Árvores de busca binária
(IED-001)

Prof. Dr. Silvio do Lago Pereira

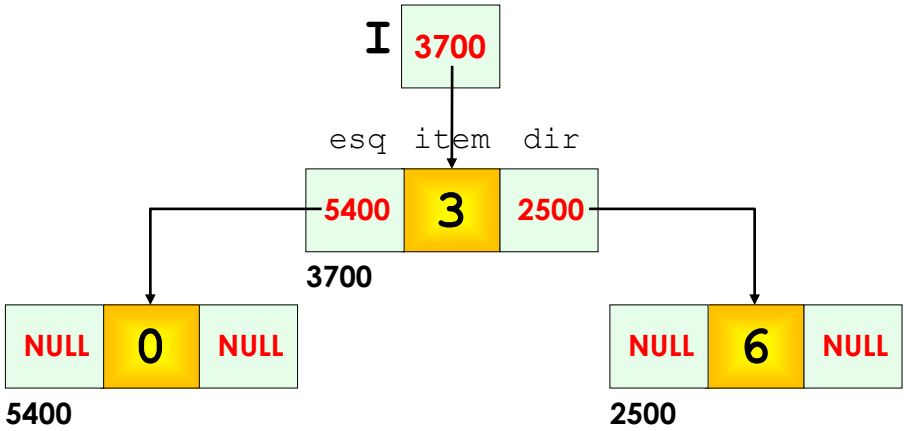
Departamento de Tecnologia da Informação

Faculdade de Tecnologia de São Paulo

Árvores de busca binária

Exemplo 1. O tipo `Arv`

```
typedef int Item;
typedef struct arv {
    struct arv *esq;
    Item item;
    struct arv *dir;
} *Arv;
```



`Arv` ≡ `struct arv *`

Observações:

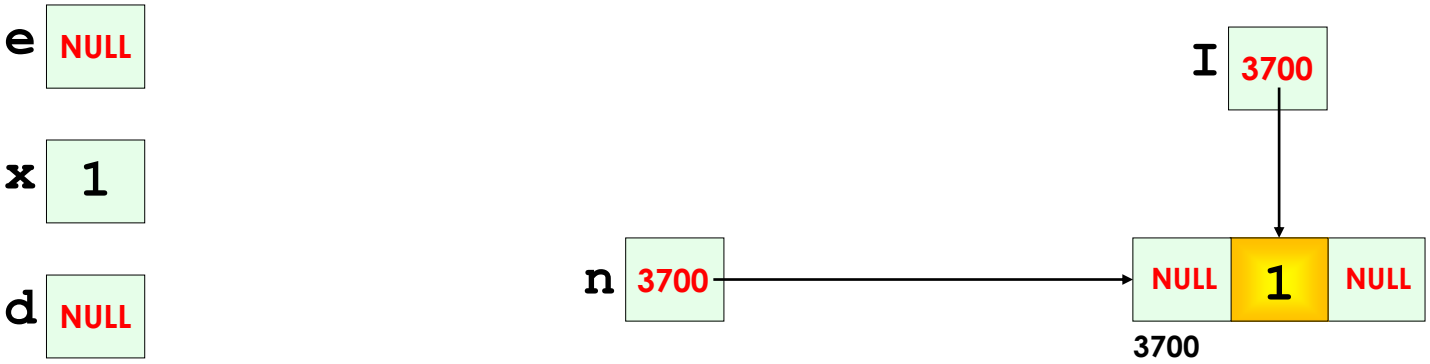
- O tipo **`Item`** indica o tipo dos itens armazenados na árvore binária.
- O tipo **`Arv`** é usado para declarar um ponteiro de árvore binária (que aponta a raiz da árvore).
- Se **`I`** é um ponteiro de árvore binária nulo, então a árvore binária está **vazia**.
- Se **`I`** é um ponteiro de árvore binária não-nulo, então **`I->item`** é a **raiz** da árvore, **`I->esq`** é sua subárvore **esquerda** e **`I->dir`** é sua subárvore **direita**.

