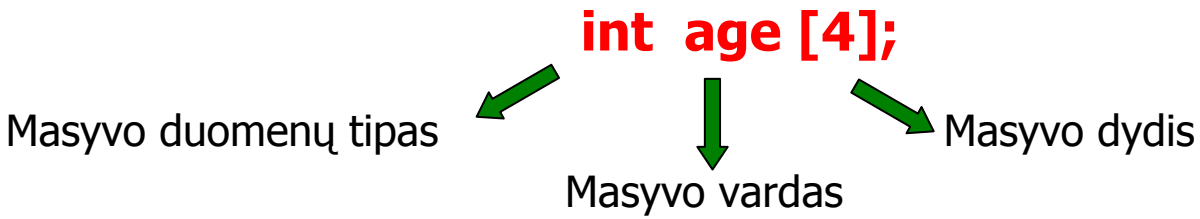
C programavimo kalba

6 paskaita

(Masyvai, rodyklės ir nuorodos)

Masyvo pavyzdys

```
#include <stdio.h>
int main ()
{ int age[4];                    // sukuriamas masyvas iš 4 int tipo elementų
    for (int  j=0; j<4; j++)
      { printf ("Iveskite skaiciu: ");
        scanf ("%d", &age[j]);
       }
    for ( j=0; j<4; j++)
       printf ("Ivesktas skaicius: %d ", age[j]);
 return 0;

}
```



Duomenų įvedimas

Masyvo užpildymas ir jo elementų skaitymas dažniausiai atliekamas **ciklo** pagalba. Svarbu neužmiršti, kad masyvo pirmo elemento indeksas yra [0].

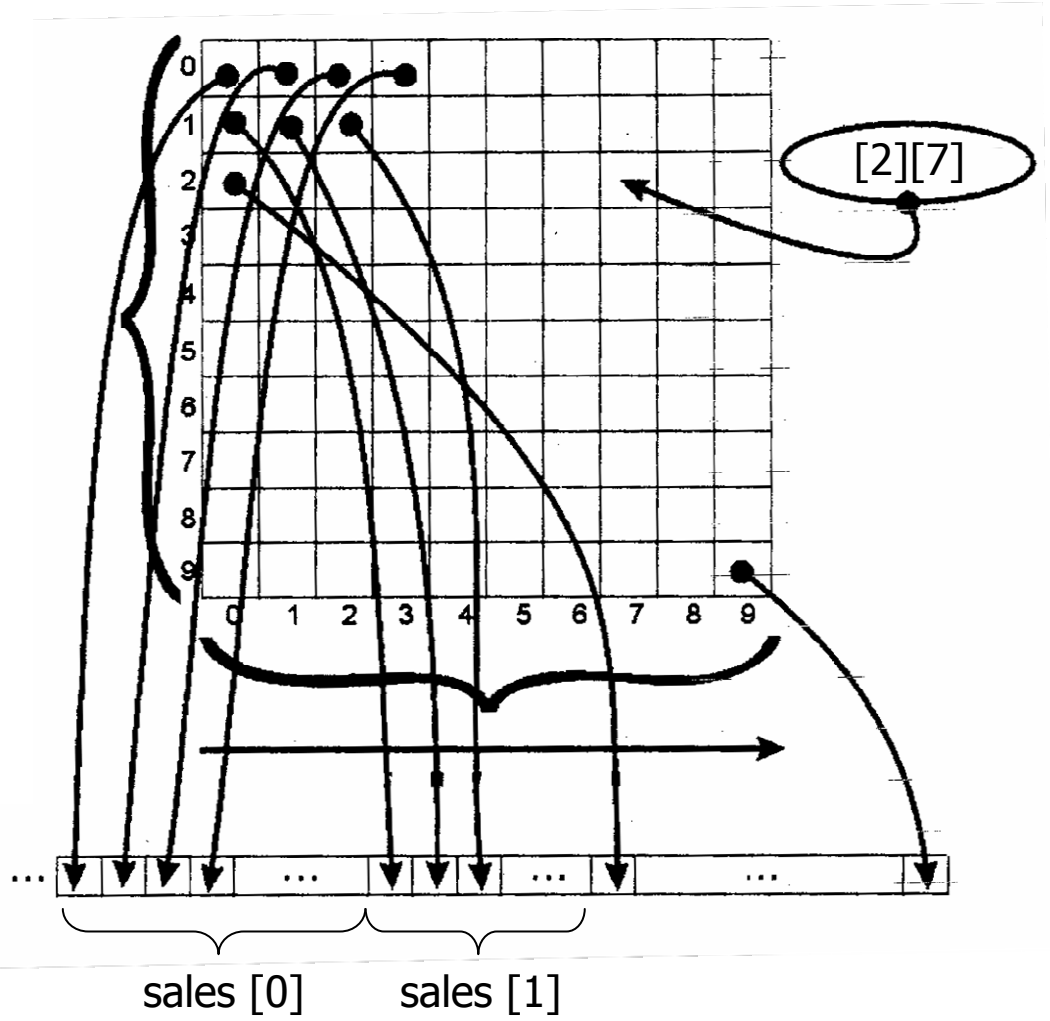
```
...
double a[20];
for(i=0; i < 20; i++)     /*masyvo užpildymas*/
    scanf("%lf", &a[i]);
for(i=0; i < 20; i++)
    printf("%f", a[i]); /*masyvo elementų spausdinimas*/
```

