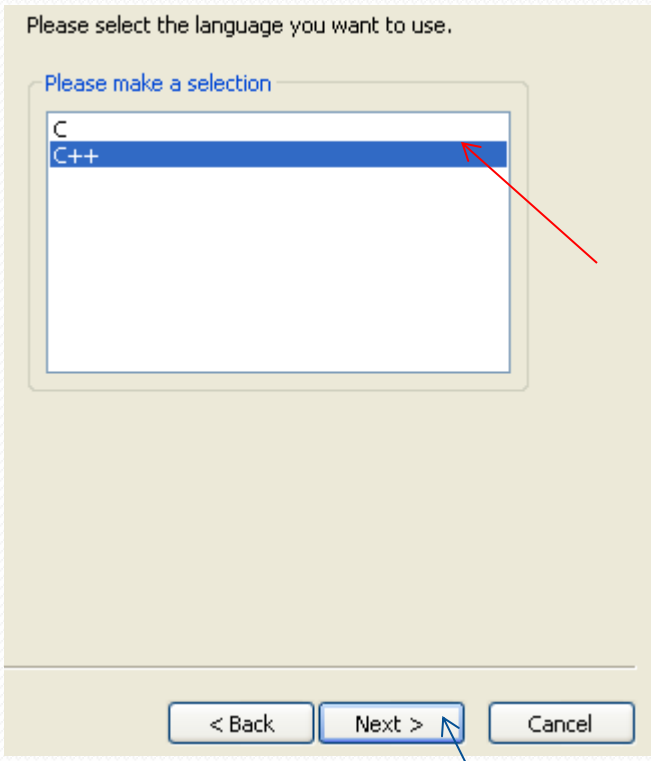

Compilers auto-detection
Compiler
Status
Set as default
GNU GCC Compiler
Detected
Microsoft Visual C++ Toolkit 2003
Microsoft Visual C++ 2005/2008
Borland C++ Compiler (5.5, 5.82)
Digital Mars Compiler
OpenWatcom (W32) Compiler
GNU GCC Compiler for MSP430
Cygwin GCC
LCC Compiler
Intel C/C++ Compiler
SDCC Compiler
Tiny C Compiler
GDC D Compiler
Digital Mars D Compiler
GNU ARM GCC Compiler
Default compiler: GNU GCC Compiler
OK
File associations
Code::Blocks is currently not the default application for C/C++ source files.
Do you want to set it as default?
You can always change associations from the environment settings later.
No, leave everything as it is
No, leave everything as it is (but ask me again next time)
Yes, associate Code::Blocks with C/C++ file types
Yes, associate Code::Blocks with every supported type (including project files from other IDEs)
OK

Naujo projekto kūrimas

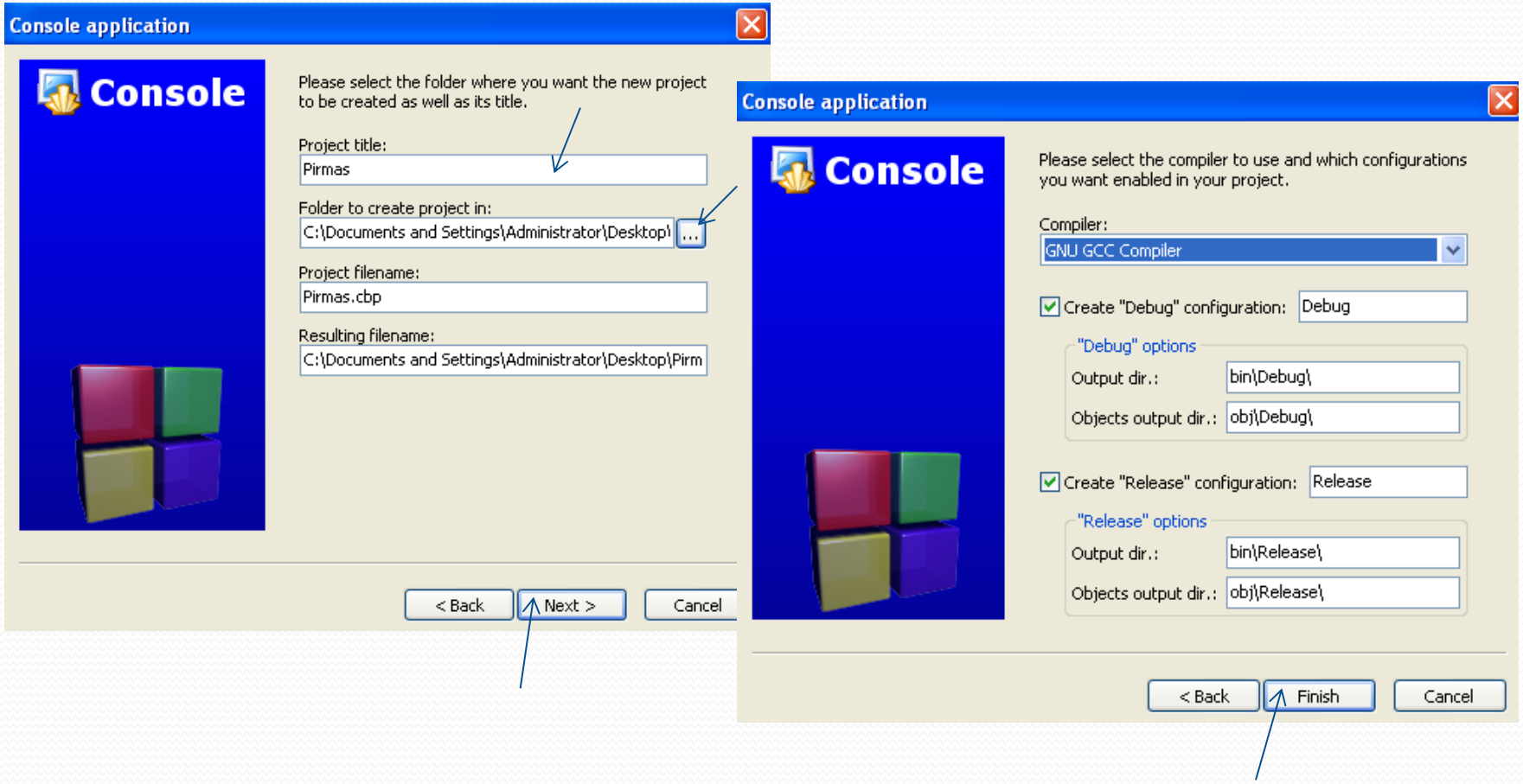


tęsinys

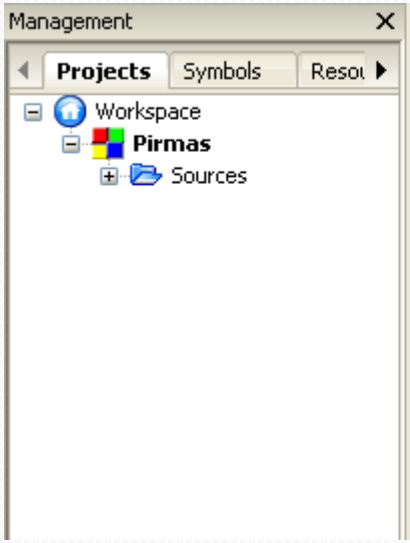

Console application
Console
Welcome to the new console application wizard!
This wizard will guide you to create a new console application.
When you 're ready to proceed, please click "Next"...
Skip this page next time
< Back
Next >
Cancel

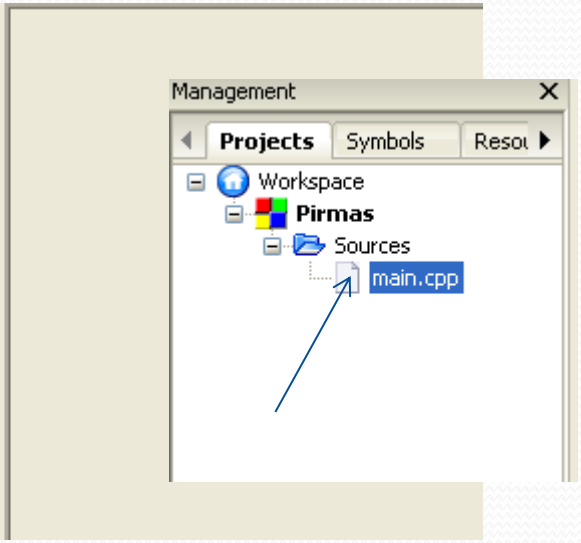

Please select the language you want to use.
Please make a selection
C
C++
< Back
Next >
Cancel

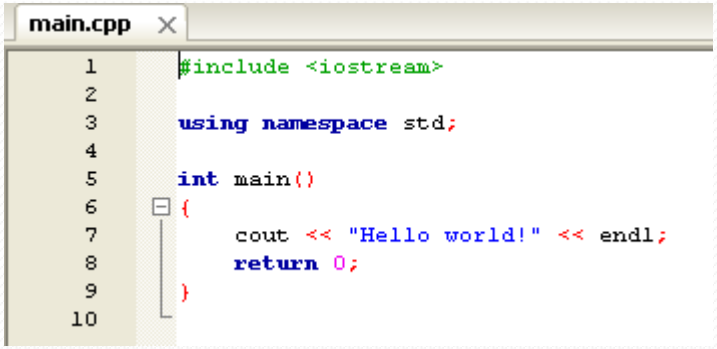
tęsinys

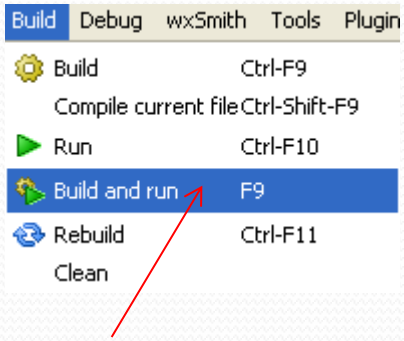

Console application
Console
Please select the folder where you want the new project to be created as well as its title.
Project title:
Pirmas
Folder to create project in:
C:\Documents and Settings\Administrator\Desktop\
...
Project filename:
Pirmas.cbp
Resulting filename:
C:\Documents and Settings\Administrator\Desktop\Pirm
< Back
Next >
Cancel
Console application
Console
Please select the compiler to use and which configurations you want enabled in your project.
Compiler:
GNU GCC Compiler
Create "Debug" configuration:
Debug
"Debug" options
Output dir.:
bin\Debug\
Objects output dir.:
obj\Debug\
Create "Release" configuration:
Release
"Release" options
Output dir.:
bin\Release\
Objects output dir.:
obj\Release\
< Back
Finish
Cancel

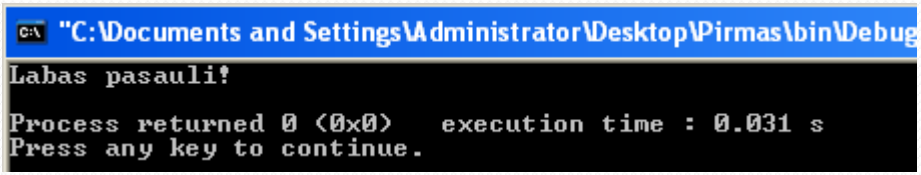
Programos langas dalis











MOKYKLŲ TOBULINIMO CENTRAS

Tema. **Programos struktūra**

MOKYKLŲ TOBULINIMO CENTRAS

Teoriniai pastebėjimai

- Programos sąvoką atitinka pagrindinė funkcija, kuri žymima vardu main;
- Identifikatoriuose didžiosios ir mažosios raidės nesutapatinamos;
- Aprašuose plačiai vartojami funkcijų prototipai;

MOKYKLŲ TOBULINIMO CENTRAS

tęsinys

- Programos objektų (struktūrų, kintamųjų, funkcijų) aprašai gali būti bet kurioje programos vietoje – svarbu tik tai, kad objektas būtų apibrėžtas prieš jį naudojant;
- C++ kalboje nedidelis standartizuotų operatorių skaičius, todėl naudojamos įvairios bibliotekų sistemos;

MOKYKLŲ TOBULINIMO CENTRAS

tęsinys

- Kreipiantis į kintamuosius ir struktūras, plačiai vartojamos rodyklės;
- Daugelį duomenų tipų galima interpretuoti keliais įvairiais būdais.

MOKYKLŲ TOBULINIMO CENTRAS

Tipinė programos C++ kalba struktūra

```
/* Instrukcijos pirminiam procesoriui */
/* Globaliniai aprašai */
/* Funkcijų prototipai */
main()
{
/* Pagrindinės funkcijos tekstas */
}
/* Funkcijų aprašai */
```

tęsinys

- komentarai – paaiškinimams skirtas simbolių rinkinys, kurio ribos žymimos simbolių poromis /* ir */; parašius eilutėje du simbolius be tarpo // toliau esantis tekstas iki eilutės pabaigos suprantamas kaip komentarai;

tęsinys

- main() – pagrindinės programos funkcijos antraštė;
- sudėtinis sakinys – tarp skliaustų {} įrašytas sakinių rinkinys, su kuriais programoje elgiamasi taip kaip su vienu sakiniu;

MOKYKLŲ TOBULINIMO CENTRAS

Programos struktūra

```
#include<iostream>     // bibliotekų įtraukimas
using namespace std;   // naudojama standartinė vardų
                          erdvė
int main (void)        // pagrindinė programos funkcija
{
        // čia rašomi veiksmai ir komandos
    return 0;                  // funkcijos grąžinama reikšmė
                                  OS
}
```

MOKYKLŲ TOBULINIMO CENTRAS

Komentaras

- C++ kode visų pirma yra įtraukiamos naudosimos bibliotekos, mūsų atveju *iostream*.
- *Programos langui užlaikyti naudojama* komandą *cin.get()*.
- *Namespace std* yra naudojama vardų srities įvesties ir išvesties srautams (*cin* ir *cout*) apibrėžti.

MOKYKLŲ TOBULINIMO CENTRAS

Komentaras

- Sukuriame pagrindinę funkciją *main()*. Prieš ją rašome žodelį *void*, kadangi šiuo atveju funkcija negražins jokių reikšmių.

Pastaba! Ne visi kompiliatoriai supranta *main()* *f*unkcijos rąžinamą *void* duomens tipą ir reikalauja *int*!

- Funkcijos turinį įrėminame laužtiniais skliaustais *{}*.

