3 tema. Įvedimas, išvedimas

C++ kalboje įvedimo/išvedimo operacijoms yra įvestos srautų klasės, tačiau yra vartojamos ir klasikinės įvedimo/išvedimo bibliotekos. Tokios bibliotekos yra trys: **stdio.h** (klasikinis buferizuotas I/O – **i**nput/**o**utput), **io.h** (žemo lygio nebuferizuotas UNIX standarto I/O), **conio.h** (specialios konsolės I/O funkcijos). Laboratoriniame darbe analizuojamos tik bibliotekos **stdio.h** funkcijos.

Aprašant I/O operacijas, vartojama "srauto" sąvoka. Srautas - tai abstrakcija, leidžianti atsiriboti nuo konkretaus I/O įrenginio (disko, juostos, terminalo). Srautai būna dviejų tipų:

✧ Tekstinis srautas - į eilutes, atskiriamas specialiais ženklais (CR - carriage return, LF - line feed), suskaidyta simbolių seka. Tekstinio srauto informacija nebūtinai yra identiška duomenims matomiems konkrečiame atvaizdavimo įrenginyje (printeryje, displėjuje), nes dalis simbolių (CR, LF) yra skirta ne vaizdavimui, o srauto perdavimo valdymui;

✧ Dvejetainis srautas - baitų seka, tiksliai atitinkanti informaciją konkrečiame įrenginyje.

Kiekviena programa automatiškai atidaro 5 standartinius tekstinius srautus:

stdin (įvedimas), **stdout** (išvedimas),

stderr (pranešimai apie klaidas), **stdaux** (papildomas),

stdprn (spausdintuvas).

Jie susiejami su standartiniais sisteminiais I/O įrenginiais. Dažniausiai srautai **stdin**, **stdout, stderr** yra susiejami su konsole (klaviatūra ir ekranas). Standartinį I/O įrenginį galima pakeisti (perskirti) DOS priemonėmis.

Standartinių srautų (klaviatūros ir ekrano) valdymui vartojamos funkcijos **printf** ir **scanf**. Jų prototipai:

int printf (const char *Šablonas, [<Reikšmių sąrašas>]);

int scanf(const char *Šablonas, <Atminties laukų sąrašas>);

Šablomas - tai simbolių eilutė su įterptais rašomų arba skaitomų duomenų formatais, kurie aprašo duomenims skiriamo lauko struktūrą arba jų interpretavimo būdą. Funkcijoje **scanf** skaitomiems duomenims skiriami atminties laukai yra nurodomi adresais. Tipinė formato struktūra yra tokia:

% [<Lauko dydis>][.<Tikslumas>]<Duomenų tipas>

Duomenų tipai yra žymimi raidėmis. Pagrindiniai tipai yra tokie:

d arba **i** - sveikasis, **o** - aštuntainis sveikasis,

x - šešioliktainis sveikasis, **f** - slankaus kablelio,

e - rodiklinė forma, **c** - simbolinis tipas,

p - rodyklė.

Susiejant srautą su konkrečiu failu, yra vartojamos failų rodyklės, kurios nurodo tvarkomų failų parametrų saugojimui skirtas **FILE** tipo struktūras. FILE tipas apibrėžtas faile **stdio.h**. Failo rodyklės aprašo sintaksė:

FILE *<Rodyklė>;

Failo rodyklė inicializuojama funkcija **fopen**(), kurios prototipas:

FILE *fopen(char *failo_vardas, char *naudojimo_būdas>);

Pagrindinių buferizuoto I/O funkcijų prototipai:

```
/* Jei nėra kitokios pastabos, klaidą nurodo funkcijos reikšmė EOF */
int   fclose(FILE *rod);                 //      Failo uždarymas
int   putc(int simbolis, FILE *rod);     //  Simbolio rašymas
int   getc(FILE *rod);                   //  Simbolio skaitymas
int   putw(int sk, FILE *rod);           //  Skaičiaus rašymas
int   getw(FILE *rod);                   //  Skaičiaus skaitymas
          // Skaito eilutę s iš n simbolių, klaidą rodo reikšmė NULL
char   *fgets(char *s, int n, FILE *rod);
int   fputs(const char *s, FILE *rod);     //  Eilutės s rašymas
/*   Siunčia n po t baitų turinčių blokų iš failo rod į buferį buf.
Funkcijos reikšmė rodo perskaitytų blokų skaičių                  */
```

int fread(void *buf, int t, int n, FILE *rod);