Svarbu: užpildymui nurodomas masyvo adresas: &a[i].

Masyvo dydį galima apibrėžti ir per konstantas:

```
 ...
#define MAX 45  // const int MAX = 45;     galima ir taip apibrėžti konstantą
 main( )
{
  int a[MAX];
}
```

Dvimačiai masyvai



Dvimačiai masyvai

```
#include <stdio.h>
const int Salesman = 4;
const int Menuo = 3;

void main()
{ int d, m;
   float sales [Salesman][Menuo];          // tas pats kaip    sales[4][3]
    for (d = 0;  d < Salesman;  d++)
       for(m = 0; m < Menuo; m++)
       { printf("Iveskite %d pardavejo %d menesio pardavimu suma:", d+1, m+1);
         scanf("%f", &sales[d][m]);
        }
   printf("\n\t\t\t Menuo\n\t\t       1        2        3 \n");
    for (d = 0;  d < Salesman;  d++)
       {  printf("\nPard. %d  pardavimai: ", d+1);
             for (m = 0; m < Menuo; m++)
                  printf("  %f ", sales[d][m]);
       }
 printf("\n"); }
```

Dvimačių masyvų inicializacija

Pradinių reikšmių priskyrimas masyvo elementams (inizializacija)

int arr[ ] = { 12, 4, 5 }; // vienmatis 3 elementų masyvas

int masyvas [3] [3] = { 1, 2, 3, 4, 5, 6, 7, 8, 9};

int masyvas [3] [3] = { {1, 2, 3} , {4, 5, 6} , {7, 8, 9} };

int masyvas [] [3] = { {1, 2, 3} , {4, 5, 6} , {7, 8, 9} };

```
void main( )
{ int masyvas [ ] [3] = { {1, 2, 3} , {4, 5, 6} , {7, 8, 9} };
   int i, j;
    for(i=0; i < 3; i++)
      {  for(j=0; j < 3; j++)
           printf("  %d ",  masyvas[i] [j]);
           printf ("\n");
       }
}
```

Masyvo perdavimas funkcijai

Masyvai gali būti naudojami, kaip funkcijų argumentai.