MOKYKLŲ TOBULINIMO CENTRAS

Tema. Informacijos išvedimas į ekraną

MOKYKLŲ TOBULINIMO CENTRAS

Išmoksime

1. Parengti programą, kuri ekrane rodys tekstinį užrašą “Labas, pasauli”.
2. Parengti programą, kuri ekrane rodys skaičius 1 2 3 4 atskirose eilutėse:

```
1
2
3
4
```

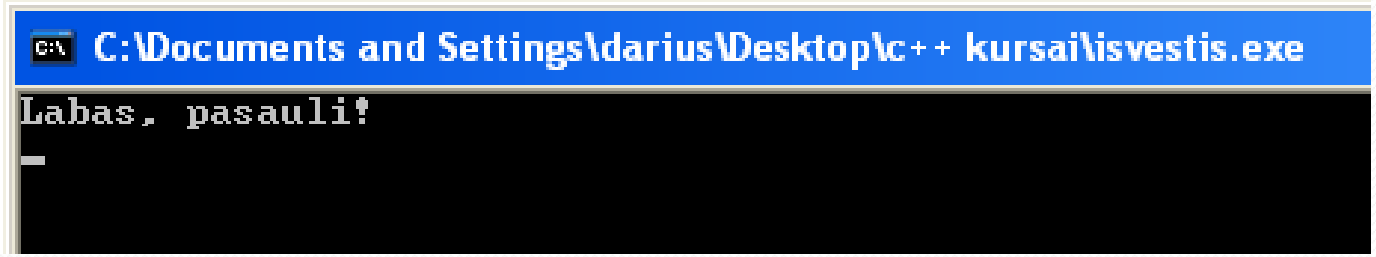
MOKYKLŲ TOBULINIMO CENTRAS

Informacijos išvedimas į ekraną

Funkcija `cout <<`

Teksto išvedimas į ekraną

```
int main (void)
{
cout << "Labas, pasauli!" << endl;
return 0;
}
```



Skaičiaus (int) išvedimas į ekraną

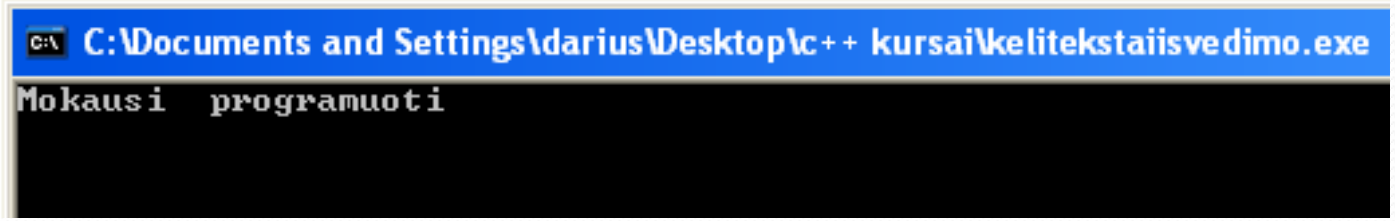
```
int main (void)
{
cout << 10 << endl;
return 0;
}
```

Skaičiaus (double) išvedimas į ekraną

```
int main (void)
{
cout << 2.57 << endl;
return 0;
}
```

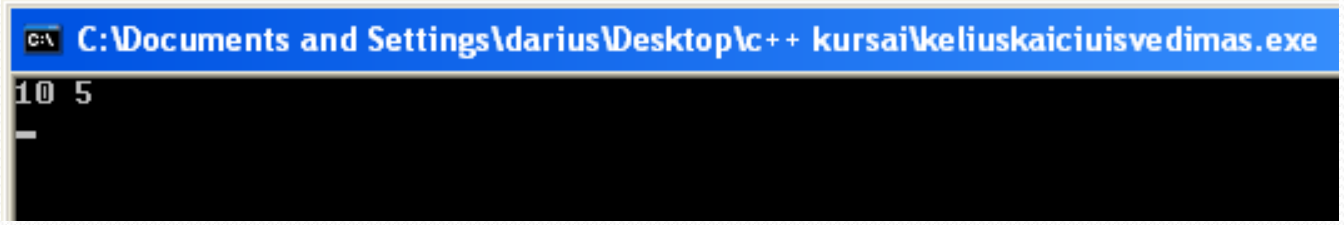
Kelių žodžių išvedimas į ekraną

```
int main (void)
{
cout << "Mokausi " << " programuoti" << endl;
return 0;
}
```



Kelių skaičių išvedimas į ekraną

```
int main (void)
{
cout << 10 << " " << 5 << endl;
return 0;
}
```



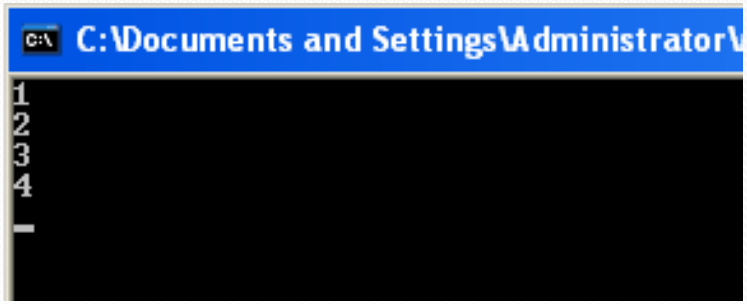
MOKYKLŲ TOBULINIMO CENTRAS

Užduoties darymas

2 užduotis

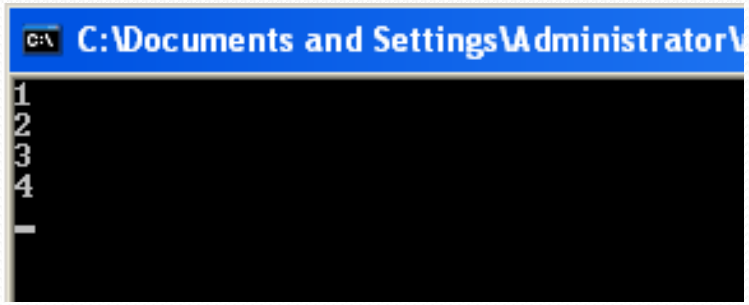
Informacijos išvedimas stulpeliu

```
int main (void)
{
cout << 1 << endl;
cout << 2 << endl;
cout << 3 << endl;
cout << 4 << endl;
return 0;
}
```



arba naudojant simbolį ‘\n’

```
int main (void)
{
cout << 1 << '\n' << 2 << '\n' << 3 << '\n' << 4
   << '\n';
return 0;
}
```



Teksto išvedimas keliomis eilutėmis

```
int main (void)
{
cout << “Pirmas\n” << “Antras\n”;
 cout << "Pirmas\nAntras";
return 0;
}
```

```
Pirmas
Antras
Pirmas
Antras
```

MOKYKLŲ TOBULINIMO CENTRAS

Tema. **Kintamieji**

MOKYKLŲ TOBULINIMO CENTRAS

Išmoksime

Parenkite programą, kuri ekrane rodytų apskaičiuoto reiškinio

(pirmas*antras) / (trecias-ketvirtas)

reikšmę.

MOKYKLŲ TOBULINIMO CENTRAS

Vardų sudarymo taisyklė

Kaip Paskalyje.

Kintamųjų vardų pavyzdžiai:

pirmas, Pirmas, skaicius1, galutinis_rezultatas

Negalima naudoti šių žodžių vardams sudaryti

Žodžiai, kurių negalima vartoti kintamųjų vardams

Nustatant kintamųjų vardus būtina žinoti, kad žemiau pateiktoje lentelėje išvardinti žodžiai yra C++ kalboje rezervuoti .

C++ raktiniai žodžiai

Asm	Auto	Break	Case	Catch	Char	Class	Const	Continue
Default	Delete	Do	Double	Else	Enum	Extern	Float	For
Friend	Goto	If	Inline	Int	Long	New	Operator	Private
Protected	Public	Register	Return	Short	Signed	Sizeof	Static	Struct
Switch	Template	This	Throw	Try	Typedef	Union	Unsigned	Virtual
Void	Volatile	While						

Kintamųjų tipai

- Int - natūralieji skaičiai.
- Float - realieji skaičiai.
- Double - realieji skaičiai.
- Bool – loginiai kintamieji.
- Char - simbolis.

MOKYKLŲ TOBULINIMO CENTRAS

Reikšmių diapazonas

Tipas	Saugoma reikšmė
Unsigned char	Reikšmių diapazonas nuo 0 iki 255.
Signed char	Reikšmių diapazonas nuo –128 iki 127.
Signed int	Reikšmių diapazonas nuo –32768 iki 32767
Unsigned int	Reikšmių diapazonas nuo 0 iki 65535
Unsigned long	Reikšmių diapazonas nuo 0 iki 4294967295
Signed long	Reikšmių diapazonas nuo -2147483648 iki 2147483647
Float	Reikšmių diapazonas nuo $-3.4*10^{-38}$ iki $3.4*10^{38}$
Double	Reikšmių diapazonas nuo $1.7*10^{-308}$ iki $1.7*10^{308}$

MOKYKLŲ TOBULINIMO CENTRAS

papildomai

Name	Description	Size*	Range*
`char`	Character or small integer.	1byte	signed: -128 to 127 unsigned: 0 to 255
`short int (short)`	Short Integer.	2bytes	signed: -32768 to 32767 unsigned: 0 to 65535
`int`	Integer.	4bytes	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
`long int (long)`	Long integer.	4bytes	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
`bool`	Boolean value. It can take one of two values: true or false.	1byte	`true` or `false`
`float`	Floating point number.	4bytes	+/- 3.4e +/- 38 (~7 digits)
`double`	Double precision floating point number.	8bytes	+/- 1.7e +/- 308 (~15 digits)
`long double`	Long double precision floating point number.	8bytes	+/- 1.7e +/- 308 (~15 digits)
`wchar_t`	Wide character.	2 *or* 4 bytes	1 wide character

MOKYKLŲ TOBULINIMO CENTRAS

Kintamųjų aprašymas

```
int main()
{
int pirmas;
int antras;
char simbolis;
unsigned int i;
short int i;
long l;
char vardas[10];
}
```

int pirmas, antras

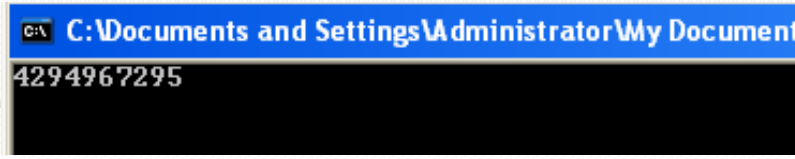
Reikšmių priskyrimas kintamiesiems

int pirmas = 10;

int b = 2; -> int b(2);

Priskiriant reikšmes, reikia žinoti reikšmių kitimo ribas:

unsigned int blogai=-1;



unsigned int gerai=10;

tęsinys

Pagal nutylėjimą realaus skaičiaus tipas yra double.

```
double realus1 = 51.52;
double realus2=6.02e23;
double realus3=1.6e-19;
```

Jei Jūs norite naudoti real arba long double reikia pridėti sufiksus f arba l:

```
float realus=10.1501f
long double realusilgas= 6.85159L
```

Simboliai ir eilutės

```
char raide='d';  // viengubos kabutės
char eilute[10]="c++";
```

```
// mano pirmoji eilutė
int main ()
{
  string eilute = "Mokausi programuoti C++ kalba";
  cout << eilute;
return 0;
}
```

string eilute (" Mokausi programuoti C++ kalba ");

Kintamųjų matomumas

```
#include <iostream>
using  namespace std;
int Integer;
char aSimbolis;
char  eilute[20];
int main()
{
float  Amzius;
cout << "Jusu amzius";
cin >> Amzius;
.....  }
```

// globalūs kintamieji

// lokalūs kintamieji

// instrukcijos (veiksmai)

Kiti specialūs simboliai

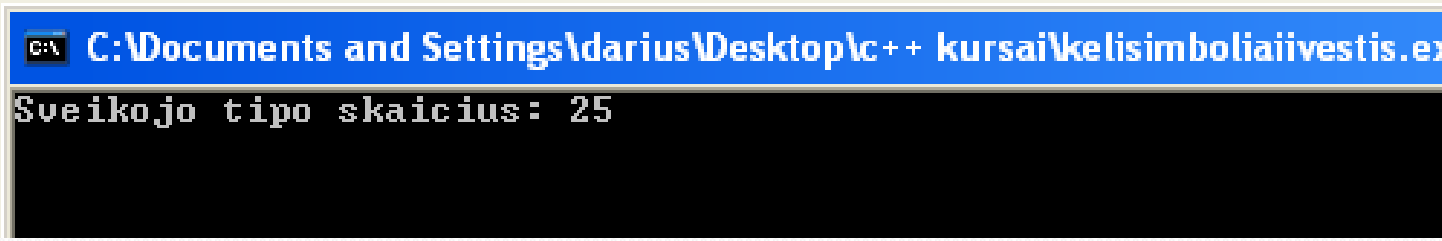
\n	newline
\r	carriage return
\t	tab
\v	vertical tab
\b	backspace
\f	form feed (page feed)
\a	alert (beep)
\'	single quote (')
\"	double quote (")
\?	question mark (?)
\\	backslash (\)

Simbolis	Paskirtis
\a	Signalinis (skambučio) simbolis
\b	Sugrąžinimo simbolis
\f	
\n	Naujos eilutės simbolis
\r	
\t	Horizontalios tabuliacijos simbolis
\v	Vertikalios tabuliacijos simbolis
\\	
\?	Klausimo ženklas
\’	Vienos kabutės
\’’	Dvigubos kabutės
\0	Nulinis simbolis
\ooo	Aštuonetainė reikšmė, pvz.\007
\xhhh	Šešioliktainė reikšmė,pvz\xFFFF

Kintamųjų tipų keitimas

Iš realaus į sveiko tipo skaičių

```
int main (void)
{
float realus=25.5;
cout << "Sveikojo tipo skaicius: " << int(realus)
   << endl;
return 0;
}
```



