

Estruturas

- Variáveis possuem tipos diferentes, de acordo com a necessidade:
 - Nomes são guardados em vetores de *char*
 - Idades, em variáveis *unsigned int*
 - Saldos bancários, em variáveis *float*

Estruturas

- Esta quantidade de dados pode começar a "tumultuar" o código.
- Se considerarmos que essas variáveis são atributos de alguma 'entidade' maior, gostaríamos de agrupá-las para descrever essa 'entidade'.

Estruturas

- Um cliente de um banco, por exemplo possui estes atributos, entre outros:
 - Nome, guardado em um vetor de *char*
 - Idade, em uma variável *unsigned int*
 - Saldo bancário, em uma variável *float*
- Solução: ESTRUTURAS

Estruturas

```
struct nome_struct  
{  
    tipo_membro1 nome_membro1;  
    tipo_membro2 nome_membro2;  
    ...  
    tipo_membroN nome_membroN;  
};
```

Memória dinâmica

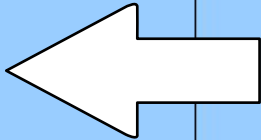
```
#include <string.h>
struct dados_pessoais {
    int dia, mes, ano;
    char nome[200];
};

void main()
{
    struct dados_pessoais p1;
    p1.dia = 1;
    p1.mes = 1;
    p1.ano = 1980;
    strcpy(p1.nome, "Fulano de tal");
    printf("%s - %2d/%2d/%2d\n",
        p1.nome, p1.dia, p1.mes, p1.ano);
}
```

Memória dinâmica

```
#include <string.h>
struct dados_pessoais {
    int dia, mes, ano;
    char nome[200];
};

void main()
{
    struct dados_pessoais p1;
    p1.dia = 1;
    p1.mes = 1;
    p1.ano = 1980;
    strcpy(p1.nome, "Fulano de tal");
    printf("%s - %2d/%2d/%2d\n",
        p1.nome, p1.dia, p1.mes, p1.ano);
}
```

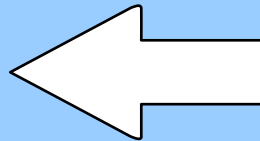


Define-se uma estrutura de nome *dados_pessoais*, que guarda o nome da pessoa e valores de dia, mes e ano

Memória dinâmica

```
#include <string.h>
struct dados_pessoais {
    int dia, mes, ano;
    char nome[200];
};

void main()
{
    struct dados_pessoais p1;
    p1.dia = 1;
    p1.mes = 1;
    p1.ano = 1980;
    strcpy(p1.nome, "Fulano de tal");
    printf("%s - %2d/%2d/%2d\n",
        p1.nome, p1.dia, p1.mes, p1.ano);
}
```




Declaramos uma
estrutura do tipo
dados_pessoais, de
nome *p1*

Memória dinâmica

```
#include <string.h>
struct dados_pessoais {
    int dia, mes, ano;
    char nome[200];
};

void main()
{
    struct dados_pessoais p1;
    p1.dia = 1;
    p1.mes = 1;
    p1.ano = 1980;
    strcpy(p1.nome, "Fulano de tal");
    printf("%s - %2d/%2d/%2d\n",
        p1.nome, p1.dia, p1.mes, p1.ano);
}
```

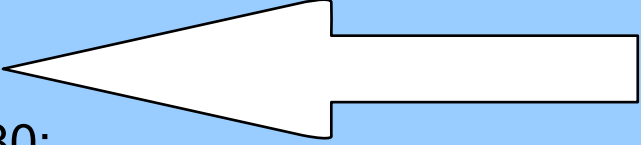


Definimos que o *dia*
da pessoa *p1* é 1

Memória dinâmica

```
#include <string.h>
struct dados_pessoais {
    int dia, mes, ano;
    char nome[200];
};

void main()
{
    struct dados_pessoais p1;
    p1.dia = 1;
    p1.mes = 1;
    p1.ano = 1980;
    strcpy(p1.nome, "Fulano de tal");
    printf("%s - %2d/%2d/%2d\n",
        p1.nome, p1.dia, p1.mes, p1.ano);
}
```

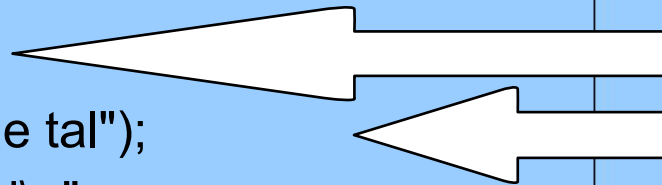


Definimos que o *mes*
da pessoa *p1* é 1

Memória dinâmica

```
#include <string.h>
struct dados_pessoais {
    int dia, mes, ano;
    char nome[200];
};

void main()
{
    struct dados_pessoais p1;
    p1.dia = 1;
    p1.mes = 1;
    p1.ano = 1980;
    strcpy(p1.nome, "Fulano de tal");
    printf("%s - %2d/%2d/%2d\n",
        p1.nome, p1.dia, p1.mes, p1.ano);
}
```

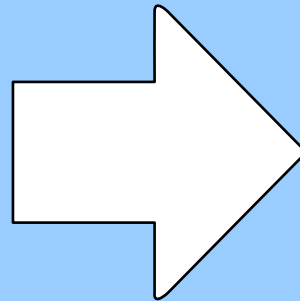


Etc. etc.

Memória dinâmica

```
#include <string.h>
struct dados_pessoais {
    int dia, mes, ano;
    char nome[200];
};

void main()
{
    struct dados_pessoais p1;
    p1.dia = 1;
    p1.mes = 1;
    p1.ano = 1980;
    strcpy(p1.nome, "Fulano de tal");
    printf("%s - %2d/%2d/%2d\n",
        p1.nome, p1.dia, p1.mes, p1.ano);
}
```



Fulano de tal - 1/ 1/1980

Estruturas

- Também são válidos ponteiros para estruturas.
Se tivermos, por exemplo:

```
struct nome_struct  
{  
    int a, b, c;  
};
```

- E em algum ponto do código, tivermos a declaração:

```
struct nome_struct st, *p_st;  
p_st = &st;
```

Estruturas

- As seguintes expressões são equivalentes:

- $st.a$, $p_st \rightarrow a$ e $(*p_st).a$
- $st.b$, $p_st \rightarrow b$ e $(*p_st).b$
- $st.c$, $p_st \rightarrow c$ e $(*p_st).c$

- $*p_st.a$ e $*(p_st.a)$
- $*p_st.b$ e $*(p_st.b)$
- $*p_st.c$ e $*(p_st.c)$