Napredni C kurs

Namenski računarski sistemi
Čas 06, 2021/2022

Poravnavanje strukture - pragma pack(n)

- Prilikom rada sa memorijom, velik broj procesora ne može da pročita proizvoljan deo memorije, već se čitanje memorije radi u paketima po N bajtova
- Adrese paketa od po N bajtova su uvek deljive sa N
- Da bi se brže dobavio podatak veličine do N bajtova, zabranjuje se da se podatak nalazi u dva paketa
- Veličine paketa N su najčešće 1, 4 i 8 bajta
- Primeri:
  - Ako je veličina paketa 1 bajt, tada procesor može da pristupa bilo kojem bajtu u memoriji
  - Ako je veličina paketa 4 bajta, tada procesor učitava uvek po 4 bajta i to sa adresa deljivih sa 4
  - Ako je veličina paketa 8 bajtova, tada procesor učitava uvek po 8 bajtova i to sa adresa deljivih sa 8

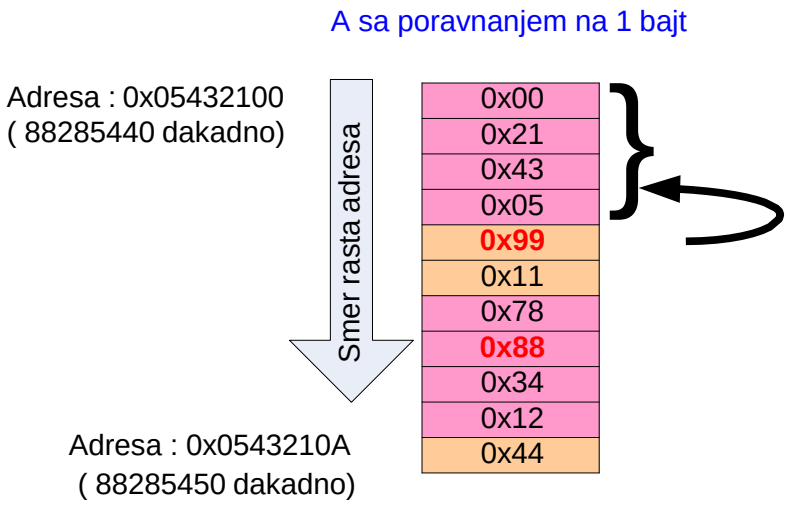
Poravnavanje strukture - pragma pack(n)

Pravila za pakovanje podataka pack(n):

- Memorija se deli na blokove veličine n bajtova
- Svaka struktura zauzima memorijski prostor počevši od memorijske adrese deljive sa n
- Svaki podatak koji zauzima ne više bajtova od n, smešta se u memoriju počevši od adrese deljive sa brojem bajtova koje on zauzima, tj. na adresu koja je njegov umnožak. Na primer, adrese za int su deljive sa 4, za short sa 2, itd...
- Svaki podatak koji zauzima više od n bajtova, deli se u delove od po n bajtova, pri čemu poslednji deo ne mora da bude popunjen do kraja.
- Svaki podatak koji zauzima više od n bajtova, smešta se u memoriji od adrese deljive sa n
- Ako neki podatak ne može da stane unutar trenutnog bloka, on se smešta od početka slededeg bloka
- Količina memorije koju zauzima struktura uvek je deljiva sa n, jer uključuje i prazan prostor na kraju poslednjeg bloka

Poravnavanje strukture - pragma pack(n)

Primer 1: Izvršiti poravnavanje date strukture koristeći pakovanje strukture po 1 bajt (pragma pack(1))