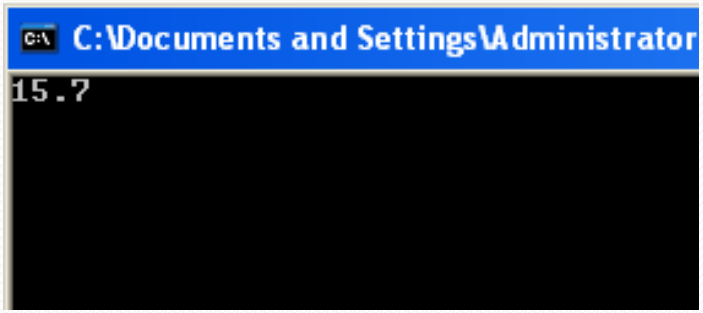
Kodas float skaičių 25.5 konvertuoja į sveiko tipo skaičių 25.

tęsinys

```
int i;
float f = 3.14;
i = (int) f;
```

Konstantų naudojimas

```
#define PI 3.14
int main (void)
{
const int A = 125;
  float rezultatas;
  rezultatas=5*PI;
 cout << rezultatas;
 cin.get();
 cin.get();
 return 0;
}
```



Uždavinio darymas


```
#include <iomanip>   // s
int main(void)
{  int pirmas=5,
   antras=10,trecias=11,ketvirtas=8;
   double rez;
   rez=(double)(pirmas*antras) / (trecias-
   ketvirtas);
   cout << setprecision(4) << rez << endl;
   return 0;  }
```

MOKYKLŲ TOBULINIMO CENTRAS

Tema. **Informacijos skaitymas iš klaviatūros**

MOKYKLŲ TOBULINIMO CENTRAS

Funkcija cin>>

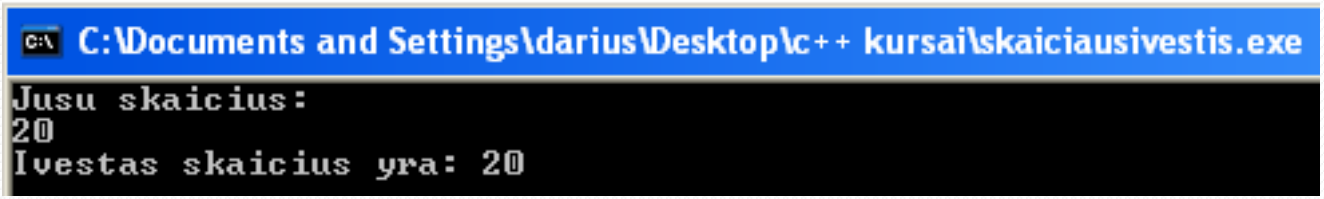
MOKYKLŲ TOBULINIMO CENTRAS

Užduotis

- Parenkite programą, kuri nuskaitytų iš klaviatūros kintamųjų *pirmas*, *antras*, *trecias* ir *ketvirtas* reikšmes ir apskaičiuotų reiškinį (pirmas*antras) / (trecias-ketvirtas)

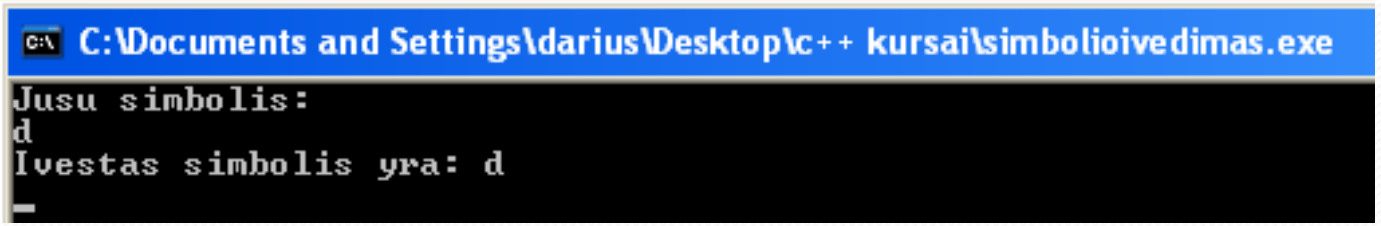
Skaičiaus skaitymas iš klaviatūros

```
int main (void)
{
int skaicius = 0;
cout << "Jusu skaicius: " << endl;
cin >> skaicius;
cout << "Ivestas skaicius yra: " << skaicius << endl;
return 0;
}
```



Simbolio skaitymas iš klaviatūros

```
int main (void)
{
   char  sim;
   cout << "Jusu simbolis: " << endl;
   cin >> sim;
   cout << "Ivestas simbolis yra: " << sim << endl;
   return 0;
}
```



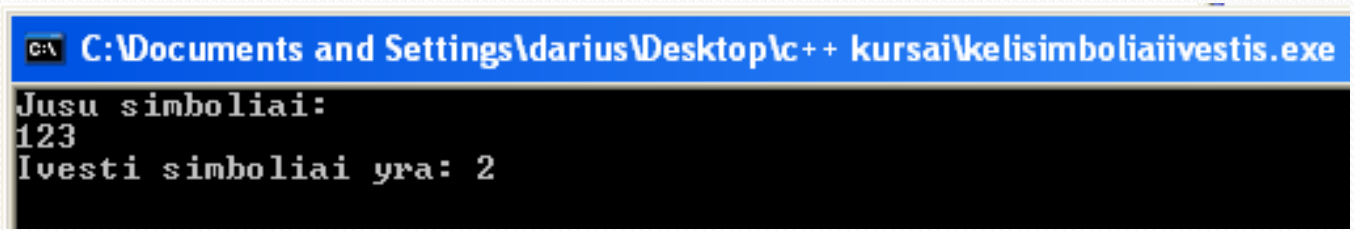
MOKYKLŲ TOBULINIMO CENTRAS

Kelių skaičių skaitymas iš klaviatūros

```
int main (void)
{
  int pirmas, antras;
  cout << "Jusu skaiciai: " << endl;
  cin >> pirmas  >> antras;
  cout << "Ivesti skaiciai yra: " << pirmas << “  ”<<
   antras <<   endl;
  return 0;
}
```

Kelių simbolių skaitymas iš klaviatūros

```
int main (void)
{
   char keli[3];
   cout << "Jusu simboliai: " << endl;
   cin >> keli;
   cout << "Ivesti simboliai yra: " << keli[1] <<
   endl;
   return 0;
}
```



MOKYKLŲ TOBULINIMO CENTRAS

Eilutė su cin

cin>> skaito iki pirmo tarpo simbolio, kad nuskaitytų tekstą iš kelių žodžių reikia naudoti funkciją getline().

tęsinys

```
int main ()
{   string eilute;
    cout << "Koks yra Jusu megstamos masinos
   modelis: ";
    getline (cin, eilute);
    cout << "Man patinka: " << eilute << ".\n";
    return 0;
}
```

```
Koks yra Jusu megstamos masinos modelis: BMW X6
Man patinka: BMW X6.
```

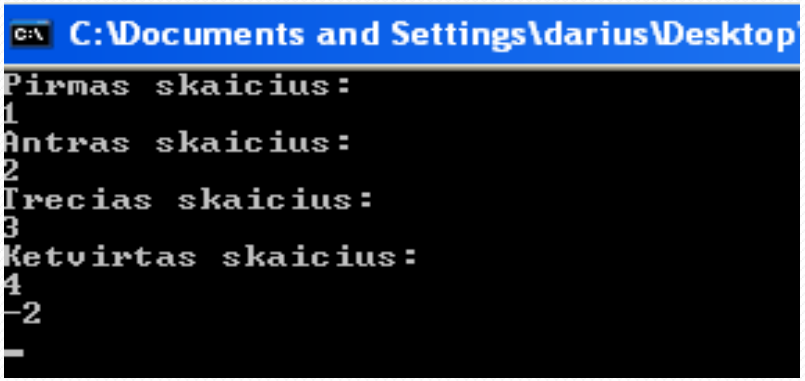
MOKYKLŲ TOBULINIMO CENTRAS

Uždavinio darymas

```
int main (void)
{
int pirmas,antras,trecias,ketvirtas;
double rez;
cout << "Pirmas skaicius: " << endl;
cin >> pirmas;
cout << "Antras skaicius: " << endl;
cin >> antras;
```

tęsinys

```
cout << "Trecias skaicius: " << endl;
cin >> trecias;
cout << "Ketvirtas skaicius: " << endl;
cin >> ketvirtas;
rez=(double)(pirmas*antras) / (trecias-ketvirtas);
cout << rez << endl;
cin.get();
cin.get();
return 0;
}
```



MOKYKLŲ TOBULINIMO CENTRAS

Tema. **Pagrindinės matematinės operacijos**

MOKYKLŲ TOBULINIMO CENTRAS

Aritmetinės operacijos

+ - sudėtis

- - atimtis

* - daugyba

/ - dalyba (sveikosios dalies gavimas)

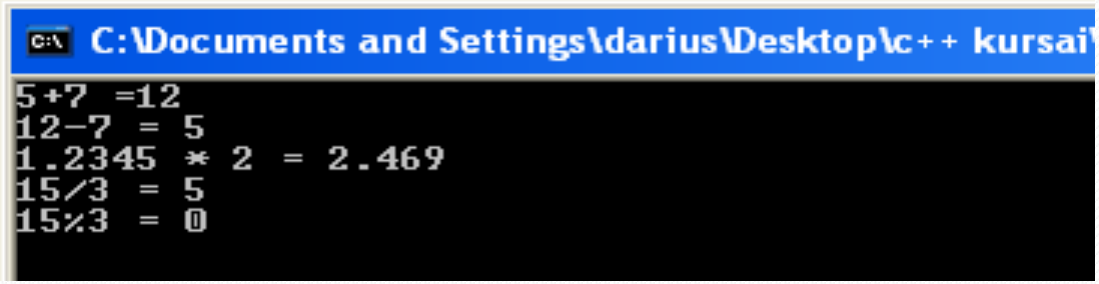
% - dalybos modulis (liekanos nustatymas)

++ - inkrementas

-- - dekrementas

MOKYKLŲ TOBULINIMO CENTRAS

Pavyzdys




```
int main(void)
{
cout << "5+7 =" << 5+7 << endl;
cout << "12-7 = " << 12-7 << endl;
cout << "1.2345 * 2 = " << 1.2345 * 2 << endl;
cout << "15/3 = " << 15/3 << endl;
cout << "15 %3 = " << 15%3 << endl;
return 0;
}
```

Papildomai

- Apskaičiuoti reiškinius:

```
2 + 34 / 3 * (4 % 5);
1 + 59 % 5 + 7 / 4;
5 * (4 / 3 * (5 % 7 * 2));
1 + 59 % (5+7) / 4;
9 / 4 * 3 + 34 % 3 * 4 % 5;
1 + 59 % 5 + 7 / 4;
```

MOKYKLŲ TOBULINIMO CENTRAS

Didinimo vienetu komanda (++)

kiekis ++; atitinka kiekis= kiekis + 1

MOKYKLŲ TOBULINIMO CENTRAS

Teoriniai pastebėjimai

- Programos gali patalpinti didinimo operatorių ++ iki arba po kintamojo: ++variable; variable++;
- Kai operatorius ++ randasi prieš kintamąjį , tai jis yra vadinamas priešdidinimo operatoriumi.
- Antras operatorius ++ randasi po kintamojo ir yra vadinamas podidinimo operatoriumi. C++ kalba juos supranta skirtingai.

MOKYKLŲ TOBULINIMO CENTRAS

Aiškinimas

1 pavyzdys

```
B = 10;
A = B++;
// A = 10, B = 11;
```

Kintamojo B reikšmė priskiriama kintamąjam A, tada B reikšmė padidinama

Aiškinimas

2 pavyzdys

```
B = 11;
A = ++ B;    //  A = 12, B = 12
```

B reikšmė padidinama, ir gauta reikšmė priskiriama kintamąjam A.

MOKYKLŲ TOBULINIMO CENTRAS

Mažinimo vienetu operatorius

- Operatorius -- reiškia mažinimą vienetu. Kaip ir padidinimo operacijose C++ kalba turi prieš- ir pomažinimą.

MOKYKLŲ TOBULINIMO CENTRAS

Sudėtinės operacijos

Operatorius		Pavyzdžiai ir paaiškinimai
++	k++ ++k	k reikšmė panaudojama, po to didinama vienetu. k reikšmė didinama vienetu, po to panaudojama.
−−	k−− −−k	k reikšmė panaudojama, po to mažinama vienetu. k reikšmė mažinama vienetu, po to panaudojama.
+=	`s += k;`	`s = s + k;`
−=	`s −= k;`	`s = s − k;`

Lyginimo ir loginės operacijos

Lyginimo operacijos

== - ar lygios dvi reikšmės?
> - ar pirma didesnė už antrą?
< - ar pirma mažesnė už antrą?
>= - ar pirma nemažesnė už antrą?
<= - ar pirma nedidesnė už antrą?
!= - ar nelygios dvi reikšmės?

Loginės operacijos

|| - loginė sudėtis arba (or)
! – loginis neigimas ne (not)
&& - loginė daugyba ir (and)

Loginės operacijos

Jei kairė pusė False– rezultatas False

```
((10 == 7) && (-1 > 2))  // false ( false && false ).
```

Jei kairė pusė True– rezultatas True

```
((1 == 8) || (10 > 8))      // true ( false || true ).
```

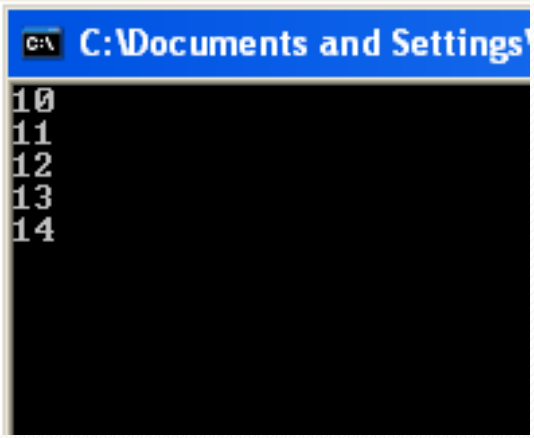
Kablelio operatorius (,)

- Naudojamas atskirti du arba daugiau išraiškų, kai viena išraiška nekinta.

```
d = (c=10, c+1);
// c=10, d = 11
```

Panaudojimo pavyzdys

```
int main(void)
{
int d, c;
 for (int i=0; i<5; i++)
{
d = (c=10, c+i);
 cout <<d<<'\n';
 }
return 0;
}
```



MOKYKLŲ TOBULINIMO CENTRAS

Operacijų vykdymo tvarka

- Kad išvengti painiavos, C++ kalboje kiekvienai operacijai suteikiamas prioritetas, nustatantis veiksmų eilę. Kadangi C++ vykdo aritmetines operacijas tam tikra tvarka, tai ir jūsų programos atitinkama tvarka vykdys veiksmus.
- Sekančioje skaidrėje pateikta C++ operacijų vykdymo tvarkos lentelė.