Árvores binárias

Exemplo 2. Criação de nó (recapitulação)

```
Arv arv(Arv e, Item x, Arv d) {
    Arv n = malloc(sizeof(struct arv));
    n->esq  = e;
    n->item = x;
    n->dir  = d;
    return n;
  }
```

```
Arv I = arv(NULL,1,NULL);
```



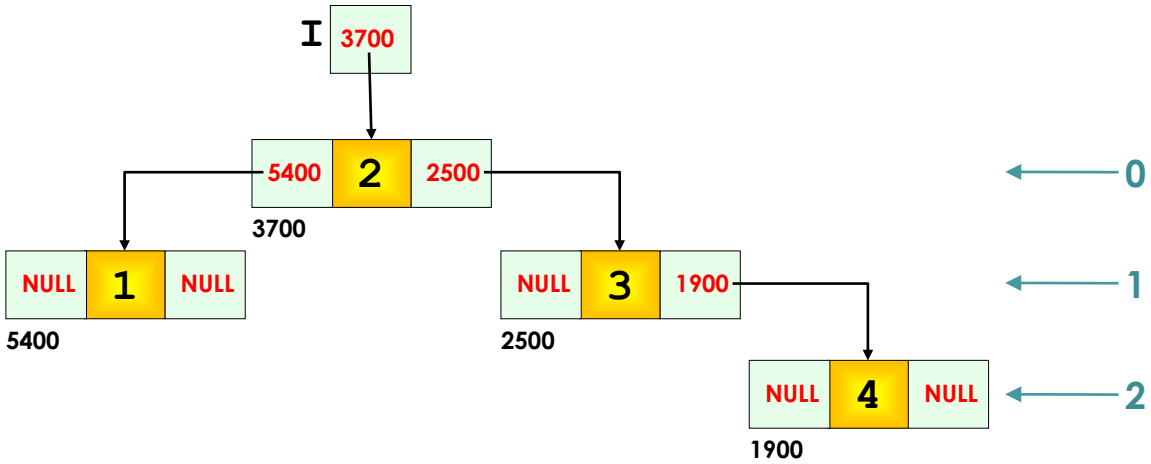
Usando a função **`arv()`** é possível criar qualquer árvore binária desejada!

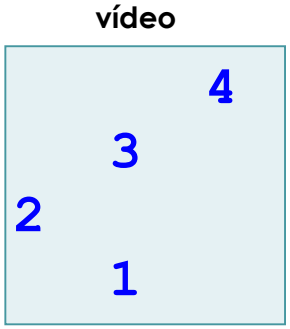
Árvores binárias

Exemplo 3. Exibição de árvore binária (recapitulação)

```
void exibe(Arv A,int n) {
    if( A==NULL ) return;
    exibe(A->dir,n+1);
    printf("%*s%d\n",3*n,"",A->item);
    exibe(A->esq,n+1);
}
```

```
exibe(I,0);
```





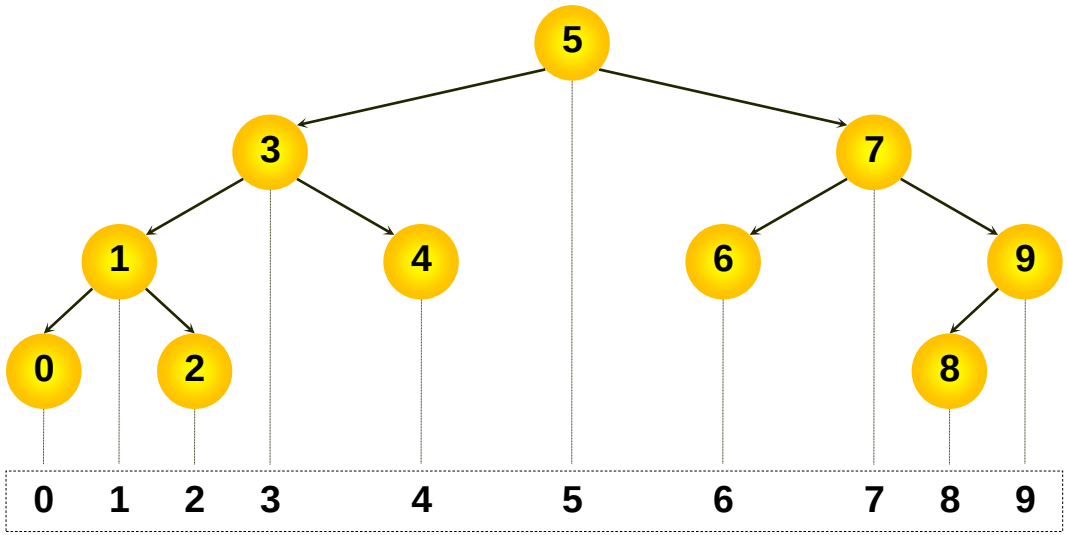
Note que essa função exibe a árvore de modo que possamos ver a hierarquia entre os itens!