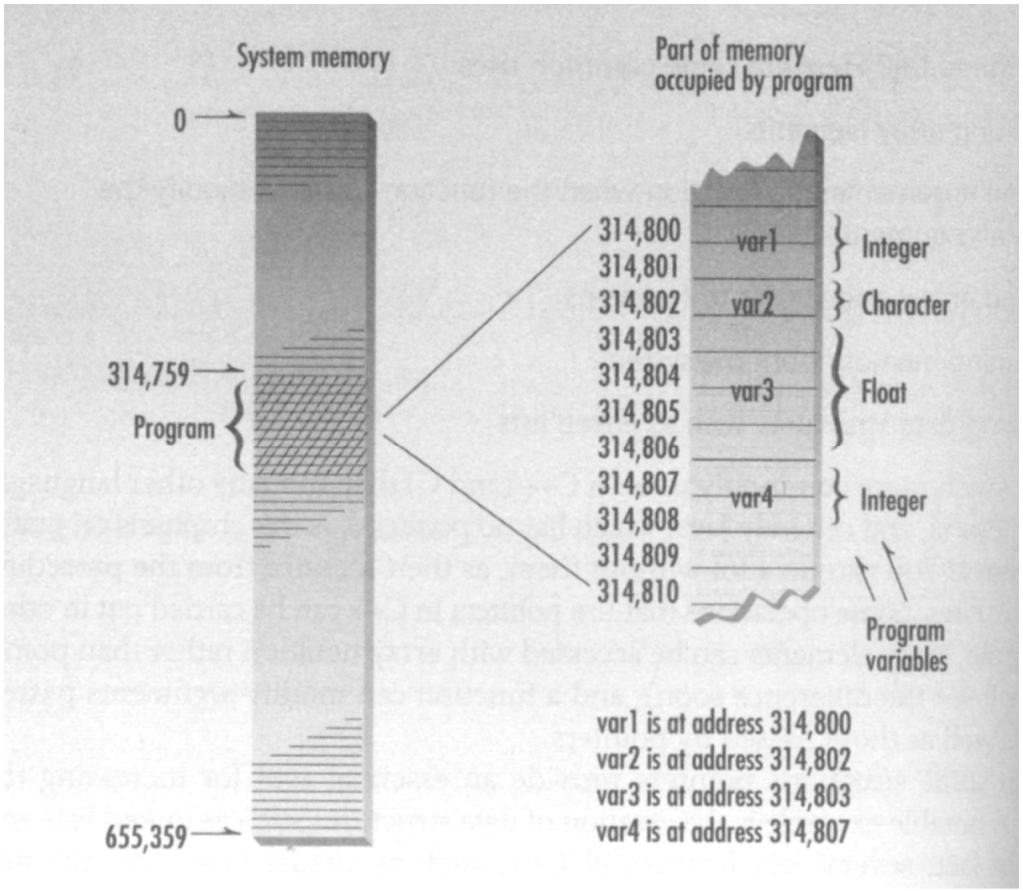
```
#include <stdio.h>
void display ( int [3][3] );

void main( )
{ int masyvas [3] [3] = { {1, 2, 3} , {4, 5, 6} , {7, 8, 9} };
   display (masyvas);
}
```

```
void display (int array [3] [3])
{ int i, j;
    for( i=0; i < 3; i++ )
      {  for( j=0; j < 3; j++ )
           printf("  %d ",  array [i] [j]);
           printf ("\n");
       }
}
```