Operacija	Apibūdinimas	Pavyzdys
::	Kintamojo apibrėžimo sritis	class_name::class_member_name::variable_name
::	Globalaus kintamojo apibrėžimas	::variable_name
.	Elemento išrinkimas	object.member_name
->	Elemento išrinkimas	pointer-member_name
[]	Indeksacija	pointer[element]
()	Funkcijos kvietimas	expression(parameters)
()	Reiškinio rezultatas	type(parameters)
Sizeof	Objekto dydis	sizeof expression
Sizeof	Tipo dydis	sizeof(type)
++	Inkrementavimas po	variable++
++	Inkrementavimas prieš	++variable
--	Dekrementavimas po	variable--
--	Dekrementavimas prieš	--variable
&	Objekto adresas	&variable
*	Pervardinimas	*pointer
New	Sukūrimas (išdėstymas)	new type
Delete	Panaikinimas (ištrynimas)	delete pointer
delete[]	Masyvo panaikinimas	delete pointer
~	Papildymas	~expression
!	Inversija	!expression
+	Pridėti	+1
-	Atimti	-1
.*	Elemento išrinkimas	object.*pointer
->*	Elemento išrinkimas	object->*pointer
*	Daugyba	expression*expression
/	Dalyba	expression/expression
%	Liekana po dalybos	expression%expression
+	Sudėtis	expression+expression
-	Atimtis	expression-expression

MOKYKLŲ TOBULINIMO CENTRAS

Tema. Valdymo sakinys IF

MOKYKLŲ TOBULINIMO CENTRAS

Teoriniai pastebėjimai

Programa yra komandų seka. Visos anksčiau parašytos C++ programos vykdo komandas nuosekliai. Bet gali tekti vykdyti atskirą komandų grupę, esant patenkintai tam tikrai sąlygai, ir priešingai - kai sąlyga nepatenkinta, reikia vykdyti kitą komandų grupę. Kitaip tariant reikia, kad programa darytų sprendimus ir atitinkamai į juos reaguotų. Čia aprašomas operatorius *if*.

tęsinys

- Programos naudoja operatorių *if – else* vienos operatorių grupės vykdymui, jei sąlyga patenkinta, ir kitos, jeigu sąlyga nepatenkinta;
- Kombinuojant kelis operatorius *if – else*, galima tikrinti kelias sąlygas
- Naudojant loginius C++ operatorius IR ir ARBA, galima tikrinti kelias sudėtines sąlygas.

Operatorius IF (vienas veiksmas)

if (salyga)

veiksmas;

```
if (a == 150)
cout << "a lygu 150";
```

Operatorius IF (keli veiksmai)

```
if (salyga)
{
veiksmas1;
.
veiksmas n;
}
```

```
if (a == 150)
{
cout << "a lygu ";
cout << a;
}
```

MOKYKLŲ TOBULINIMO CENTRAS

Operatorius IF-ELSE

- Junginio IF-ELSE pagalba yra aprašomi du veiksmų blokai – vienas iškart po operatoriaus if, kitas iškart po operatoriaus else.

MOKYKLŲ TOBULINIMO CENTRAS

IF-ELSE (vienas veiksmas)

```
if (sąlyga)
 veiksmas;
else
veiksmas;
```

```
if (a == 110)
cout << "x lygu 110";
else
cout << "x nelygu 110";
```

Sąlygos operatorius (?)

```
int main ()
{
int a,b,c;
 a=-11;
b=10;
c = (a>b) ? a : b;
 cout << c;
 cin.get();
 return 0;
}
```

MOKYKLŲ TOBULINIMO CENTRAS

Papildomai

```
7==5 ? 4 : 3
7==5+2 ? 4 : 3
5>3 ? a : b
a>b ? a : b
```

MOKYKLŲ TOBULINIMO CENTRAS

IF-ELSE (keli veiksmai)

```
if (sąlyga)
        veiksmas(ai);
else
        veiksmas(ai);
```

Maksimumo paieška tarp dviejų skaičių

```
int main(void)
{
 int n,m;
cout << "Iveskite du skaicius" << endl;
cin >> n >> m;
  if (n>m)
 {
   cout “Didesnis skaicius:”;
   cout << n;
}
```

tęsinys

```
else
 {
   cout “Didesnis skaicius:”;
   cout << m;
 }
return 0;
 }
```

```
Iveskite du skaicius
5
8
Didesnis skaicius: 8
```

MOKYKLŲ TOBULINIMO CENTRAS

Persipynę IF-ELSE sakiniai

- Vienas IF operatorius leidžia daryti tik vieną sprendimą. Tačiau daugelyje atvejų programa turi padaryti eilę sprendimų. Tuo tikslu gali būti naudojami persipynę IF-ELSE operatoriai.

Maksimumo paieška tarp trijų skaičių

```
#include<iostream.h>
int main(void)
{
     int n,m,l;
      cout << "Iveskite du skaicius" << endl;
     cin >> n >> m >> l;
     if (n>m && n>l)
     {
              cout << n;
              }
              else if (m>n && m>l)
              {
                       cout << m;

                       }
                       else
                       {
                            cout << l;
                            }
                            cin.get();
                            cin.get();
}
```

((n > m) && (n >1))

MOKYKLŲ TOBULINIMO CENTRAS

Neigimo operatoriaus pavyzdys

- Neigimo operatoriaus taikymo pavyzdys:

```
if(( !jūs turite šunį)
cout<<"jūs privalote įsigyti šunį"<<endl;
```

MOKYKLŲ TOBULINIMO CENTRAS

Praktinis darbas

Parenkite programą, kuri ekrane parodytų

užrašą “Jūs atspėjote skaičių, jeigu įvestas skaičius sutapo su priskirtu programoje”,

užrašą “Skaičius mažesnis”, jei įvestas skaičius mažesnis, už priskirtą programoje,

užrašą “Skaičius didesnis”, jei įvestas skaičius didesnis

už priskirtą programoje.

MOKYKLŲ TOBULINIMO CENTRAS

Tema. **WHILE ciklas**

MOKYKLŲ TOBULINIMO CENTRAS

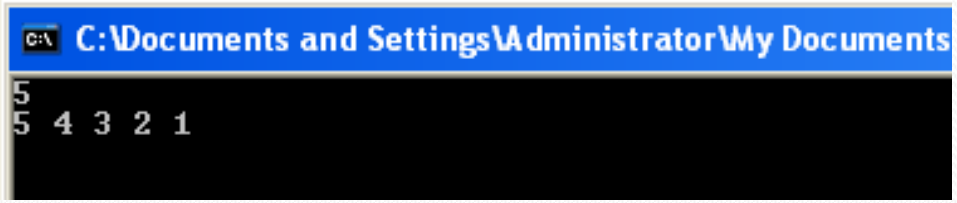
Ciklo struktūra

WHILE (sąlyga)

veiksmas (ai)

Pavyzdys1

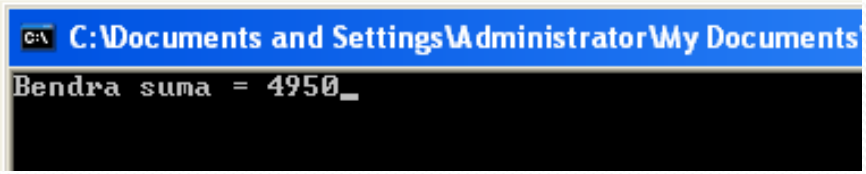
```
cin >> n;
while (n>0)
{
cout << n << "  ";
n--;
}
```



Pavyzdys2

Programa randa skaičių nuo 0 iki 100 sumą

```
int main(void)
{
 int suma =0;
 int skaicius =0;
 while (skaicius < 100)
{
   suma+=skaicius;
   skaicius++;
 }
cout << "Bendra suma = "  << suma ;
return 0;   }
```



MOKYKLŲ TOBULINIMO CENTRAS

Tema. **do-while ciklas**

MOKYKLŲ TOBULINIMO CENTRAS

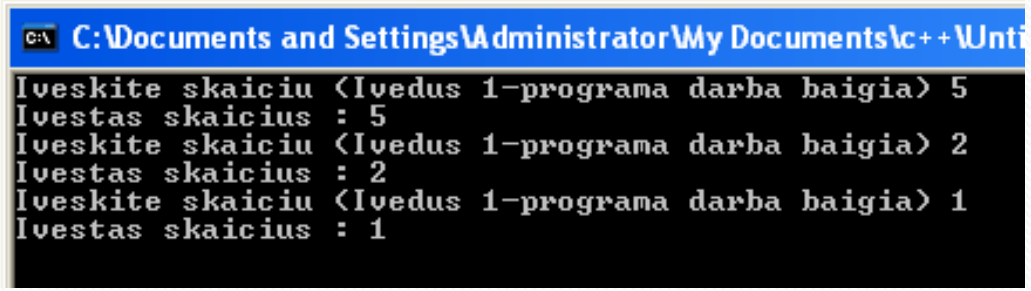
Ciklo struktūra

```
do
{
    veiksmai;
}
while (sąlygos tikrinimas);
```

Pavyzdys

Programa spausdina ekrane įvestą skaičių, kol neįvedamas skaičius 1.

```
int main ()
{
  unsigned long n;
   do {
    cout << "Iveskite skaiciu (Ivedus 1-programa darba
   baigia) ";
    cin >> n;
   cout << "Ivestas skaicius : " << n << "\n";
  }   while (n != 1);
return 0;
}
```

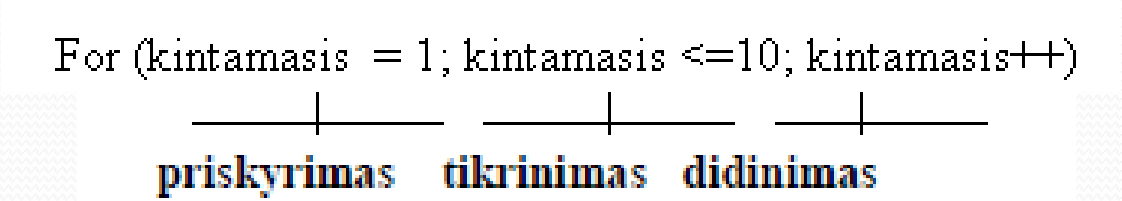


MOKYKLŲ TOBULINIMO CENTRAS

Tema. **FOR ciklas**

Ciklo struktūra

- FOR ciklas leidžia pakartoti vieną ar kelis veiksmus keletą kartų.
- Ciklo struktūra:

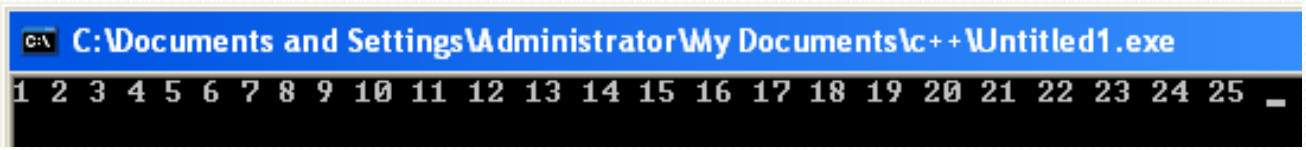


Šiuo atveju ciklas bus vykdomas 10 kartų.

Pavyzdys

Programa išveda į ekraną skaičius nuo 1 iki 25

```
int main(void)
{
int skaicius;
for (skaicius =1; skaicius <=25; skaicius ++)
cout << skaicius << " ";
return 0;
}
```



MOKYKLŲ TOBULINIMO CENTRAS

Kintamojo žingsnio keitimas

- Visi iki šiol nagrinėti *for* ciklai didindavo valdantyjį kintamąjį vienetu. Bet tai nėra vienintelis būdas. Programa išveda kas penktą skaičių:

Programa išveda kas penktą skaičių

```
int main(void)
{
int skaicius;
for (skaicius =1; skaicius <=25; skaicius +=5)
cout << skaicius << " ";
return 0;
}
```

Žingsnio mažinimas

- Naudojant ciklą for nebūtina skaitliuką tik didinti. Programa naudoja ciklą for skaičių nuo 26 iki 1 išvedimui mažėjančia tvarka.

Pavyzdys

```
int main(void)
{
int skaicius;
for (skaicius = 26; skaicius >=1; skaicius --)
cout << skaicius << " ";
return 0;
}
```

MOKYKLŲ TOBULINIMO CENTRAS

Pasipraktikuokite

- Parašykite programą, kuri rastų skaičių nuo n iki m sumą.
- Parašykite programą, kuri intervale nuo n iki m, rastų lyginių skaičių kiekį.

MOKYKLŲ TOBULINIMO CENTRAS

Veiksmas Break

- Norint nutraukti ciklą jo viduje galima naudoti *BREAK*.

Pavyzdys

```
int main (void)
{
 int n;
  for (n=1; n<10; n++)
{
 cout << n << "  ";
    if (n==5)
    {
      cout << "Ciklas nutraukiamas";
      break;
    } }
return 0;
}
```

Programa nutraukia darbą, kai kintamojo reikšmė tampa 5.

MOKYKLŲ TOBULINIMO CENTRAS

Tema. **SWITCH konstrukcija**

- Be sąlyginių sakinių, programavime taip pat labai dažnai reikalinga *switch* konstrukcija.

