KAZALCI IN STRUKTURE

Deklaracija kazalca na strukturo:

**struct ime\_strukture \*ime\_kazalca;**

Primer:

struct podatki

{

char ime[20];

char priimek[25];

char naslov[50];

}; lahko tudi:

struct podatki oseba; **struct podatki oseba,\*poseba;**

struct podatki \*poseba;

Ko inicializiramo:

**poseba=&oseba;**

kazalec kaže na naslov **prvega byta prve komponente** deklariranestrukture.

Do vrednosti posamezne komponente strukture lahko pridemo na dva načina:

1. (\*poseba).ime /\* obvezna uporaba ( ) \*/

2. poseba -> ime /\* NE PIŠEMO ime strukture \*/

Primer 1:

Napiši program, ki prebere ime, priimek in naslov neke osebe in izpiše njen priimek in naslov.

rešitev a:

#include <iostream>

#include <cstdio>

using namespace std;

struct podatki

{

char ime[20];

char priimek[25];

char naslov[50];

};

int main()

{

struct podatki oseba,\*poseba;

poseba=&oseba;

gets(oseba.ime);

gets(oseba.priimek);

gets(oseba.naslov);

cout<<endl<< "Izpis imena in priimka osebe:";

cout<<**(\*poseba).ime** <<" " <<**(\*poseba).priimek**;

return 0;

}

rešitev b:

#include <iostream>

#include <cstdio>

using namespace std;

struct podatki

{

char ime[20];

char priimek[25];

char naslov[50];

};

int main()

{

struct podatki oseba,\*poseba;

poseba=&oseba;

gets(oseba.ime);

gets(oseba.priimek);

gets(oseba.naslov);

cout<<endl<< "Izpis imena in priimka osebe:";

cout<< poseba->ime<<" "<< poseba->priimek;

return 0;

}

Z operatorjem -> se dostopa tudi do naslova posamezne komponente strukture:

&poseba -> priimek

Primer 2:

Napiši program, ki prebere ime, priimek in naslov desetih oseb in izpiše vse osebe, ki živijo na iskanem naslovu.

rešitev a:

#include <iostream>

#include <cstdio>

using namespace std;

struct podatki

{

char ime[20];

char priimek[25];

char naslov[50];

};

int main()

{

struct podatki oseba[10],\*poseba;

int i;

char iskano[20];

**poseba=oseba;**

for(i=0;i<10;i++)

{

gets(oseba[i].ime);

gets(oseba[i].priimek);

gets(oseba[i].naslov);

}

gets(iskano);

for(i=0;i<10;i++)

{

if (strcmp(iskano,oseba[i].naslov)==0)

{

cout<<endl<< poseba->ime<<" "<< poseba->priimek;

poseba++;

}

}

return 0;

}

rešitev b:

#include <iostream>

#include <cstdio>

using namespace std;

struct podatki

{

char ime[20];

char priimek[25];

char naslov[50];

};

int main()

{

struct podatki oseba[10],\*poseba;

int i;

char iskano[20];

**poseba=oseba;**

for(i=0;i<2;i++)

{

gets(oseba[i].ime);

gets(oseba[i].priimek);

gets(oseba[i].naslov);

}

gets(iskano);

for(i=0;i<2;i++)

{

if (strcmp(iskano,oseba[i].naslov)==0)

{

cout<<endl<< **(\*(poseba+i)).ime<<" ";**

cout << **(\*(poseba+i)).priimek**;

}

}

return 0;

}

rešitev c:

#include <iostream>

#include <cstdio>

using namespace std;

struct podatki

{

char ime[20];

char priimek[25];

char naslov[50];

};

int main()

{

struct podatki oseba[10],\*poseba;

int i;

char iskano[20];

**poseba=&oseba[0];**

for(i=0;i<2;i++)

{

gets(oseba[i].ime);

gets(oseba[i].priimek);

gets(oseba[i].naslov);

}

gets(iskano);

for(i=0;i<2;i++)

{

if (strcmp(iskano,oseba[i].naslov)==0)

{

cout<<endl<< **poseba[i].ime<<" "<< poseba[i].priimek**;

}

}

return 0;

}