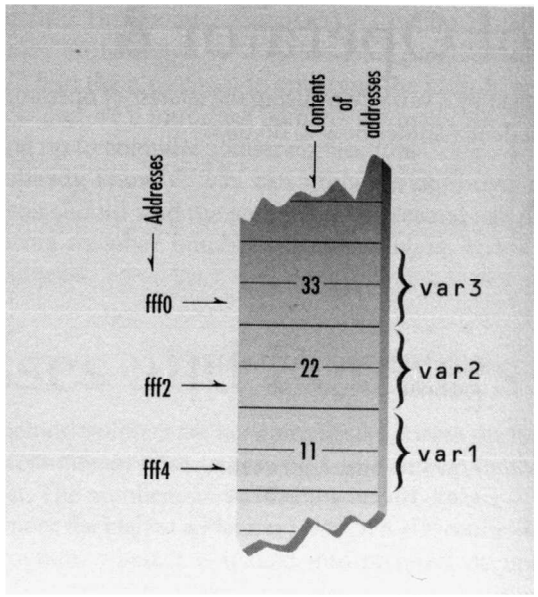
Rodyklės (pointers) ir adresai

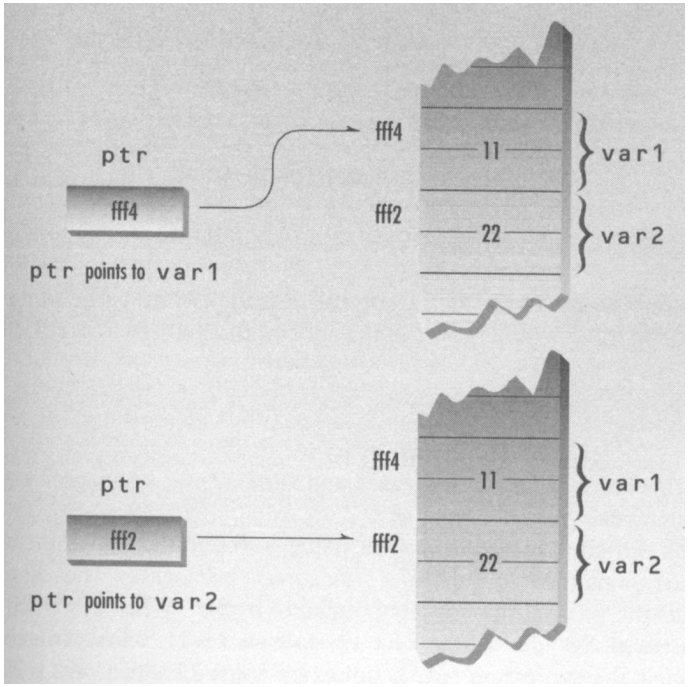
Kiekvienam programoje aprašytam kintamajam kompiliatorius skiria atminties lauką, kurį apibūdina visa grupė parametrų:

Adresas, *Lauko dydis,* *Saugomų duomenų tipas ir reikšmė*



Rodyklės (pointers) ir adresai





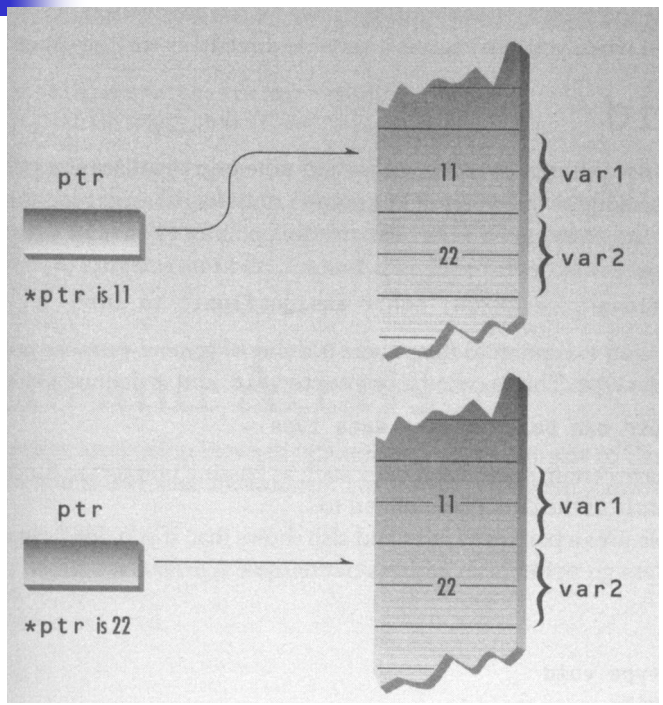
Kintamasis – tai atminties segmentas turintis vardą. Pvz. var1 = 11

Kompiuteris pasiekia savo atmintį ne per kinamųjų vardus, o per **adresus**.

Rodyklė – tai kintamasis, kuriame saugomas atminties adresas.
Kitaip dar sakoma, kad rodyklė rodo į kintamąjį, kurio adresą ji saugo.

Rodyklė turi būti apibrėžiama, kaip ir bet kuris kitas kintamasis, pvz. **int *ptr;**

Rodyklės




```
int  *ptr0,  var0;
float  *ptr1,  var1;
double  *ptr3,  var3;
```

rodyklės kintamieji

Su rodyklėmis naudojami du operatoriai:

- **&** - grąžina kintamojo adresą;
- ***** - parašyta prieš rodyklę, grąžina kintamojo reikšmę (*de-referencing* operator);

```
int var1 = 11;  int var2 = 22;
int *ptr = &var1;
ptr = var1;        // klaida
var2 = *ptr;       // dabar var2 reikšmė 11;
```