Struktūra

```
switch (kintamasis) {          //jei sveikasis skaičius
case 1:
{koks nors veiksmas}
break;
case 2:
{koks nors kitas veiksmas}
break;
....
default:
{numatytasis veiksmas}
```

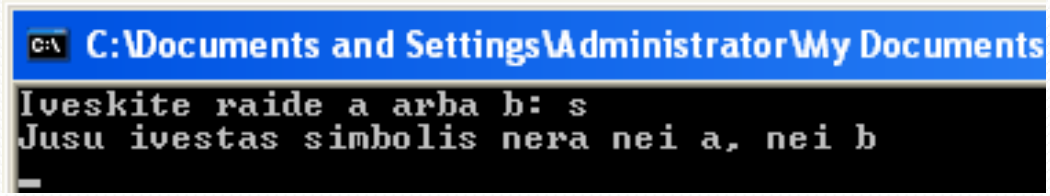
Pavyzdys1

Programa į ekraną atitinkamai išveda tekstą, priklausomai nuo įvesto skaičiaus reikšmės


```
cin >> n;
switch(n)
{
   case 5:
         cout << "Ivestas skaicius yra 5\n";
         break;
   case 6:
         cout << "Ivestas skaicius yra 6\n";
         break;
   default:
         cout << "Ivestas skaicius nera 5 arba 6\n";
         break;
}
```

Pavyzdys2

Programa, priklausomai nuo įvestos raidės,
į ekraną atitinkamai išveda tekstą



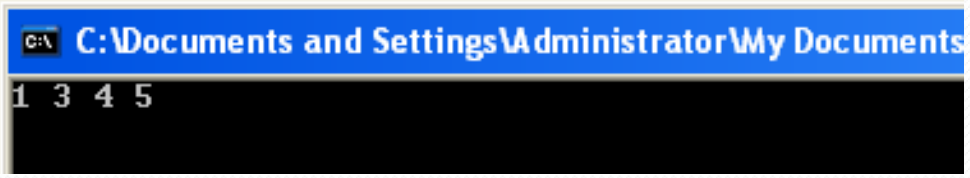

```
cin >> sim;
switch(sim)
{
  case 'a': case 'A':
      cout << "Ivedete raide a\n";
      break;
case 'b': case 'B':
    cout << "Ivedete raide a\n";
    break;
default:
    cout << "Jusu ivestas simbolis nera nei a, nei b\n";
    break;
   }
```

MOKYKLŲ TOBULINIMO CENTRAS

Tema. **CONTINUE konstrukcija**

Pavyzdys

Programa ignoruoja ciklo iteracija, kai n=2



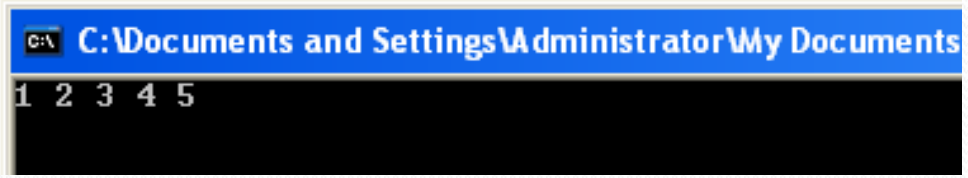

```
int main (void)
{
  for (int n=1; n<=5; n++) {
    if (n==2) continue;
    cout << n << ", ";
  }
return 0;
}
```

MOKYKLŲ TOBULINIMO CENTRAS

Tema. **GOTO konstrukcija**

Pavyzdys

Programa išveda skaičius nuo 1 iki 5, kol sąlyga teisinga




```
int main (void)
{  int i=1;
     loop:
     cout << i << " ";
     i++;
       if (i<=5) goto loop;
return 0;
}
```

MOKYKLŲ TOBULINIMO CENTRAS

II dalis

MOKYKLŲ TOBULINIMO CENTRAS

Tema. **Masyvai**

Teoriniai pastebėjimai

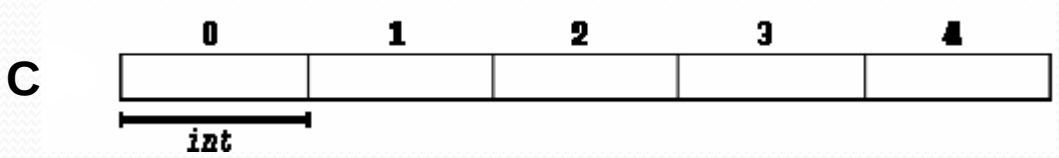
- Masyvas tai grupė kintamųjų, turinčių tą patį tipą. Norėdami kreiptis į kokį nors masyvo elementą, privaloma nurodyti masyvo vardą ir masyvo elemento numerį arba kitaip vadinamą indeksą.

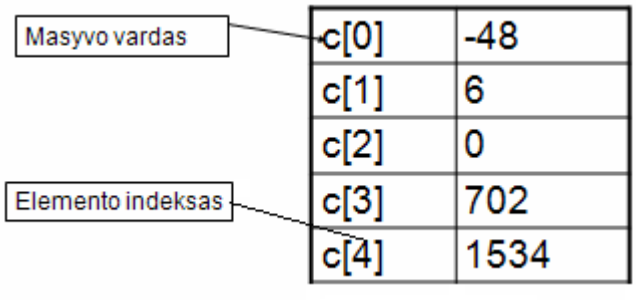
tęsinys

- Numeracija prasideda nuo 0, t.y. Pirmojo masyvo elemento numeris (indeksas) yra 0, antrojo 1, ir t.t. N-tojo masyvo elemento numeris yra N-1. Masyvo elementų indeksas nurodomas laužtiniuose skliaustuose [].

Masyvo iliustracija

5 elementų masyvas ***c***:





Vienmačio masyvo aprašymas

```
int mas[10];
```

Vienmačio masyvo užpildymas

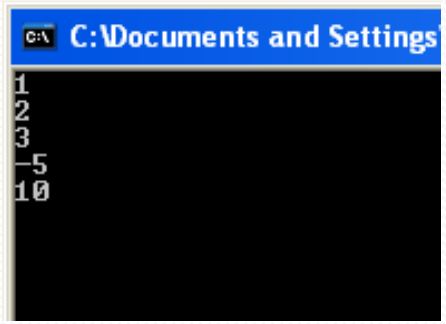
1. Visi elementai užpildomi 0:

int mas[10] = {0};

2.Elementai užpildomi reikšmėmis:

int masyvas[5] = {1,2,3,-5,10};

int masyvas[] = {1,2,3,-5,10};



3. Blogai. Reikšmių 6, o masyvo dydis tik 5.

int blogai[5] = {7,-1,3,-3,0,8};

MOKYKLŲ TOBULINIMO CENTRAS

Reikšmės priskyrimas elementui

```
mas[2] = 15;
```

Masyvo elemento turinio priskyrimas kintamąjam

```
a = mas[1];
```

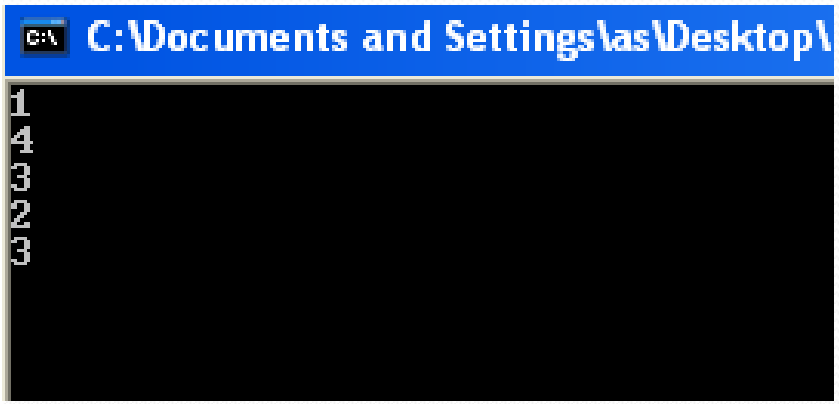
Pavyzdys

```
int mas [] = {1, 2, 3, 4, 5};
int main (void)
{
 int i, suma=0;
  for ( i=0 ; i<5 ; i++ )
  {
    suma += mas[i];
  }
  cout << suma;
return 0;
}
```

Programa randa masyvo elementų reikšmių sumą

Pavyzdys2 – Naudojant konstantą

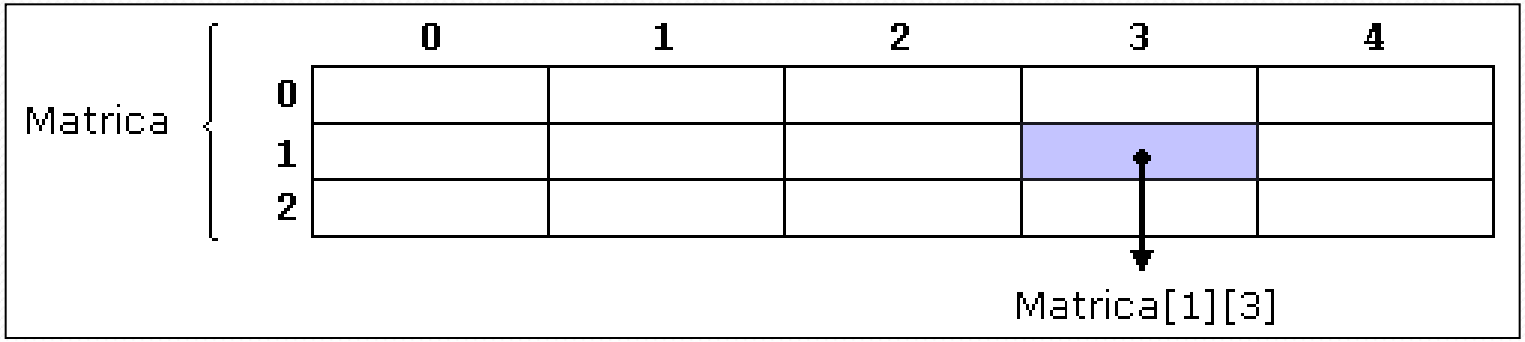
```
#include<iostream>
# define SIZE 5

using namespace std;

int main(void)
{
     int i;
     int mas[SIZE]={1,4,3,2,3};

     for (i=0; i<SIZE; i++)
     {
          cout << mas[i] << "\n";
          }
          cin.get();
          cin.get();
          return 0;
          }
```



MOKYKLŲ TOBULINIMO CENTRAS

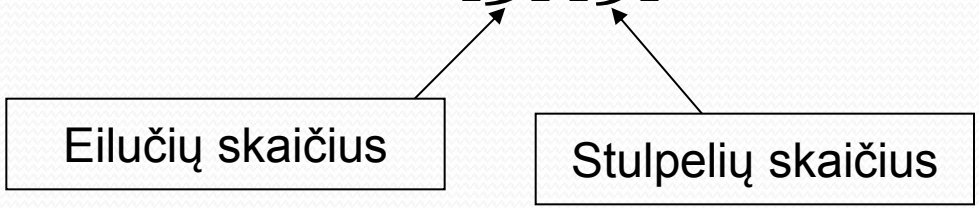
Dvimatis masyvas

Iliustracija



Aprašymas

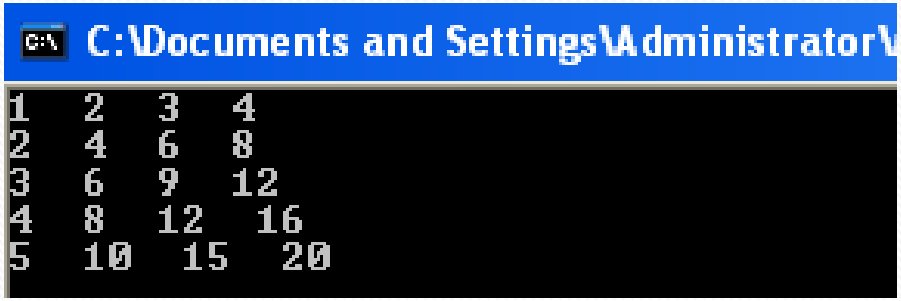
```
int Matrica [3][5];
```



Pavyzdys

Programa užpildo matricą reikšmėmis pagal pateiktą formulę: (i+1)*(j+1);

```
int Matrica [5][4];
int main (void)
{
int i,j;
for (i=0;i<5;i++)
    for (j=0;j<4;j++)
   {
      Matrica[i][j]=(i+1)*(j+1);     // užpildymas
    }
```



tęsinys – išvedimas į ekraną

```
for (i=0;i<5;i++)
 {
    for (j=0;j<4;j++)
    {
     cout << Matrica[i][j] << "  ";
    }
    cout << '\n';
}
return 0;
}
```

MOKYKLŲ TOBULINIMO CENTRAS

Tema. **Simbolių eilutės**

Aprašymas

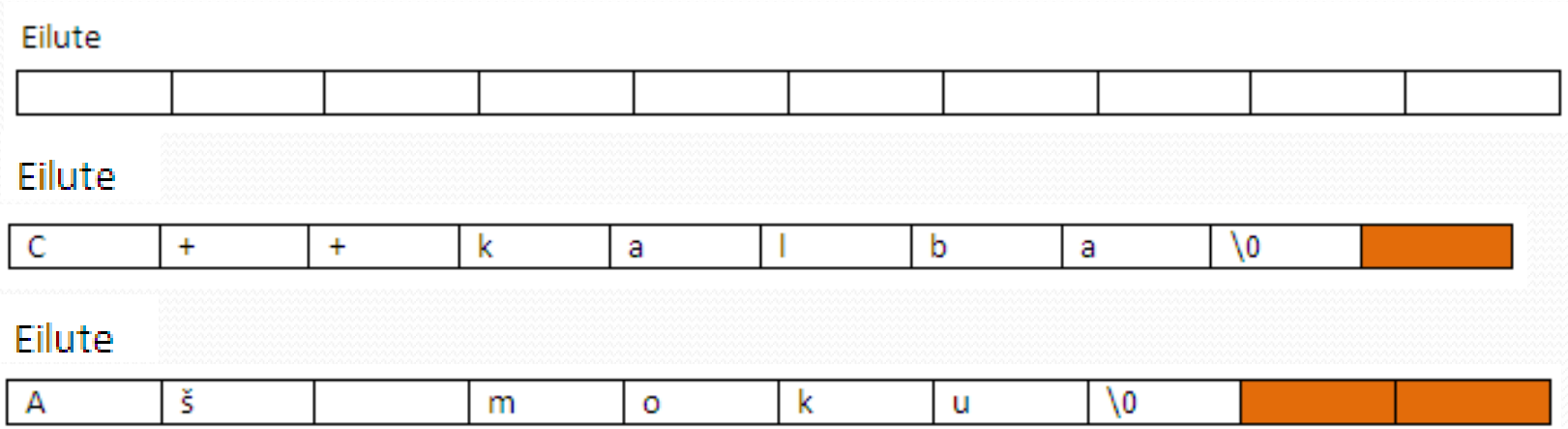
char Eilute [10];

Turime masyvą, kuris saugoja 10 elementų, kurių tipas char.