/* Siunčia n po t baitų turinčių blokų iš buferio buf į failą rod. Funkcijos reikšmė rodo perskaitytų blokų skaičių */

int fwrite(void *buf, int t, int n, FILE *rod);

/* Į failą r, panaudojant šabloną f, rašomi sąrašo elementai. Šablonas ir argumentų sąrašas sudaromi taip pat, kaip ir funkcijai printf */

int fprintf(FILE *rod, const *char f [,<sąrašas>]);

/* Skaitymas pagal šabloną. Funkcijos reikšmė - sėkmingai perskaitytų elementų skaičius */

int fscanf(FILE *rod, const *char f [,<sąrašas>]);

int feof(FILE *rod); // Failo pabaiga - nenulinė funkcijos reikšmė

int ferror(FILE *rod); // Klaida - nenulinė funkcijos reikšmė

int remove(const *char vardas); // Šalinamas failas

Failų apdorojimo programoms apdorojamų failų vardai gali būti perduodami pagrindinės funkcijos parametrais. Funkcijos prototipas:

int main(int n, char *argv[ ]);

Čia **n** - argumentų skaičius, o **argv** - argumentams skirto eilučių masyvo rodyklė. Argumentų reikšmės yra nurodomos programos iškvietimo komandoje ir yra atskiriamos tarpais. Nulinė argumentų masyvo eilutė - tai pačios programos pavadinimas.

3.1 pratimas. Naudodamiesi aplinkos pagalbine informacija (Ctrl+F1) išsiaiškinkite, kaip sudaryta struktūrizuoto tekstinio failo DUOM.TXT papildymo programa. Failo eilutėse yra saugomi duomenys apie įvairiems asmenims priklausančius telefonus. Duomenų elementams yra skiriamos tokios eilučių atkarpos: pavardė - [1, 20], vardas - [21, 40], telefono numeris - [41,50]. Sukurkite tokios struktūros failą ir patikrinkite programą. Programos vykdymo pabaigos sąlygą išsiaiškinkite nagrinėdami jos tekstą.

```
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
      FILE   *failas;
      char   buf[50],   e1[20],   e2[20],   e3[10];
      const  char   vardas[] = "DUOM.TXT";
      int    Init (const char *, char *);
   void   Duomenys( FILE * );
   void   Rodyti  ( FILE * );
//---------------------------------------------------
   main() {
  buf[0]= '\0';
                            // Failo paruošimas darbui
  if ( !Init(vardas, "a") ) exit(1);
               // Duomenys įvedami klaviatūra į failą
  Duomenys( failas ); fclose( failas );
                        // Failo paruošimas skaitymui
  if (!Init (vardas,"r"))  exit(1);
  Rodyti( failas );     // Duomenys siunčiami į ekraną
  fclose(failas);
  getch();
} //---------------------------------------------------
int Init (const char *v, char *mode) {
// Naujo failo atidarymas skaitymui/rašymui
//   arba seno papildymui
if ((failas = fopen(v, mode)) == NULL) { // Failo nebuvo
                                // Naujo sukurti nepavyko
if ((failas = fopen(v, "w+")) == NULL)
 { fprintf(stderr,
         "Atidaryti failą rašymui/skaitymui nepavyko.\n");
   return 0; /* Klaidos pozymis */    }
  else return 1; }  // Naujai sukurtas failas
 else   return 2;   // Surastas failas ir atidarytas
} //---------------------------------------------------
void   Duomenys( FILE * failas){
 int i;
 while (buf[0] != 'q')  {
// Ivedimo pabaiga nurodoma eilute, kurios pirmas simbolis yra
// 'q'
  buf[0]= '\0';
      // Strukturizuotos eilutes sudarymo ir radymo ciklai
  printf("Nurodykite pavarde, varda ir telefono numeri:\n");
  scanf("%s %s %s", e1, e2, e3);
  strcat( buf,  e1);
  for (i= strlen(buf); i<20; i++)   strcat(buf, " ");
    strcat(buf, e2);
  for (i= strlen(buf); i<40; i++)   strcat(buf, " ");
    strcat(buf, e3);      strcat(buf, "\n");
  if (buf[0] != 'q')
```

```
    fwrite (buf, strlen(buf), 1, failas);
 } } //-----------------------------------------------
void   Rodyti  ( FILE * ){
  while (fgets(buf, strlen(buf)+1, failas))
        printf("%s",buf);
}
```

✓Sudarykite pagalbinę funkciją buferinio kintamojo **buf** papildymui formatuotais laukais.

