Software Básico

Assembly: Estrutura

Estrutura

- Coleção de valores com tipos diferentes
- Elementos de uma *struct* geralmente são armazenados em sequência na memória
  - Mas nem sempre de forma contígua

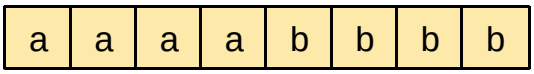
```
struct s {
  int a;
  int b;
};

struct s s1;
```

a	a	a	a	b	b	b	b

Estrutura

- O mapeamento de uma variável global do tipo estrutura para Assembly deve seguir a mesma ordem dos campos e a regra de alinhamento

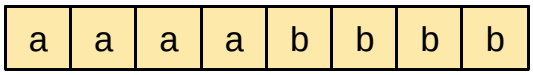
```
struct s {
  int a;
  int b;
};

struct s s1;
```



Estrutura

```
struct s {
  int a;
  int b;
};

struct s s1;

int main() {
  …
}
```

```
.data


s1:
   .int 0  /* campo ‘a’ */
   .int 0  /* campo ‘b’ */

.text
.globl main
main:
   …
```



Estrutura

```
struct s {
  int a;
  int b;
};

struct s s1;

int main() {
  …
}
```

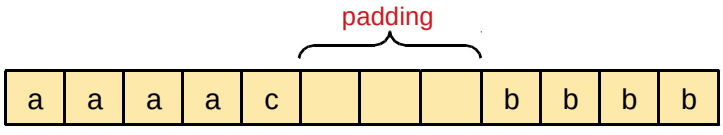
```
.data

.align 8
s1:
   .int 0  /* campo ‘a’ */
   .int 0  /* campo ‘b’ */

.text
.globl main
main:
   …
```

Para garantir que o início da estrutura esteja alinhado, podemos usar a diretiva “**.align X**”, que diz para o compilador que o endereço inicial deve ser múltiplo de X.

Estrutura

- O *padding* entre os campos ou no final deve ser inserido manualmente
- Podemos usar “*.zero*” para inserir uma quantidade de bytes na posição desejada

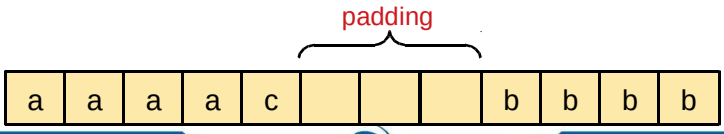
```
struct s {
  int  a;
  char c;
  int  b;
};

struct s s2;
```



Estrutura

```
struct s {
  int  a;
  char c;
  int  b;
};

struct s s2;

int main() {
  …
}
```

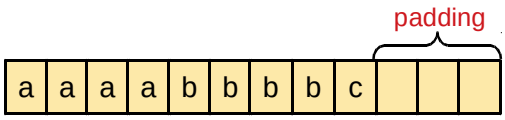
```
.data

.align 8
s2:
   .int  0  /* campo 'a' */
   .byte 0  /* campo 'c' */
   .zero 3  /* padding   */
   .int  0  /* campo 'b' */

.text
   …
```



Alinhamento de Memória

- Outro exemplo...

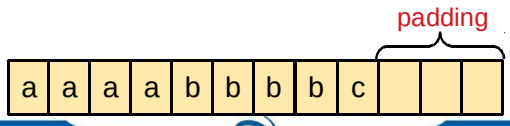
```
struct s {
  int  a;
  int  b;
  char c;
};

struct s s3;
```



Estrutura

```
struct s {
  int  a;
  int  b;
  char c;
};

struct s s3;

int main() {
  …
}
```

```
.data

.align 8
s3:
   .int  0  /* campo ‘a’ */
   .int  0  /* campo ‘b’ */
   .byte 0  /* campo ‘c’ */
   .zero 3  /* padding   */

.text
   …
```



Acessando os Campos da Estrutura

Acessando os Campos

- O acesso aos campos é dado através do deslocamento a partir do início da estrutura

```
struct s {
  int a;
  int b;
};

struct s s1;

int main() {
   s1.a = 100;
   s1.b = 200;
}
```

```
.data

.align 8
s1: .int 0   # a: s1+0
    .int 0   # b: s1+4

.text
main:
   …
   movq $s1, %r12
   movl $100,  (%r12)
   movl $200, 4(%r12)
```


Acessando os Campos

```
struct s {
  char a;
  int  b;
  char c;
};

struct s s1;

int main() {
   s1.b = 100;
   s1.c = s1.a;
}
```

```
.data

.align 8
s1: .byte 0    # a: +0
    .zero 3    # padding
    .int  0    # b: +4
    .byte 0    # c: +8
    .zero 3    # padding

.text
main:
   …
   movq $s1, %r12
   movl $100,4(%r12)
   movb (%r12), %al
   movb al,8(%r12);
```

Acessando os Campos

```
struct s {
  int a;
  int b;
};

struct w {
  int x;
  int y;
};

struct s s1;
struct w w1;

int main() {
   w1.y = s1.a;
}
```

```
.data

.align 8
s1: .int  0    # a: +0
    .int  0    # b: +4

.align 8
w1: .int 0     # x: +0
    .int 0     # y: +4

.text
main:
   …
   movq $s1, %r12
   movq $w1, %r13
   movl (%r12), %eax
   movl %eax,  4(%r13)
```