Užpildymas

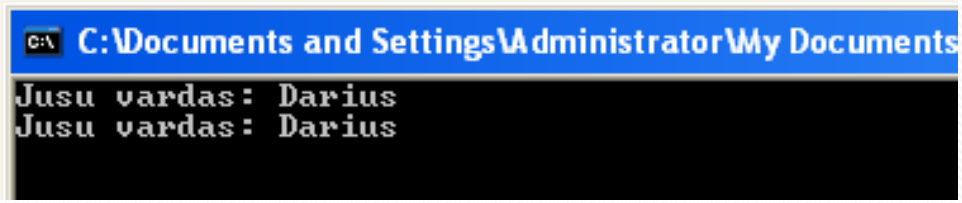
char Eilute[] = { ‘C', ‘+', ‘+', ‘k', ‘a', ‘l', ‘b', ‘a', '\o' };

char Eilute[] = “Aš moku";



Pavyzdys

Programa parodo Jūsų įvestą vardą




```
int main (void)
{
   char tekstas[] = "Jusu vardas: ";
   char vardas [30];
   cout << tekstas;
   cin >> vardas;
   cout << tekstas << vardas;
return 0;
}
```

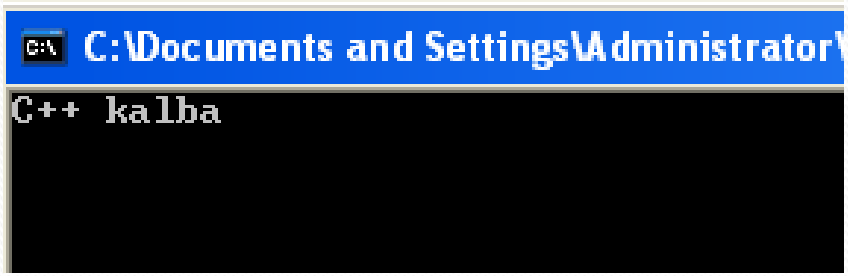
MOKYKLŲ TOBULINIMO CENTRAS

Konvertavimas

- Simbolių eilutė, saugoma char masyve, lengvai konvertuojama į eilutę (string).

Pavyzdys

```
int main (void)
{
 string eilute;
char  tekstas[]=“C++ kalba";
eilute= tekstas;
cout <<eilute;
return 0;
}
```



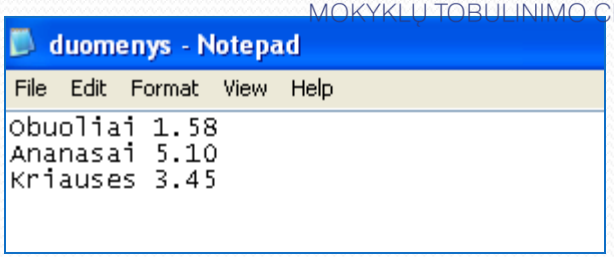
MOKYKLŲ TOBULINIMO CENTRAS

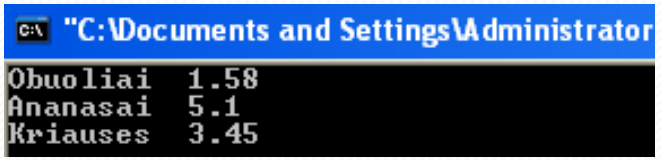
Tema. **Darbas su failais.**

- Biblioteka <fstream>

Pavyzdys

```
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
ifstream in ("duomenys.dat");
```





tęsinys

```
char daiktas[20];
float kaina;


if (!in)
    {
    cout << "Neimanoma atidaryti failo
   duomenys.dat\n";
    return 1;
    }
```

tęsinys

```
in >> daiktas >> kaina;
cout << daiktas << "  "  << kaina << "\n";

in >> daiktas >> kaina;
cout << daiktas << "  "  << kaina << "\n";

in >> daiktas >> kaina;
cout << daiktas << "  "  << kaina << "\n";
```

tęsinys

```
    in.close();
    return 0;
}
```

Informacijos išvedimas į failą

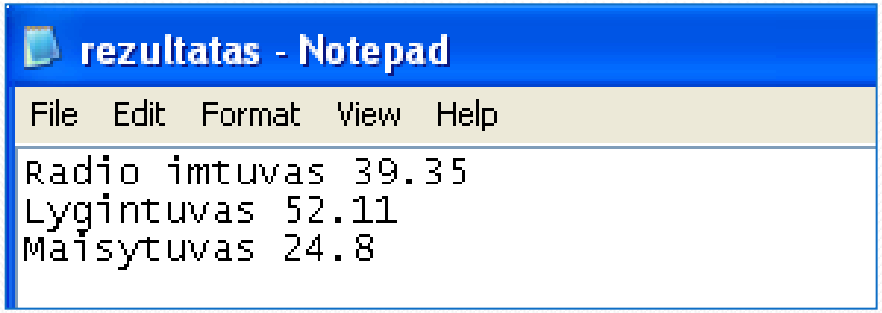
```
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
    ofstream out ("rezultatas.dat");
```

tęsinys

```
out << "Radio imtuvas" << 39.35 << endl;
out << "Lygintuvas" << 52.11 << endl;
out << "Maisytuvas" << 24.80 << endl;
```

tęsinys

```
    out.close();
    return 0;
}
```



MOKYKLŲ TOBULINIMO CENTRAS

Tema. Funkcijos

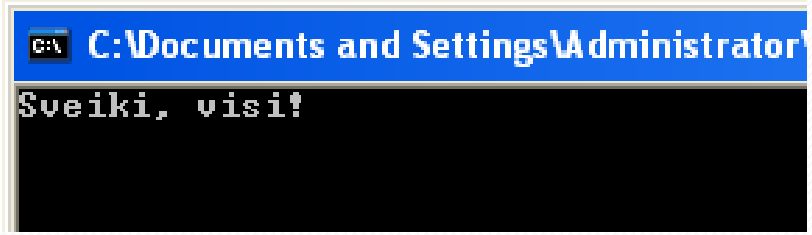
MOKYKLŲ TOBULINIMO CENTRAS

Funkcijos struktūra

```
Tipas Vardas (parametrai)
{
veiksmai
}
```

Funkcija be tipo (void)

```
void pasisveikinimas ()
{
 cout << "Sveiki, visi!";
}
int main (void)
{   pasisveikinimas();
return 0;  }
```

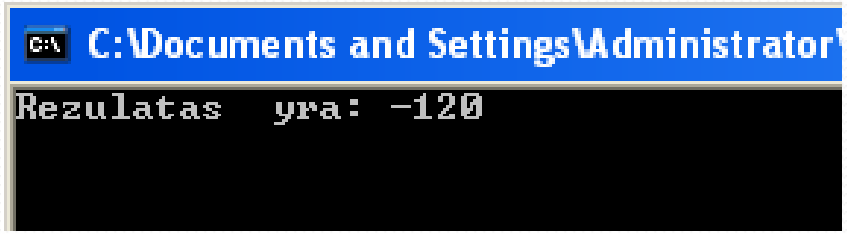


Funkcija grąžina vieną reikšmę

```
int sandauga (int pirmas, int antras)
{
  int rezultatas;
  rezultatas = pirmas*antras;
  return (rezultatas);
}
int main (void)
{
```

tęsinys

```
int rez;
    rez = sandauga (15,-8);
    cout << "Rezulatas  yra: " << rez;
return 0;
}
```

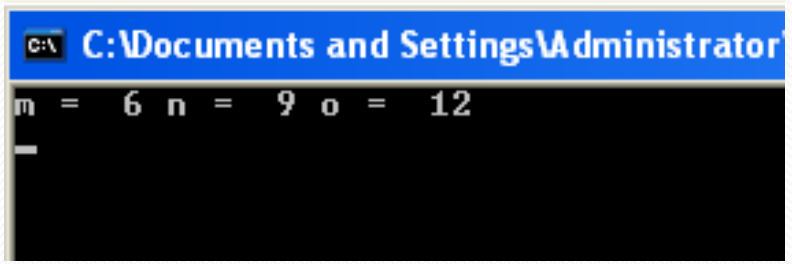


Funkcija grąžina kelias reikšmes

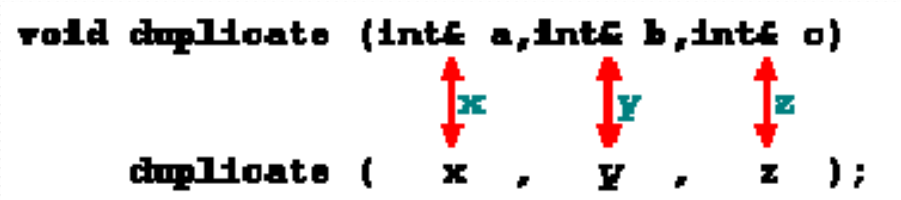
```
void trigubas (int& m, int& n, int& o)
{
  m = m*3;
  n = n*3;
  o = o*3;
}
```

tęsinys

```
int main (void)
{
   int m=2, n=3, o=4;
   trigubas(m,n,o);
   cout << "m =  " << m << " n =  " << n << "  " o
   =  " << o << endl;
return 0;
}
```



Kaip tai vyksta?

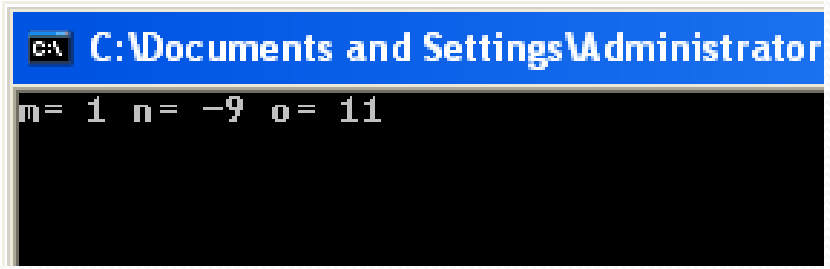


Kelių reikšmių grąžinimas, naudojant pradinę

```
void zingsnis (int m, int& n, int& o)
{
  n = m-10;
  o = m+10;
}
```

tęsinys

```
int main (void)
{
   int m=1, n, o;
   zingsnis(m,n,o);
   cout << "m= " << m << " n= " << n <<  " o= "
   << o << endl;
return 0;
}
```

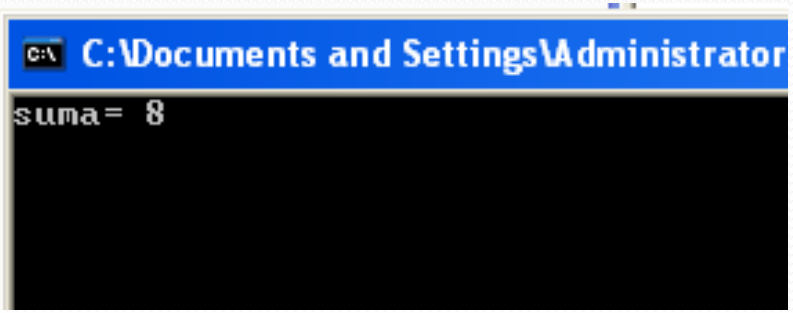


Pradinės reikšmės funkcijos argumentų sąraše

```
int suma (int i, int j=5)
{
  int rez;
  rez= i+j;
  return rez;
}
```

tęsinys

```
int main (void)
{
   cout << "suma= " << suma(3) << endl;
return 0;
}
```

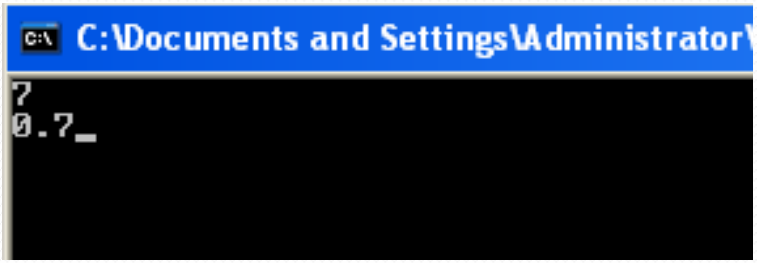


Persidengiančios funkcijos

```
#include <iostream>
using namespace std;
int  veiksmai (int i, int j)
{
    return (i+j);
}
double  veiksmai (double i, double j)
{
    return (i-j);
}
```

tęsinys

```
int main (void)
{
    int a=3, b=4;
    double m=1.45, n=0.75;
    cout << veiksmai (a,b) << '\n';
    cout << veiksmai (m,n);
return 0;
}
```



MOKYKLŲ TOBULINIMO CENTRAS

Rekursija

```
long faktorialas (long sk)
{
   if (sk>1)
    return (sk * faktorialas (sk-1));
    else
      return (1);
}
```

tęsinys

```
int main (void)
{
    long skaicius;
    cout << "Prasome ivesti skaiciu: ";
    cin >> skaicius;
    cout << "Skaiciaus " << skaicius << " faktorialas
   yra lygus: " << faktorialas(skaicius);
return 0;
}
```

MOKYKLŲ TOBULINIMO CENTRAS

Funkcijų prototipų aprašymas

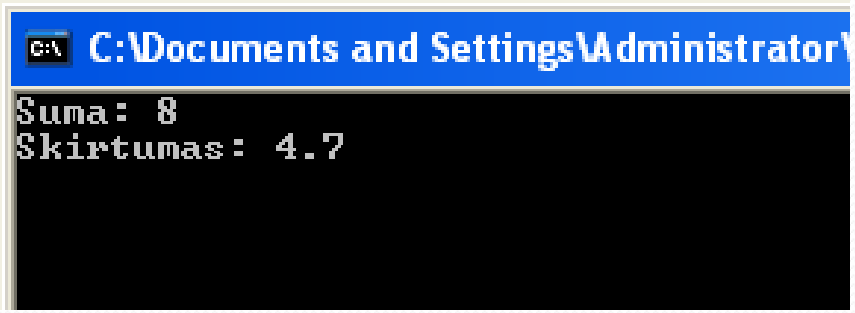
```
#include <iostream>
using namespace std;
int sudetis(int a, int b);
float atimtis(float c, float d);
```

tęsinys

```
int main (void)
{
  int suma = sudetis(5,3);
  float skirtumas = atimtis(9.5,4.8);
  cout << "Suma: "<< suma << endl;
  cout << "Skirtumas: "<< skirtumas << endl;
return 0;
}
```

tęsinys

```
int sudetis(int a, int b)
{
    return a+b;
 }
float atimtis(float c, float d)
{
    return c-d;
 }
```



