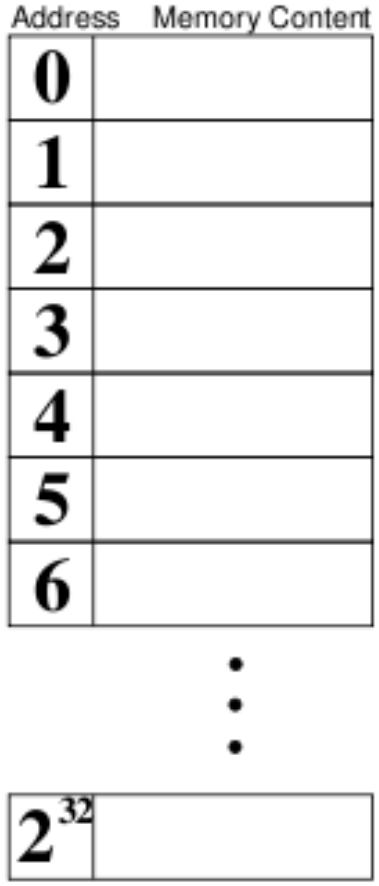
Pointeri i strukture u programskom jeziku C

—

jako kratki pregled

Pointeri

Memoriju računala možemo zamišljati kao jednodimenzionalno polje byteova, pri čemu svaki byte ima svoju adresu. Bitno je naglasiti da je (što se C-a tiče) 0 ilegalna adresa. Na donjoj slici vidimo shematski prikaz memorije na 32-bitnoj arhitekturi.



Kada deklariramo varijablu (npr. `double x;`) u memoriji se zauzme niz uzastopnih memorijskih lokacija potrebnih za spremanje deklarirane varijable (za `double` je potrebno 8 byteova). Adresu prve od memorijskih lokacija na kojima je zapisana varijabla `x` nazivamo adresa varijable `x`, a sadržaj memorijskih lokacija na kojima je zapisana varijabla `x` zovemo vrijednost varijable `x`. Tip varijable `x` daje nam informaciju o tome koliko je byteova potrebno za zapisivanje varijable `x` i na koji način interpretirati sadržaj tih lokacija.

Kada deklariramo varijablu (npr. `double x;`) u memoriji se zauzme niz uzastopnih memorijskih lokacija potrebnih za spremanje deklarirane varijable (za `double` je potrebno 8 byteova). Adresu prve od memorijskih lokacija na kojima je zapisana varijabla `x` nazivamo adresa varijable `x`, a sadržaj memorijskih lokacija na kojima je zapisana varijabla `x` zovemo vrijednost varijable `x`. Tip varijable `x` daje nam informaciju o tome koliko je byteova potrebno za zapisivanje varijable `x` i na koji način interpretirati sadržaj tih lokacija.
Prema tome, varijabla je simboličko ime za skupinu uzastopnih memorijskih lokacija i jedinstveno je određena svojim tipom i adresom.

Pointer (ili pokazivač) je varijabla čiju vrijednost interpretiramo kao adresu neke druge varijable. Radi toga je potrebno za svaki pojedini tip `T` imati posebni tip pointera `T*`, budući da moramo znati ne samo gdje počinje zapis varijable na koju naš pointer “pokazuje”, nego i koliko byteova je potrebno za zapis te varijable i kako interpretirati njenu vrijednost.

Pointerska aritmetika je dio C standarda koji omogućava da polje identificiramo s pointerom na prvi element polja. Točnije radi se o tome da je definirano "zbrajanje" pointera i cijelih brojeva ili preciznije - definiran je operator `+` na uređenom paru bilo kojeg pointerskog tipa i bilo kojeg cjelobrojnog tipa.
Način na koji ovo "zbrajanje" funkcionira je sljedeći. Recimo da imamo dvije varijable: `T* pt` (gdje je `T` neki proizvoljni tip) i `int n`. Rezultat izvrjednjavanja izraza `pt+n` bit će varijabla tipa `T*` čija vrijednost će biti vrijednost od `pt` uvećana za `n*sizeof(T)`.
Ovaj mehanizam omogućava da se sintaksa za adresiranje u poljima – `niz[n]` realizira kao pokrata za `*(niz+n)`.

Pogledajte i primjer na sljedećem slideu.

```
int main(void){
char   *char_ptr   = (char*)   100;
int    *int_ptr    = (int*)    100;
double *double_ptr = (double*) 100;
printf("Inicijalne vrijednosti pointera:\n");
printf("\tchar_ptr   = %d\n", char_ptr);
printf("\tint_ptr    = %d\n", int_ptr);
printf("\tdouble_ptr = %d\n", double_ptr);
char_ptr   = char_ptr   + 1;
int_ptr    = int_ptr    + 1;
double_ptr = double_ptr + 1;
printf("\n\nVrijednosti pointera nakon inkrementa:\n");
printf("\tchar_ptr   = %d\n", char_ptr);
printf("\tint_ptr    = %d\n", int_ptr);
printf("\tdouble_ptr = %d\n", double_ptr);
return 0;
}
```

Ispis programa s prethodnog slidea:

```
Inicijalne vrijednosti pointera:
char_ptr   = 100
int_ptr    = 100
double_ptr = 100


Vrijednosti pointera nakon inkrementa:
char_ptr   = 101
int_ptr    = 104
double_ptr = 108
```

Koristeći činjenicu da nam pointeri mogu glumiti polja, polja varijabilne duljine dobivamo koristeći dinamičko alociranje memorije. Polje od `n` elemenata tipa `T` konstruiramo na sljedeći način:

```
T* polje = malloc(n * sizeof(T));
```

Funkciji `malloc` kažemo koliko točno byteova želimo rezervirati i kao povratnu vrijednost dobijemo pointer na početak bloka memorije tražene veličine ili (u slučaju greške) `NULL`.

Strukture

Strukture u C-u nam daju mogućnost stvaranja “novih tipova” na način da pravimo kolekciju već postojećih tipova. Ako su `T1` `T2`, . . . , `Tn` neki tipovi, onda novi tip tj. strukturu kreiramo na sljedeći način:

```
struct struktura {
  T1   var1;
  T2   var2;
     .
     .
     .
  Tn   varn;
};
```

Novi tip se zove `struct struktura`. Varijable tog tipa deklatitamo kao `struct struktura S;` a pointere na taj tip kao `struct struktura* pS;`

Članovima strukture pristupamo sa `S.var1`, `S.var1`, ..., `S.varn`, ili preko pointera – `pS->var1`, `pS->var2`, ..., `pS->varn`.
Uočimo da `a->x` nije ništa drugo nego pokrata za `(*a).x`