✓Pakeiskite programoje blokų rašymo komandą **fwrite** eilučių rašymo komanda fputs.

✓Pakeiskite programą taip, kad jos darbą būtų galima nutraukti įvedus vieno simbolio eilutę: "Q" arba "q". Norint tai padaryti, reikia iš klaviatūros perskaityti iš karto visą eilutę ir tik po to ja skanuoti su funkcija **sscanf**.

pratimas. 3.2 Panaudodami duomenų srautų apdorojimo klasę **`ofstream`** įvedame duomenis į matricą ir atlikę veiksmus (sukeičiami vietomis stulpeliai su didžiausia ir mažiausia matricos reikšme) išvedame rezultatus.

```
#include <fstream.h>
#include <stdio.h>
#include <stdlib.h>
#include <iomanip.h>
const   num = 15;
int     Kiek = 15;
typedef int   mas [15];

                                  // Duomenų skaitymas
void read (char *is, int a[num][num],  int &n );
                                        //
Spausdinimas
void print(char *os, int a[num][num],  int n,
           char *text );
                                  // Didžiausio paieška
int  max  (int a[num][num], int n );
                                  // Mažiausio paieška
int  min  (int a[num][num], int n );
                                // Stulpelių sukeitimas
void swap (int a[num][num], int n,
           int mine, int maxe );
                                // Failų paruošimas
int  test (char *is,  char *os    );
//-------------------------------------------------
void main(void ) {
  int a[num][num], n;
                                // Failų vardams saugoti.
  char inbuff[12], outbuff[12];
           // Rodyklės į failų vardus *ifile ir *ofile
cout<<"Koks duomenu failo vardas?\n";
char *ifile = gets( inbuff  );
  cout<<"Koks rezultatu failo vardas?\n";
  char *ofile = gets( outbuff );
  if (test (ifile, ofile )){
      cout<<"Klaida atidarant failus";  exit( 0 ); }
  read( ifile, a, n); // Duomenų skaitymas į matricą
                      // Duomenų spausdinimas į failą
  print(ofile, a, n, "Pradiniai duomenys");
                      // Stulpelių sukeitimas vietomis
  swap( a, n, min( a, n ),  max( a, n ));
                      // Rezultatų spausdinimas
  print(ofile, a, n, "Rezultatai");
  cout<<"Pabaiga\n\a";
}//-------------------------------------------------
void read ( char *is, int a[num][num],  int &n ){
   ifstream infile(is);   // Duomenų failo atidarymas
   infile >> n;           // Įvedama n reikšmė
   n--;                  // Matricos įvedimas
   for ( int i = 0; i <= n; i++)
      for ( int j = 0; j <= n; j++ )
          infile >> a[i][j];
   infile.close();              // Failo uždarymas
}//-------------------------------------------------
void print (char *os, int a[num][num],  int n,
            char *text ){
           // Rezultatų failas paruošiamas papildymui
  ofstream offile ( os, ios::app );
offile << '\n' << text <<'\n' << "     ";
for ( int j = 1; j <= n+1; j++ )
                                 offile << setw(5)<< j;
   offile << " \n    |";
for ( int j = 0; j <= n; j++ )  offile << "-----";
offile << " \n";
  for ( int i = 0; i <= n; i++){
    offile << setw( 3 ) << (i+1) <<" |";
    for ( int j = 0; j <= n; j++ )
```

```
          offile << setw(5)<< a[i][j];
    offile <<" \n";             }
}//-------------------------------------------------
         // Randamas stulpelis su didžiausios reikšme
int  max (int a[num][num], int n ){
  int m = -1000,  p = -1;   // p - stulpelio numeris.
   for ( int i = 0; i <= n; i++)
      for ( int j = 0; j <= n; j++ )
          if ( a[i][j] > m ) { m = a[i][j];  p = j; }
  return p;
}//-------------------------------------------------
int  min      ( int a[num][num], int n ){
                     // Randamas mažiausios reikšmės
   int m = 1000,  p = -1;     // stulpelio numeris p.
   for ( int i = 0; i <= n; i++)
      for ( int j = 0; j <= n; j++ )
          if ( a[i][j] < m ) { m = a[i][j];  p = j; }
  return p;
}//-------------------------------------------------
void swap     ( int a[num][num], int n,
                int mine, int maxe ){
 int p;         // Sukeičiami du stulpeliai vietomis.
   for ( int i = 0; i <= n; i++){
     p = a[i][mine]; a[i][mine] = a[i][maxe];
     a[i][maxe] = p;}
}//-------------------------------------------------
int  test     ( char *is,  char *os    ){
                     // Patikrinamas duomenų failas.
  ifstream duomenys( is );
if (!duomenys)  return 1;
  duomenys.close();
                    // Patikrinamas rezultatų failas.
ofstream rezultatai ( os );
if (!rezultatai)  return 1;
           // Į rezultatų failą išvedamas pranešimas.
  rezultatai << "Matrica \n";    rezultatai.close();
  return 0;
}//-------------------------------------------------
```