MOKYKLŲ TOBULINIMO CENTRAS

Eilutės funkcijos

Funkcija	Trumpas aprašymas
`strcpy`	Kopijuoja eilute2 į eilutę1
`strcat`	Prie eilutės1 pabaigos prijungia eilutę2
`strcmp`	Lygina eilutę1 su eilute2, skiria didžiasiais ir mažasiais raides
`stricmp`	Lygina eilutę1 su eilute2, neskiria didžiųjų ir mažųjų raidžių
`strdup`	Sukuria eilutės kopiją
`strlen`	Išveda eilutės simbolių skaičių
`strlwr`	Visi eilutės simboliai pakeičiami mažosiomis raidėmis
`strupr`	Visi eilutės simboliai pakeičiami didžiosiomis raidėmis
`strncat`	Eilutės2 nurodytas simbolių skaičius prijungiamas prie eilutės1 galo
`strncpy`	Kopijuoja nurodytą eilutės2 simbolių skaičių į eilutę1
`strrev`	Eilutės reversas
`isalnum(s)`	Tiesa, jei s yra skaičius arba raidė
`isalpha(s)`	Tiesa, jei s yra raidė
`islower(s)`	Tiesa, jei s yra mažoji raidė
`isupper(s)`	Tiesa, jei s yra didžioji raidė

Iliustracija

```
   char str[20]="Reiksme";
   char str[20];

   tipas funkcija (eilutė1, eilutė2);
   char strcpy (char str1, char str2);
```

Pvz.1:

```
#include <iostream.h>
#include <string.h>
void main()
{
char strs1[20]="mm";
char strs2[20]="pp";
strcpy (strs1, "Labas ");
strcpy (strs2, "rytas");
strcat(strs1, strs2);
cout << strs1<<"\n";
}
```

MOKYKLŲ TOBULINIMO CENTRAS

Tema. Struktūros

MOKYKLŲ TOBULINIMO CENTRAS

Teoriniai pastebėjimai

- Visi kintamieji, iki šiol kuriais naudojotės, priklausė baziniams C/C++ duomenų tipams:
  - Sveiko tipo (int) kintamieji ir konstantos;
  - Realaus tipo (float) kintamieji ir konstantos;
  - Dvigubo tikslumo (double) kintamieji ir konstantos;
  - Simbolinio tipo (char) kintamieji ir konstantos;

Šie duomenys galėjo sudaryti masyvus, tačiau masyvo elementais gali būti *tik to paties bazinio tipo duomenys*

- **Struktūra** – tai vienodo arba skirtingo tipo kintamųjų rinkinys.

MOKYKLŲ TOBULINIMO CENTRAS

Struktūros mūsų gyvenime

- Sąrašai
  - Įmonės darbuotojai
    - Vardas, pavardė, asmens kodas, adresas, gimimo metai, išsilavinimas, paso Nr.
  - Telefono numeriai
    - Vardas, pavardė, adresas, tel. nr.

Struktūrų sintaksė

```
struct  struktūros_vardas
{
   lauko_tipas1   lauko_vardas1;
   lauko_tipas2   lauko_vardas2;
   (.....)
lauko_vardasn;
};
```

```
struct preke
{
int svoris;
int kiekis;
float kaina;
};
```

Struktūrą galima naudoti tik paskelbus

Struktūros objekto paskelbimas

```
struct  struktūros_vardas
{
   lauko_tipas1    lauko_vardas1;
   lauko_tipas2   lauko_vardas2;
   (.....)
    lauko_tipasn  lauko_vardasn;
} objekto(-ų)_vardas(-ai);
```

arba

```
struct struktūros_vardas
{
   lauko_tipas1    lauko_vardas1;
   lauko_tipas2    lauko_vardas2;
   lauko_tipas3    lauko_vardas3;
   (....)
   } ;
struktūros_vardas    objekto(-ų)_vardas(-ai);
```

MOKYKLŲ TOBULINIMO CENTRAS

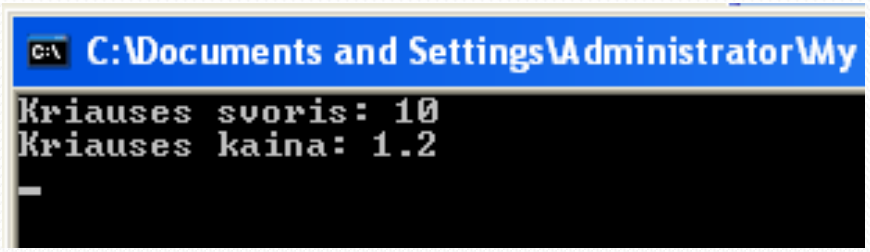
Pavyzdys

```
struct   preke {
      int svoris;
      float kaina;
};
preke kriause;
preke ananasas, kivi;
```

tęsinys

```
int main(void)
{
    kriause.svoris=10;
    kriause.kaina=1.2;
    cout << "Kriauses svoris: " << kriause.svoris <<
   endl;
    cout << "Kriauses kaina: " << kriause.kaina <<
   endl;
return 0;
 }
```

Kreipiantis į struktūros elementus, vartojami sudėtiniai vardai



MOKYKLŲ TOBULINIMO CENTRAS

Struktūros – funkcijų argumentai

- Struktūros laikomos vartotojo sukurtu duomenų tipu, todėl kaip ir bet kuris bazinis duomenų tipas taip ir struktūros gali būti perduotos per funkcijos argumentų sąrašą. Perdavimas atliekamas kopijuojant reikšmes.

Iliustracija

```
struct zmogus
 {
    char lytis[10];
    float ugis;
};
```

Funkcijos prototipas

```
void  Funkcija(struct zmogus);
```

Pavyzdys

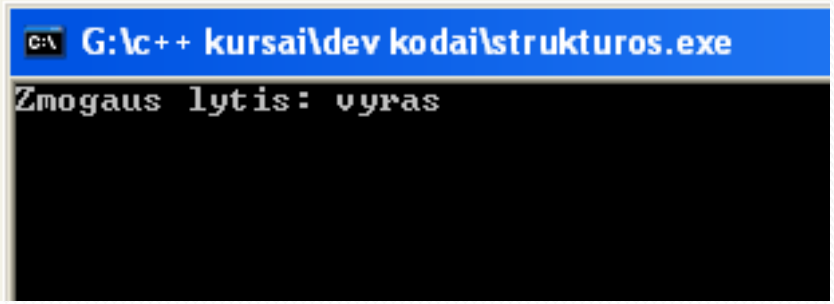
```
struct zmogus
 {
     char lytis[10];
     float ugis;
 }   baltaodis, juodaodis;

void  funkcija(struct zmogus);  // prototipas
```

tęsinys

```
int main()
{
   strcpy(baltaodis.lytis,"vyras");
   strcpy(juodaodis.lytis,"moteris");
   baltaodis.ugis=81.5;
   juodaodis.ugis=54.8;
   cout << "Zmogaus lytis: " ;
   funkcija(baltaodis);
 return 0;
  }
```

tęsinys

```
void  funkcija(zmogus tipas)
{
    cout << tipas.lytis;
 }
```



MOKYKLŲ TOBULINIMO CENTRAS

Teoriniai pastebėjimai

- Funkcija taip pat gali grąžinti struktūros tipo objektą. Tuo atveju struktūros duomenys perduodami išviečiančiai funkcijai. Pavyzdžiui funkcijos prototipas:
- struct zmogus funkcija(int, float);

Pavyzdys

```
#include <cmath>


struct duomenys
{
   double kampas;
};
```

tęsinys

```
struct rezultatas
{
  double sinusas;
  double kosinusas;
  double tangentas;
};

rezultatas  skaiciuoti(struct  duomenys  manostruktura);
```

tęsinys

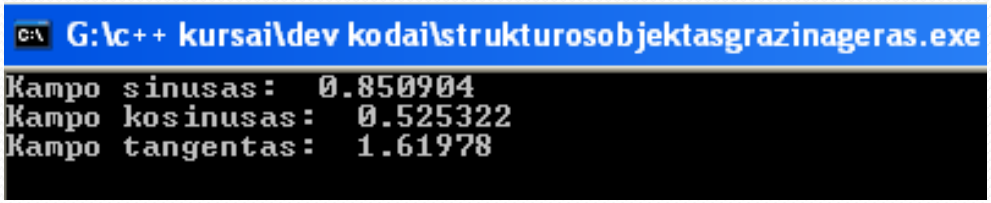
```
int main ()
{
    duomenys  ivedimas;
    rezultatas  isvedimas;

    ivedimas.kampas = 45;
    isvedimas = skaiciuoti(ivedimas);
```

tęsinys

```
 cout << "Kampo sinusas:  " << isvedimas.sinusas << "\n";
 cout << "Kampo kosinusas:  " << isvedimas.kosinusas << "\n";
 cout << "Kampo tangentas:  " << isvedimas.tangentas << "\n";
return 0;
}
```

tęsinys

```
rezultatas   skaiciuoti(struct duomenys manostruktura)
{
   rezultatas atsakymas;
   atsakymas.sinusas = sin(manostruktura.kampas);
   atsakymas.kosinusas = cos(manostruktura.kampas);
   atsakymas.tangentas = tan(manostruktura.kampas);

   return atsakymas;
};
```



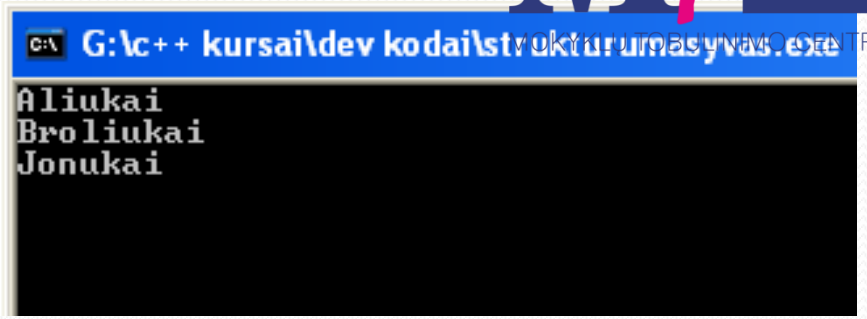
Struktūrų masyvas

Dažniausiai naudojamos ne pavienės struktūros, o jų masyvai, kurie sudaro duomenų bazių pagrindą.
Struktūrų masyvai apibrėžiami analogiškai, kaip ir įprastiniai bazinių duomenų tipų masyvai.

Pavyzdys

```
struct dvd
{    char pavadinimas[15];
     float kaina;
     int kiekis;
};
dvd diskas [3];
```

tęsinys




```
int main ()
{
    strcpy(diskas[0].pavadinimas, "Aliukai");
    strcpy(diskas[1].pavadinimas, "Broliukai");
    strcpy(diskas[2].pavadinimas, "Jonukai");
      for (int i = 0; i < 3; i++)
          cout << diskas[i].pavadinimas << endl;
            return 0;
}
```

MOKYKLŲ TOBULINIMO CENTRAS

Struktūra struktūroje

```
struct mokinys
 {
    char pavarde[40];
    char vardas[40];
```

tęsinys

```
struct gimimodiena
{
unsigned int metai;
unsigned int menuo;
unsigned int diena;
}  data;
};
```

```
mokinys mok[2];  // 2 mokiniu irasu masyvas
```

tęsinys

```
int main ()
{
for (int i=1;i<=2; i++)
{
   printf ("Iveskite pavarde \n");

   scanf("%s",  mok[i].pavarde);
   printf( "Iveskite varda \n");
   scanf("%s",  mok[i].vardas);
```

tęsinys

```
 printf("Iveskite gimimo metus\n");
scanf("%d",  &mok[i].data.metai);
printf( "Iveskite menesi  \n");
scanf("%d",  &mok[i].data.menuo);
printf( "Iveskite diena  \n");
scanf("%d", & mok[i].data.diena);
}
```

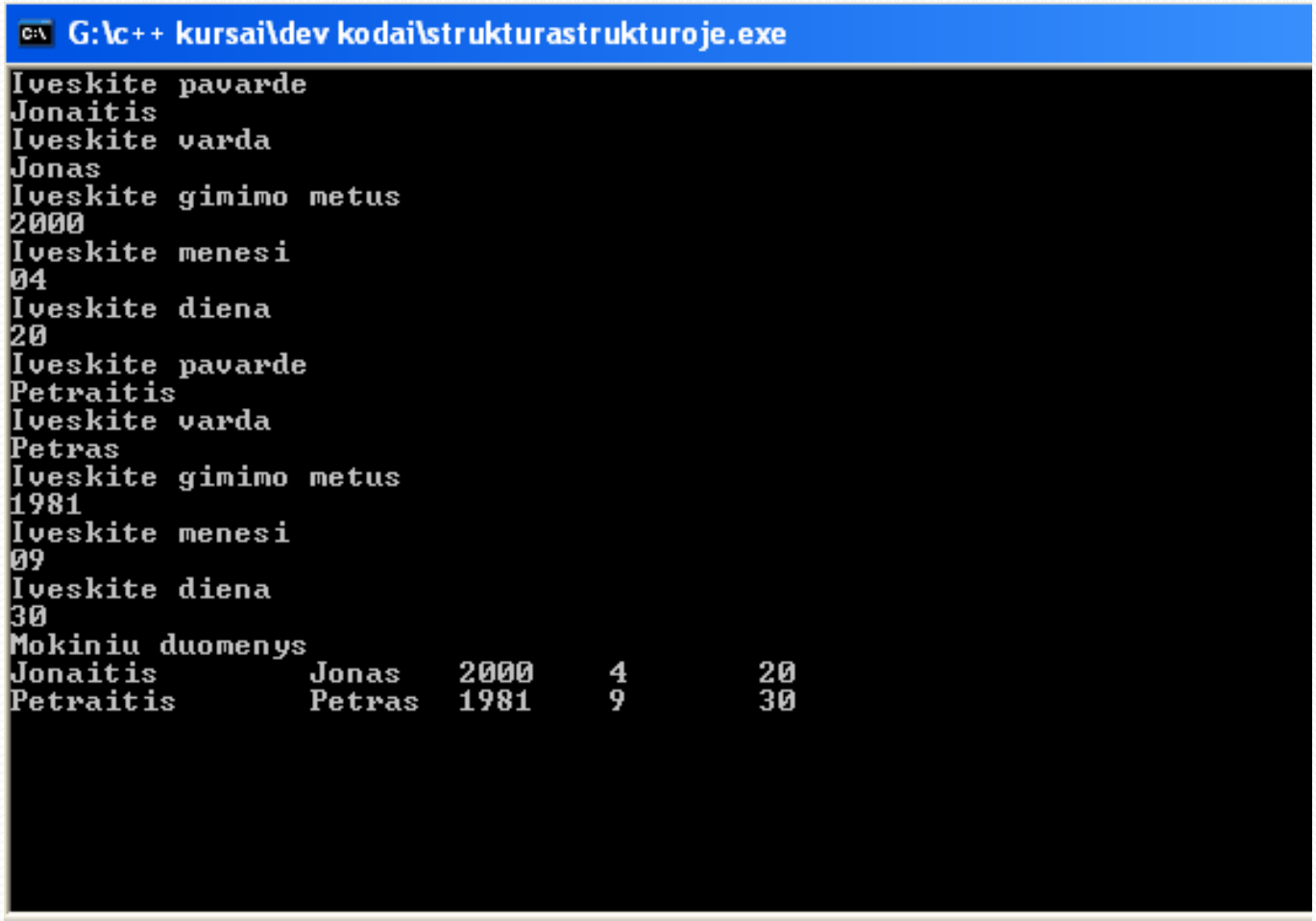
tęsinys

```
printf( "Mokiniu duomenys  \n");
   for (int i=1;i<=2; i++)
{
    printf("%s",  mok[i].pavarde);
    printf("\t");
    printf("%s",  mok[i].vardas);
    printf("\t");
   printf( "%d",  mok[i].data.metai);
   printf("\t");
```

tęsinys

```
    printf( "%d",  mok[i].data.menuo);
    printf("\t");
    printf( "%d",  mok[i].data.diena);
          printf("\n");
}
return 0;
}
```