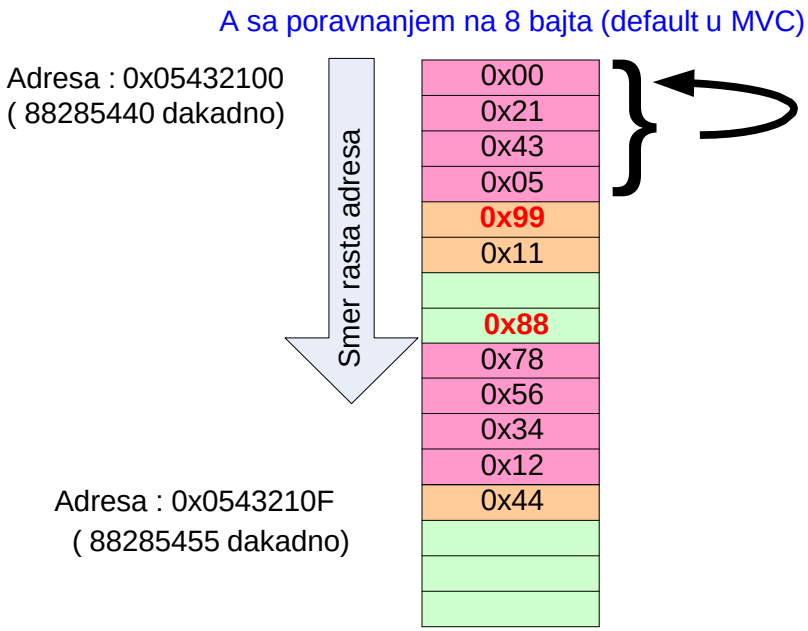



```
typedef struct
{
      unsigned char* p1;
      unsigned short p2;
      unsigned long p3;
      unsigned char  p4;
} struktura_T;

struktura_T A;
A.p1 = (unsigned char*)&A.p1;
A.p2 =  0x1122;
A.p3 = 0x12345678;
A.p4 =  0x44;

A.p1[4] =  0x99;
*(A.p1 + 7) =  0x88;
```

Ukupna veličina strukture: 11 bajtova

Poravnavanje strukture - pragma pack(n)

Primer 2: Izvršiti poravnavanje date strukture koristeći pakovanje strukture po 4 bajta (pragma pack(4))




```
typedef struct
{
      unsigned char* p1;
      unsigned short p2;
      unsigned long p3;
      unsigned char  p4;
} struktura_T;

struktura_T A;
A.p1 = (unsigned char*)&A.p1;
A.p2 =  0x1122;
A.p3 = 0x12345678;
A.p4 =  0x44;

A.p1[4] =  0x99;
*(A.p1 + 7) =  0x88;
```

Ukupna veličina strukture: 16 bajtova

Poravnavanje strukture - pragma pack(n)

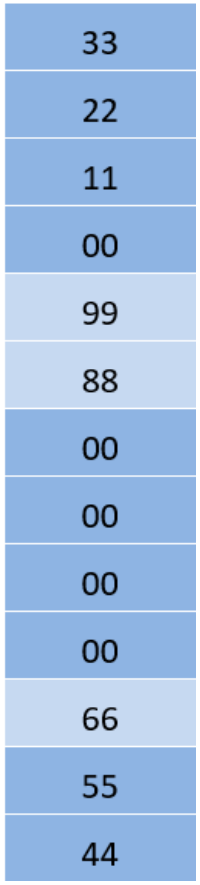
Zadatak 1: Napisati veličinu sledede strukture i upisati raspored polja date strukture u memoriji, u pravougaonike, za pack(1) i Little Endian format.

```
struct Struktura
{
    long l;
    short s;
    char* c;
    char cc;
    short ss;
};
Struktura st;
st.l = 0x112233;
st.s = 0x8899;
st.c = 0x00;
st.cc = 0x7766;
st.ss = 0x4455;
```

Poravnavanje strukture - pragma pack(n)

Zadatak 1: Napisati veličinu sledede strukture i upisati raspored polja date strukture u memoriji, u pravougaonike, za pack(1) i Little Endian format.

```
struct Struktura
{
   long l;
   short s;
   char* c;
   char cc;
   short ss;
};
Struktura st;
st.l = 0x112233;
st.s = 0x8899;
st.c = 0x00;
st.cc = 0x7766;
st.ss = 0x4455;
```



