Ponteiros e registros

Algoritmos e Programação II

Declaração

```
struct data{
int dia;
int mes;
int ano;
};

struct data hoje;
struct data *p;
p = &hoje;
```

C++

```
data hoje;
data *p;
```

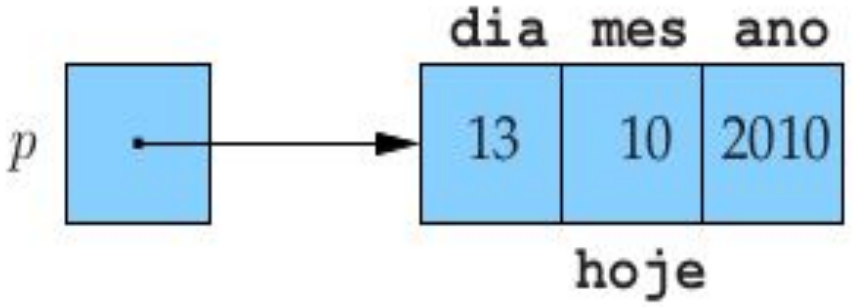


Figura 15.1: Representação do ponteiro p e do registro `hoje`.

Atribuições

```
struct data hoje;
struct data *p;
p = &hoje;
```

```
(*p).dia = 11;
```

Ou

```
p->dia = 11;
```

```c
#include <stdio.h>

struct data {
   int dia;
   int mes;
   int ano;
};

int main(void)
{
   struct data hoje, *p;

   p = &hoje;
   p->dia = 13;
   p->mes = 10;
   p->ano = 2010;
   printf("A data de hoje é %d/%d/%d\n", hoje.dia, hoje.mes, hoje.ano);

   return 0;
}
```

// C++: data hoje, *p;

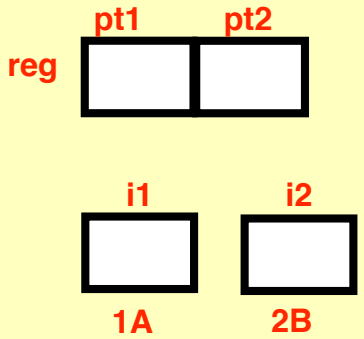
Registros contendo ponteiros

```
struct reg_pts{
    int* pt1;
    int* pt2;
};
```

Vamos declarar uma variável do tipo registro acima:

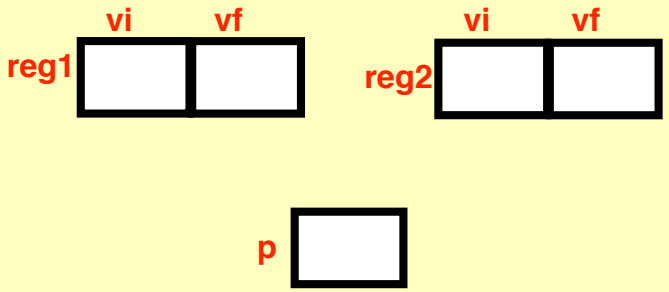
reg_pts bloco;

```c
#include <stdio.h>

struct pts_int {
    int *pt1;
    int *pt2;
};

int main(void)
{
    int i1, i2;
    struct pts_int reg;

    i2 = 100;
    reg.pt1 = &i1;
    reg.pt2 = &i2;
    *reg.pt1 = -2;
    printf("i1 = %d, *reg.pt1 = %d\n", i1, *reg.pt1);
    printf("i2 = %d, *reg.pt2 = %d\n", i2, *reg.pt2);

    return 0;
}
```



// C++: pts_int reg;

QUAL A SAÍDA DO PROGRAMA ABAIXO?




```
#include <stdio.h>

struct dois_valores {
    int vi;
    float vf;
};

int main(void)
{   // C++:      dois_valores  reg1 = {}, reg2, *p = &reg1;
    struct dois_valores reg1 = {53, 7.112}, reg2, *p = &reg1;

    reg2.vi = (*p).vf;
    reg2.vf = (*p).vi;
    printf("1: %d %f\n2: %d %f\n", reg1.vi, reg1.vf, reg2.vi, reg2.vf);

    return 0;
}
```

SIMULE A EXECUÇÃO DO PROGRAMA ABAIXO.

```
#include <stdio.h>

struct celula {
    int valor;
    struct celula *prox;
};

int main(void)
{
    struct celula reg1, reg2, *p;

    scanf("%d%d", &reg1.valor, &reg2.valor);
    reg1.prox = &reg2;
    reg2.prox = NULL;
    for (p = &reg1; p != NULL; p = p->prox)
        printf("%d ", p->valor);
    printf("\n");

    return 0;
}
```