Duomenų failas

```
4
15  1  5    6
2  25  6    7
5  -5  5  -45
1   2  3    4
```

Rezultatų failas

```
Matrica

Pradiniai duomenys
          1    2    3    4
     |--------------------
   1 |   15    1    5    6
   2 |    2   25    6    7
   3 |    5   -5    5  -45
   4 |    1    2    3    4

Rezultatai
          1    2    3    4
     |--------------------
   1 |   15    6    5    1
   2 |    2    7    6   25
   3 |    5  -45    5   -5
   4 |    1    4    3    2
```

4 tema. Struktūros

Struktūrų aprašų sintaksė:

```
struct <struktūrinio tipo vardas>
        {   <laikų aprašu sąrašas> }
           [<kintamųjų sąrašas>];
```

Laukų aprašų sąrašų elementai atskiriami kabliataškiais, o kintamųjų sąrašo elementai – kableliais. Jei apraše nėra kintamųjų sąrašo, jis apibrėžia tik naujų struktūrinių duomenų tipą, tačiau jo realizacijoms atmintyje vietos neskiria. Atskiro struktūros tipo apibrėžimo ir realizavimo pavyzdys:

```
                                        // Apibrėžimas
struct telefonas {           char     vardas[30];
                  unsigned  longint  tel;       }
                                      // Realizavimas
struct telefonas  Tel,  *P, TelMas[ 100 ];
```

Kreipiantis į struktūrų elementus vartojami sudėtiniai vardai:

<struktūros vardas>.<lauko vardas>

Galimos rodyklės į sruktūrų tipo reikšmes:

```
P = &Tel;        // struct  telefonas  *P;
```

Laukų vardų užrašymo, panaudojant rodykles, pavyzdžiai:

```
(*P)->tel,                  P->tel
```

4.1 pratimas. Klaviatūra įvedamas žmonių sąrašas spausdinamas lentele faile.

```
//  Strukturos
#include <stdio.h>
#include <stdlib.h>
#include <iomanip.h>
#include <conio.h>
struct   zmogus {  char   vardas [10];
                   char   pavarde[10];
                   int    gim_met;
                   float  ugis;        };
FILE *F;
const char Duomenys[] = "Prat17.rez";

                                  // Failo atidarymas
int Atidaro ( const char *, char *);
        // Duomenys įvedami klaviatūra ir spausdinami faile
void Lentele();
//---------------------------------------------------
void main(void ) {
  if (Atidaro( Duomenys, "w")) exit(0);
  Lentele();
  fclose( F );
  cout<<"Pabaiga\n\a";
  getch();
}//---------------------------------------------------
int Atidaro( const char *Vardas, char *Forma){
  if (( F = fopen( Vardas, Forma )) == NULL )
   { cout<< "Failas neatidarytas"; return 1;}
  else return 0;
}//---------------------------------------------------
 void Lentele(){
  struct zmogus A;  int n;
  cout<<"Kiek zmoniu bus ?\n";  cin>> n;
fprintf(
F,"................................................\n");
fprintf( F,":  Vardas  :  Pavarde  : gim. metai : ugis  :\n");
fprintf( F,"................................................\n");
 while ( n-- ){
  cout<<" Iveskite varda        \n";  cin>> A.vardas ;
  cout<<" Iveskite pavarde      \n";  cin>> A.pavarde;
  cout<<" Iveskite gimimo metus \n";  cin>> A.gim_met;
  cout<<" Iveskite ugi          \n";  cin>> A.ugis   ;
  fprintf( F, ":%12s  :%12s :   %5d    :%8.3f :\n",
           A.vardas, A.pavarde, A.gim_met, A.ugis);
  }
fprintf( F,"................................................\n");
  fclose( F );
}//---------------------------------------------------
```