- l zauzima prva 4 bajta i tamo se smešta prvo 33, pa 22, pa 11, pa 00, iako se ovoj bajt 00 ni ne piše u naredbi dodele (uvek se smesti!!!)
- s zautima 2 bajta i tu smeštamo prvo 99, pa zatim 88
- c je pokazivač, pa tu smeštamo 4 bajta. Prvo je 00, a i ostala 3 su 00 iako oni nisu napisani.
- cc zauzima 1 bajt, pa kada smeštamo 0x7766 tamo, smešta se vrednost 66, a vrednost 77 se ne smešta nigde, jer nemamo memorije za to
- ss zauzima isto 2 bajta i tamo smeštamo prvo 55, pa 44

Poravnavanje strukture - pragma pack(n)

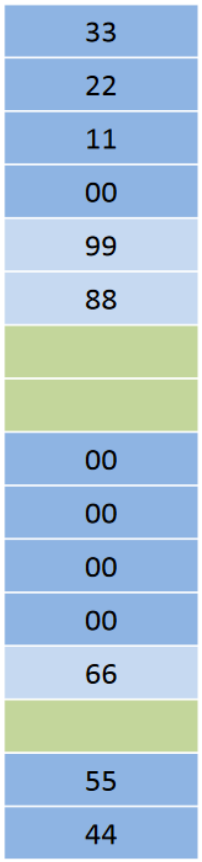
Zadatak 2: Napisati veličinu sledede strukture i upisati raspored polja date strukture u memoriji, u pravougaonike, za pack(4) i Little Endian format.

```
struct Struktura
{
    long l;
    short s;
    char* c;
    char cc;
    short ss;
};
Struktura st;
st.l = 0x112233;
st.s = 0x8899;
st.c = 0x00;
st.cc = 0x7766;
st.ss = 0x4455;
```

Poravnavanje strukture - pragma pack(n)

Zadatak 2: Napisati veličinu sledede strukture i upisati raspored polja date strukture u memoriji, u pravougaonike, za pack(4) i Little Endian format.

```
struct Struktura
{
    long l;
    short s;
    char* c;
    char cc;
    short ss;
};
Struktura st;
st.l = 0x112233;
st.s = 0x8899;
st.c = 0x00;
st.cc = 0x7766;
st.ss = 0x4455;
```



- l zauzima prva 4 bajta i tamo se smešta prvo 33, pa 22, pa 11, pa 00, iako se ovoj bajt 00 ni ne piše u naredbi dodele (uvek se smesti!!!)
- S zautima 2 bajta i tu smeštamo prvo 99, pa zatim 88
- C je pokazivač, pa tu smeštamo 4 bajta. Prvo je 00, a i ostala 3 su 00 iako oni nisu napisani. Kako ne mogu sva 4 da se stave u jedan blok, preskaču se 2 bajta
- cc zauzima 1 bajt, pa kada smeštamo 0x7766 tamo, smešta se vrednost 66, a vrednost 77 se ne smešta nigde, jer nemamo memorije za to
- ss zauzima isto 2 bajta i tamo smeštamo prvo 55, pa 44. Kako ss mora da se smesti od adrese deljive sa 2, preskačemo 1 bajt

Poravnavanje strukture - pragma pack(n)

Zadatak 3: Neka je data struktura podataka Osoba, kao što je prikazano dole. Napisati funkciju ispisiStariju() koja prihvata niz bajtova koji sadrži podatke o dve osobe definisane tipom Osoba, i ispisuje podatke o osobi koja je starija. Funkcija je data sledećim zaglavljem:

```
void ispisStariju(char *buffer);

 typedef struct {
   char ime[12];
   char prezime[12];
   char pol;
   short danR, mesecR, godinaR;
   char JMBG[14];
} Osoba;
```

Poravnavanje strukture - pragma pack(n)

Zadatak 4, za samostalan rad: Neka je data struktura podataka Osoba, kao što je prikazano dole. Napisati funkciju ispisiNajstariju() koja prihvata niz bajtova koji sadrži podatke o N osoba definisane tipom Osoba i ispisuje podatke o najstarijoj osobi. Broj osoba N je tipa podataka short int i zauzima prva dva bajta bafera. Funkcija je data sledećim zaglavljem:

```
void ispisNajstariju(char *buffer);

 typedef struct {
   char ime[12];
   char prezime[12];
   char pol;
   short danR, mesecR, godinaR;
   char JMBG[14];
} Osoba;
```