Rodyklės

Pagrindinės taisyklės, kurias reikia atsiminti apie rodykles:

- Apibrėžiant rodyklę rašoma * prieš kintamojo vardą.
- Norint gauti kintamojo adresą, reikia prieš kintamojo vardą rašyti **&** simbolį.
- Norint gauti kintamojo, į kurį rodo rodyklė, reikšmę, prieš rodyklės pavadinimą rašoma * .

```
#include <stdio.h>
  void main()
 { int x, *p;
        p = &x;                    /* inicializuojama rodyklė */
       *p = 0;                     /* kintamajam x priskiriamas 0 */
   printf( "x yra %d\n", x);
   printf("*p yra %d\n", *p);
      *p += 1;                     /* kintamasis x padidinamas 1 */
   printf("x yra %d\n", x);
      (*p)++;                      /* kintamasis x padidinamas 1 */
   printf("x yra %d\n", x); }
```

Rodyklės ir masyvai

```
#include <stdio.h>
#define ARS 5
void main(){
      int ar[ARS];
      for(int i = 0; i < ARS; i++)
     {    ar[i] = i+3;
          printf("ar[%d] = %d\n", i, ar[i]);
      }
}
```

```
#include <stdio.h>
#define ARS 5
void main(){
      int ar[ARS];
      for(int i = 0; i < ARS; i++)
     {  ar[i] = i+3;
          printf("ar[%d] = %d\n", i, *(ar+i) );
      }
}
```

Masyvo pavadinimas – tai rodyklė į 1-ąjį masyvo elementą, kurios reikšmė 1-ojo elemento adresą.

```
int array[10], *ptr;
ptr = array;
```

Nuoroda (reference)

Nuoroda – tai specialus duomenų tipas, kuris dar laikomas užslėpta rodykle, kurią naudojant automatiškai dirba su kintamojo reikšme. Dar sakoma, kad nuoroda, tai kitas kintamojo vardas arba pseudonimas.

Apibrėžiant nuorodą, jai turi būti iš karto nurodytas kintamojo vardas, į kurį sukuriama nuoroda.

Sintaksė

tipas &nuorodos_vardas = kintamojo vardas;

Pavyzdys

```
char Raide = 'A';
char &ref = Raide;
```

```
int i = 0;
int &ref = i;
ref +=10;     // tas pats, kaip i +=10
```

Rodyklės ir funkcijos

Duomenys funkcijoms perduodami per parametrų sąrašus tokiais būdais:

- kopijuojant **reikšmę**;
- naudojant **rodykles**;
- naudojant **nuorodas**.

```
#include <stdio.h>
void main()
{  void centm (double&);
   double var = 10.0;
   printf ("var = %lf  cnt\n", var);
   centm (var);
   printf ("var = %lf  coliu\n", var);
}

void centm(double &v)
{ v = v/2.54; }
```

```
#include <stdio.h>
void main()
{  void centm (double*);
   double var = 10.0;
   printf ("var = %lf  cnt\n", var);
   centm (&var);
   printf ("var = %lf  coliu\n", var);
}
void centm(double *ptr)
{ *ptr = *ptr/2.54; }
```

Masyvų perdavimas funkcijose

Masyvų perdavimui į funkcijas dažniausiai naudojamos rodyklės.

```
#include <stdio.h>
void centm (double*);
void main()
{  double array[5] = {10.0, 12.4, 3.5, 7.2, 22.4};
    centm (array);
    for (int i = 0; i  < 5; i++)
        printf ("array[%d] = %lf  coliu\n", i, array[i]);
}

void centm(double *ptr)
{ for (int j = 0; j < 5; j++)
      *ptr /=2.54;        // arba prt[j] /= 2.54;
       ptr++;
 }
```

Rodyklių aritmetika

Operacija	Pavyzdys
Lyginimas	p1 = = p2
Lyginimas ar nelygu	p1 != p2
Mažiau	p1 < p2
Mažiau arba lygu	p1 <= p2
Daugiau	p1 > p2
Daugiau arba lygu	p1 > = p2
Elementų skaičiaus skaičiavimas	p1- p2
Suskaičiuoja nuorodą nutolusią per **n** nuo duotosios	p1 + n; p1 – n