Programos darbo rezultatai faile `Prat17.rez`:

```
................................................
:    Vardas    :   Pavarde   : gim. metai :  ugis   :
................................................
:      Petras  :   Petraitis :    1933    : 203.750 :
:      Jurgis  :    Jurgelis :    1582    : 125.580 :
:       Kazys  :     Kazelis :    1258    : 125.250 :
................................................
```

Struktūrų masyvai sudaromi taip pat, kaip ir paprastų duomenų tipų. Rekomenduojama sukurti duomenų tipą, po to jį naudoti.

4.2 pratimas. Klaviatūra įvedami duomenys surašomi struktūrų masyve. Po to jie spaudinami lentele.

```
//  Strukturos
#include <stdio.h>
#include <stdlib.h>
#include <iomanip.h>
#include <conio.h>
struct   zmogus {  char   vardas [10];
                   char   pavarde[10];
                   int    gim_met;
                   float  ugis;
                                    };
FILE *F;
const char Duomenys[] = "Prat18.rez";

                                  // Failo atidarymas.
int Atidaro ( const char *, char *);
void Ivesti( zmogus *, int *); // Įvedimas klaviatūra.
```

```
void Lentele(zmogus *, int  ); // Išvedimas lentele.
//---------------------------------------------------
void main(void ) {
  zmogus A[10];   int n = 0;
  Ivesti( A, &n);
  if (Atidaro( Duomenys, "w")) exit(0);
  Lentele( A, n);
  fclose( F );   cout<<"Pabaiga\n\a";
  getch();
}//---------------------------------------------------
void Ivesti( zmogus *B, int *n){
   int i = 0;
   cout<<"Kiek zmoniu bus ?\n";  cin>> *n;
 while( i < *n ){
  cout<<"********************\n";
  cout<<(i+1) <<"-ojo zmogaus duomenys:\n";
cout<<" Iveskite varda        \n";  cin>> B[i].vardas ;
cout<<" Iveskite pavarde      \n";  cin>> B[i].pavarde;
cout<<" Iveskite gimimo metus \n";  cin>> B[i].gim_met;
cout<<" Iveskite ugi          \n";  cin>> B[i].ugis   ;
  cout<<"Aciu \n";        i++; }
}//---------------------------------------------------
int Atidaro( const char *Vardas, char *Forma){
  if (( F = fopen( Vardas, Forma )) == NULL )
   { cout<< "Failas neatidarytas"; return 1;}
  else return 0;
}//---------------------------------------------------
 void Lentele( zmogus *C, int n){
  int i = 0;
fprintf( F,".........................................\n");
fprintf( F,":  Vardas  :  Pavarde  : gim. metai : ugis  :\n");
fprintf( F,".........................................\n");
  while ( i < n ){
    fprintf( F, ":%12s  :%12s :   %5d    :%8.3f :\n",
  C[i].vardas, C[i].pavarde, C[i].gim_met, C[i].ugis);
    i++; }
fprintf( F,".........................................\n");
  fclose( F );
}//---------------------------------------------------
```

4.3 pratimas. Klaviatūra įvedami duomenys surašomi į tipizuotą failą. po to jie skaitomi iš to failo ir parodomi ekrane.

```
//  Strukturos
#include <stdio.h>
#include <stdlib.h>
#include <iomanip.h>
#include <conio.h>
FILE *failas;
const char Rezultatai[] = "Prat19.rez";
struct   telefonas {        char   vardas [10];
                            char   pavarde[10];
                            long   tel;
            } T;
                                    // Failo atidarymas
int Atidaro ( const char *, char *);
void Raso();
void Skaito();
//---------------------------------------------------
void main(void ) {
  if ( Atidaro( Rezultatai, "w") ) exit (0);
  Raso();    fclose( failas );
  if ( Atidaro( Rezultatai, "r"))  exit (0);
  Skaito();  fclose( failas );
  cout<<"Pabaiga\n\a";   getch();
}//---------------------------------------------------
void Raso(){
   char buf[50] = "";
   printf("Programos nutraukimas - eilute 'q'\n");
 while( buf[0] != 'q' ){
  printf("Nurodykite varda, pavarde it telefono numeri:\n");
     gets( buf );
     if( buf[0] != 'q'){
       sscanf( buf, "%s%s%ld",
                     T.vardas, T.pavarde, &T.tel );
   fwrite( &T, sizeof( struct telefonas ), 1, failas );
   }   }
}//---------------------------------------------------
int Atidaro( const char *Vardas, char *Forma){
  if (( failas = fopen( Vardas, Forma )) == NULL )
   { cout<< "Failas neatidarytas"; return 1;}
  else return 0;
}//---------------------------------------------------
 void Skaito(){
  printf( ".....................................\n");
  printf( ":   Vardas   :  Pavarde  : telefonas  :\n");
  printf( ".....................................\n");
  while(
    fread( &T, sizeof( struct telefonas), 1, failas))
      printf( ":%10s  :%10s : %10ld :\n",
```

```
            T.vardas, T.pavarde, T.tel);
  printf( ".....................................\n");
}//-------------------------------------------------
```