Rezultatas ekrane



MOKYKLŲ TOBULINIMO CENTRAS

Tema. Klasės

MOKYKLŲ TOBULINIMO CENTRAS

Struktūrinio programavimo problema

- Didelės programos tampa labai sudėtingos t.y. egzistuoja tūkstanciai kintamųjų ir funkcijų vardų ir sunku juos kontroliuoti.
- Duomenys dažniausiai globalūs ir gali būti pasiekiami visų funkcijų.
- Duomenys nesurišti su funkcijomis blogai atvaizduoja realų pasaulį.

MOKYKLŲ TOBULINIMO CENTRAS

Realaus pasaulio modeliavimas

- Realiame pasaulyje egzistuoja objektai (žmonės, automobiliai...).
- Objektai turi atributus-požymius (automobilis: spalvą, galią, durų skaičių...)
- Objektai turi elgseną, t.y. jie atlieka tam tikrą veiksmą, priklausomai nuo situacijos

  (automobilis sustoja, paspaudus stabdžius....).

MOKYKLŲ TOBULINIMO CENTRAS

OOP pagrindinė idėja

- Duomenys ir funkcijos, kurios operuoja tais duomenimis apjungti į vieną vienetą, vadinamą **objektu**.
- Objekto funkcijos, kurių dažniausia būna ne viena, vadinamos **metodais**.
- Objekto duomenys dažniausiai pasiekiami (nuskaitomi, modifikuojami ir t.t.) tik per metodus. Tai reiškia, kad duomenys yra paslėpti nuo atsitiktinio modifikavimo. Kitaip dar sakoma, kad duomenys yra **inkapsuliuoti** (***encapsulated***).

Taigi

- C++ programa paprastai sudaro tam tikras skaičius objektų, kurie komunikuoja tarpusavyje iškviesdami vienas kito metodus.

MOKYKLŲ TOBULINIMO CENTRAS

Objektų pavyzdžiai

- **Žmogiški objektai**

  Darbuotojai, Studentai, Pirkėjai, Pardavėjai.

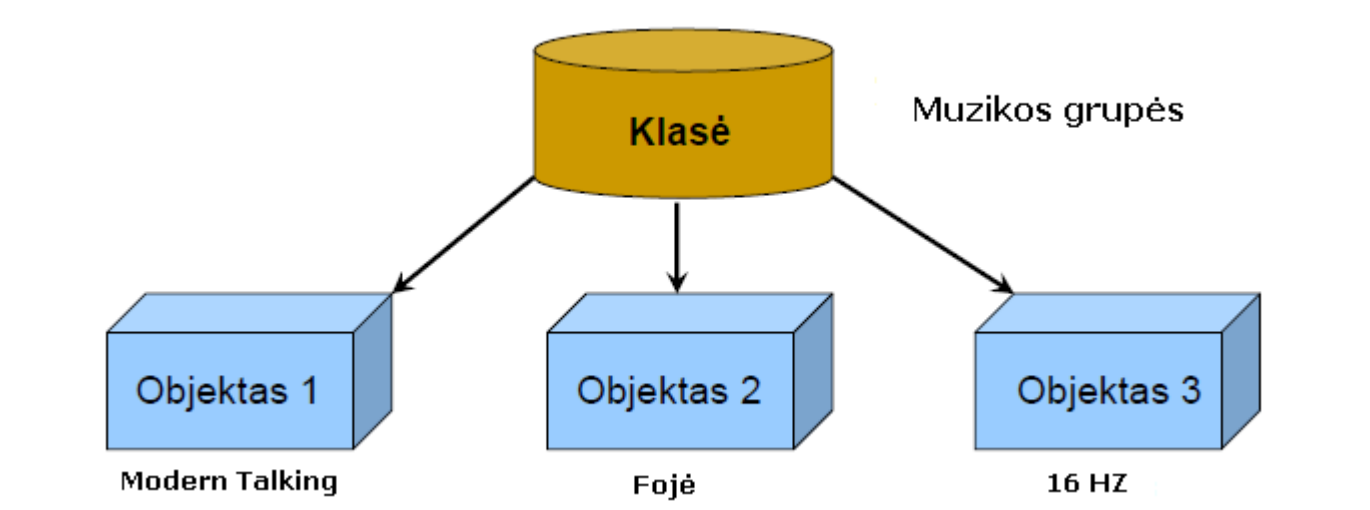
- **Kompiuteriniai žaidimai**

  Automobiliai lenktynėse, Kariai ir priešai kovos lauke.

- **Kompiuterio aplinka**
- Langai, Meniu, Mygtukai, Grafiniai objektai.

Klasės

- **Klasė – tai šablonas, pagal kurį kuriamas objektas. Ji apibrėžia kokie** duomenys ir kokios funkcijos bus tos klasės objekte.
- Klasė nekuria objekto, o tik jį aprašo.



Klasės apibrėžimas

- C++ kalboje struktūra, jungianti savyje kintamuosius, skirtus duomenims saugoti, ir funkcijas, kurios naudoja tik tuos kintamuosius, vadinama **klase**.

Klasės kintamieji vadinami **duomenimis**, o funkcijos – **metodais**.

MOKYKLŲ TOBULINIMO CENTRAS

Klasės aprašo struktūra

```
class   Klasės_vardas
{
 duomenų elementai;
      public: metodai;
}
Objektų_ sarašas;
```

Pavyzdys

```
class skaiciai
{
private: int pradiniai_duomenys;

public: void gauti(int skaicius)
                { pradiniai_duomenys = skaicius; }
 void rodyti()
 {
 cout<<"Ivesti duomenys = “ <<pradiniai_duomenys<<endl;
 }
 };
```

MOKYKLŲ TOBULINIMO CENTRAS

Klasės elementų požymiai

- Klasės elementai (duomenys ir metodai) gali turėti požymius. Požymis klasėje galioja tol, kol bus sutiktas kito požymio užrašas.
- Jeigu požymio užrašo nėra, tuomet pagal nutylėjimą bus priimtas private visiems elementams iki pirmojo sutikto požymio užrašo, jeigu jis iš viso bus.

MOKYKLŲ TOBULINIMO CENTRAS

C++ klasės elementų požymiai

- **private (lokalusis).** Duomenys ir metodai prieinami tik klasės metodams.
- **public (globalusis).** Klasės elementai prieinami tiek klasės metodams tiek ir išorinms funkcijoms.
- **protected (apsaugotasis).** Klasės elementai prieinami klasėje, kuri paveldi duotąja klasę. Paveldėtoje klasėje jie galioja *private* teisėmis.

MOKYKLŲ TOBULINIMO CENTRAS

Metodų aprašymas

- Metodai klasėje gali būti pilnai aprašyti.
- Tokius metodų aprašus tikslinga turėti, jeigu jų tekstas yra trumpas.
- Dažniausiai metodų aprašai iškeliami už klasės ribų. Tuomet klasėje rašomas tik metodo prototipas (metodas deklaruojamas).

MOKYKLŲ TOBULINIMO CENTRAS

Metodo aprašo klasės išorėje sintaksė

[Reikšmės tipas]

[Klasės Vardas] :: [Metodo vardas]
(Parametrų sarašas)

{

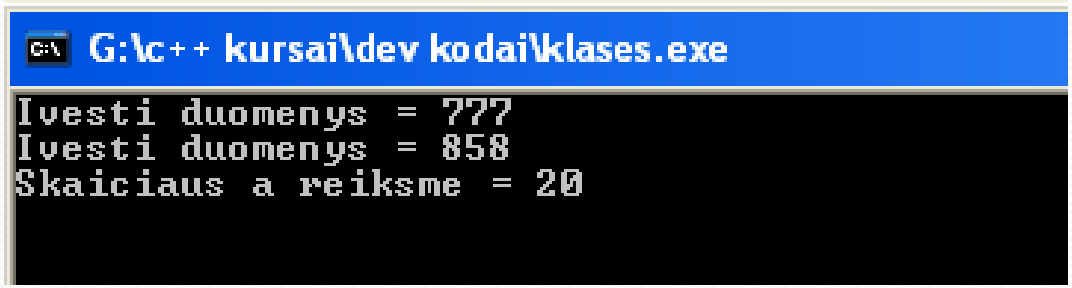
Programos tekstas

}

Išorinio metodo pavyzdys

```
class skaiciai
{
private: int pradiniai_duomenys,  a;
.......
int skaiciai::metodas_rodyti()
  {    a=20;
       cout<< "Skaiciaus a reiksme = " << a<<endl;
      return a;
}    };
```

```
int main(void)
{
.....
pirmas.skaiciai::metodas_rodyti();
.....
}
```



MOKYKLŲ TOBULINIMO CENTRAS

Klasės savybės

- Paprastai klasės duomenys būna **private**, o funkcijos **public.**
- Klasė formuoja savo vardų erdvę, todėl norint iš vienos klasės kreiptis į kitos klasės kintamąjį reikia naudoti tokią sintaksę:

klasės_vardas::kintamojo_ar_funkcijos vardas

MOKYKLŲ TOBULINIMO CENTRAS

Pavyzdys

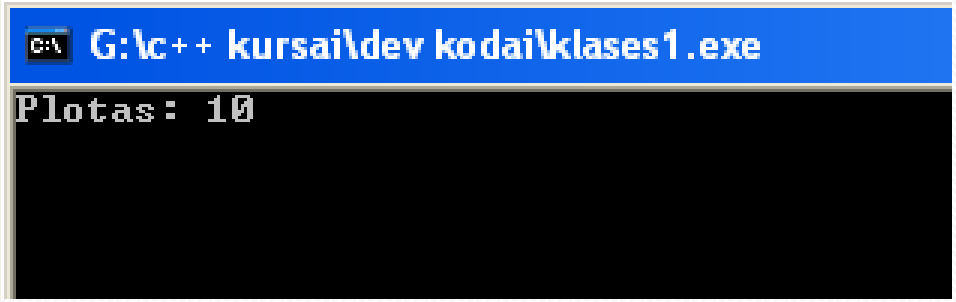
```
#include <iostream>
using namespace std;
class Staciakampis {
        int i, j;
public: void ivedimas (int,int);
int plotas ()
{
  return (i*j);
}
};
```

tęsinys

```
void Staciakampis::ivedimas (int sk1, int sk2)
{
    i = sk1;
    j = sk2;
}
```

tęsinys

```
int main (void)
{
   Staciakampis rect;
      rect.ivedimas (2,5);
      cout << "Plotas: ";
      cout<<rect.plotas()<<endl;
      return 0;
}
```



MOKYKLŲ TOBULINIMO CENTRAS

Praktinės užduotys

1. Apskaičiuokite a^2+4b, kur $b=(a-1)/2$;
2. Panaudokite sąlygos operatorių if, kad spausdintų tik lyginius skaičius.
3. Parašyti programą, kuri išvestų skaičiaus n daugybos lentelę.
4. Parašykite programą, kuri skaičiuotų nurodyto skaičiaus faktorialą.
5. Parašykite programą, kuri intervale nuo n iki m rastų nelyginius skaičius, o nuo m iki n lyginius skaičius.

tęsinys

6. Parenkite programą, kuri leistų vartotojui užpildyti masyvą iš 5 elementų. Užpildyto masyvo elementus išrūšiuokite didėjančia tvarka ir išveskite į ekraną.
7. Parenkite programą, kuri sudėtų dviejų masyvų elementus (masyvai sudaryti iš 10 elementų ir sudėtis turi būti: pirmo masyvo pradžia sudedama su antrojo masyvo pabaiga) ir rezultatą surašytų į trečią masyvą. Į ekraną turi būti išvesti pradiniai masyvai bei jų sumos masyvas.

tęsinys

8. Duotos dvi kvadratinės matricos A ir B. Rasti trečią matricą C, kuri bus sudaryta iš atitinkamų matricų A ir B elementų sumos.
9. Duotos dvi vienodo dydžio matricos A ir B. Surasti kokios matricos A elementus, kurie yra didesni už atitinkamus matricos B elementus.
10. Parašykite programą, kuri pateiktų meniu. Meniu būtų būtų galima pasirinkti: ar išvesti nurodyto skaičiaus kubą, ar kvadratinę šaknį, ar faktorialą.

MOKYKLŲ TOBULINIMO CENTRAS

Ačiū už dėmesį