✓ Pertvarkykite programą, kad pagrindinėje funkcijoje pagal vartotojo pageidavimus būtų galima pasirinkti tokius veiksmus: duomenų failo turinio išvedimas ekrane, naujo failo formavimas arba esančio failo papildymas. Programos šakojimui vartokite operatorių `case`.

✓ Papildykite programą failo rikiavimo pagal pavardes alfabeto tvarka funkcija. Rikiuojamo failo duomenys iš pradžių turi būti perrašomi į įrašų masyvą ir tik po to rikiuojami. Lyginant eilutes turi būti vartojamos ne santykiooperacijos, bet bibliotekos `string.h` funkcijos arba makrokomandos: `stricmp`, `strnicmp`, `strncmpi`, `strcmpi`.

Kuriant sudėtingesnes programas tikslinga žinoti keletą įdomesnių funkcijų savybių. Klaboje C++ realizuota labai efektyvi funkcijų savybė – polimorfizmas, kuri dar vadinama daugiavariantiškumu arba funkcijų persidengimu (overloading). Senose C++ kalbos versijosepolimorfinių funkcijų aprašai turi būti pažymimi baziniu žodžiu `overload`. Kalbos realizacijose Bordland C++ tai daryti nebūtina. Pavyzdyje demonstruojama išvedimo ekrane polimorfinė funkcija `print`, kurios darbas priklauso nuo kreipinyje užrašyto duomenų tipo.

```
//  Polimorfizmas
#include <stdio.h>
#include <conio.h>

// overload print;    // Bordland C galima praleisti.
inline void print( int    x ){ printf("%d\n"   , x); }
inline void print( double x ){ printf("%5.2f\n", x); }
inline void print( char  *x ){ printf("%s\n",    x); }

void main(void ) {
  print( 3.14 );         // Slankaus kablelio skaicius.
  print( 15   );            // Sveikas skaicius.
  print( "Simboliu eilute"); // Simboliu seka.
  getch();
}
```

Čia baziniu žodžiu inline pažymėtos įterpiamos funkcijos. Iterpiamos funkcijos yra makrokomandų analogai – pirminis procesorius įrašo funkcijos tekstą kiekvieno kreipinio į ją vietoje. Jei funkcijų tekstai trumpi, toks funkcijų tekstų įterpimas padidina programų darbo greitį.

Dar viena labai naudinga funkcijų savybė – argumentų parinkimas pagal nutylėjimą.

```
//  Argumentai
#include <stdio.h>
#include <conio.h>

struct upe { char *vardas;
             char *salis; } x, y, z, k;
//-------------------------------------------------
void Rodo( struct upe *d, char *a = NULL,
                          char *b = NULL)
         { d->vardas = a;  d->salis = b; }
//-------------------------------------------------
void main(void ) {
    Rodo( &x );
    Rodo( &y, "Nemunas" );
    Rodo( &z, "Merkys", "Lietuva" );
    Rodo( &k, "Nilas" );
 printf( "%10s  %10s \n", x.vardas, x.salis );
 printf( "%10s  %10s \n", y.vardas, y.salis );
 printf( "%10s  %10s \n", z.vardas, z.salis );
 printf( "%10s  %10s \n", k.vardas, k.salis );
   getch();
}
```

Ekrane matysime:

```
 (NULL)    (NULL)
Nemunas   (NULL)
Merkys   Lietuva
Nilas     (NULL)
```