Árvores de busca binária

Árvore de busca binária

é uma árvore binária vazia ou uma árvore binária em que:

- todo item na subárvore esquerda é **menor ou igual** à raiz;
- todo item na subárvore direita é **maior** que a raiz;
- cada subárvore é também uma **árvore de busca binária**.

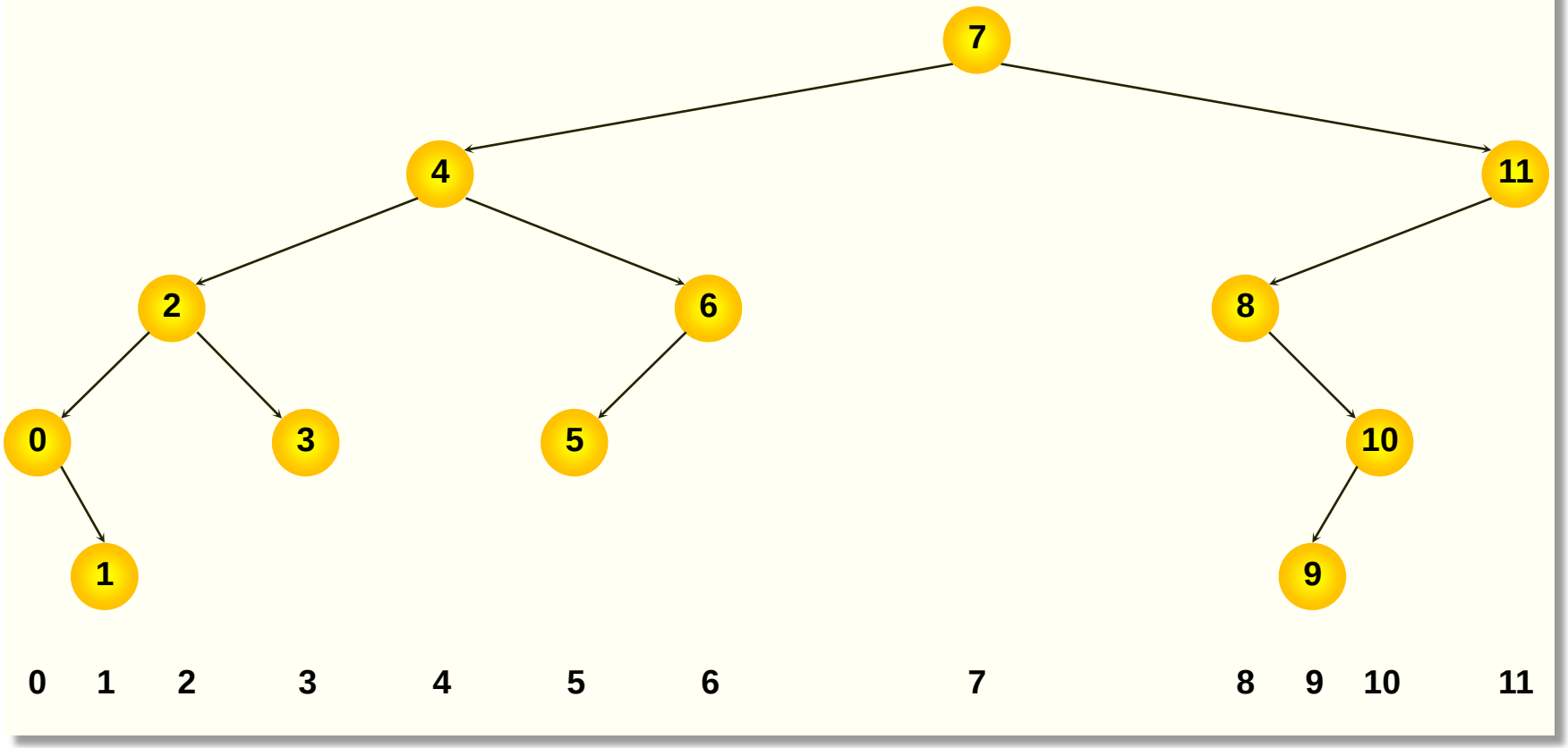


A projeção de uma árvore de busca binária produz uma sequência ordenada crescente!

Árvores de busca binária

Exercício 1. Formação de árvore de busca binária

Qual a árvore de busca binária formada pela inserção dos itens **7**, **4**, **6**, **11**, **8**, **2**, **5**, **0**, **10**, **3**, **1** e **9** nessa ordem?

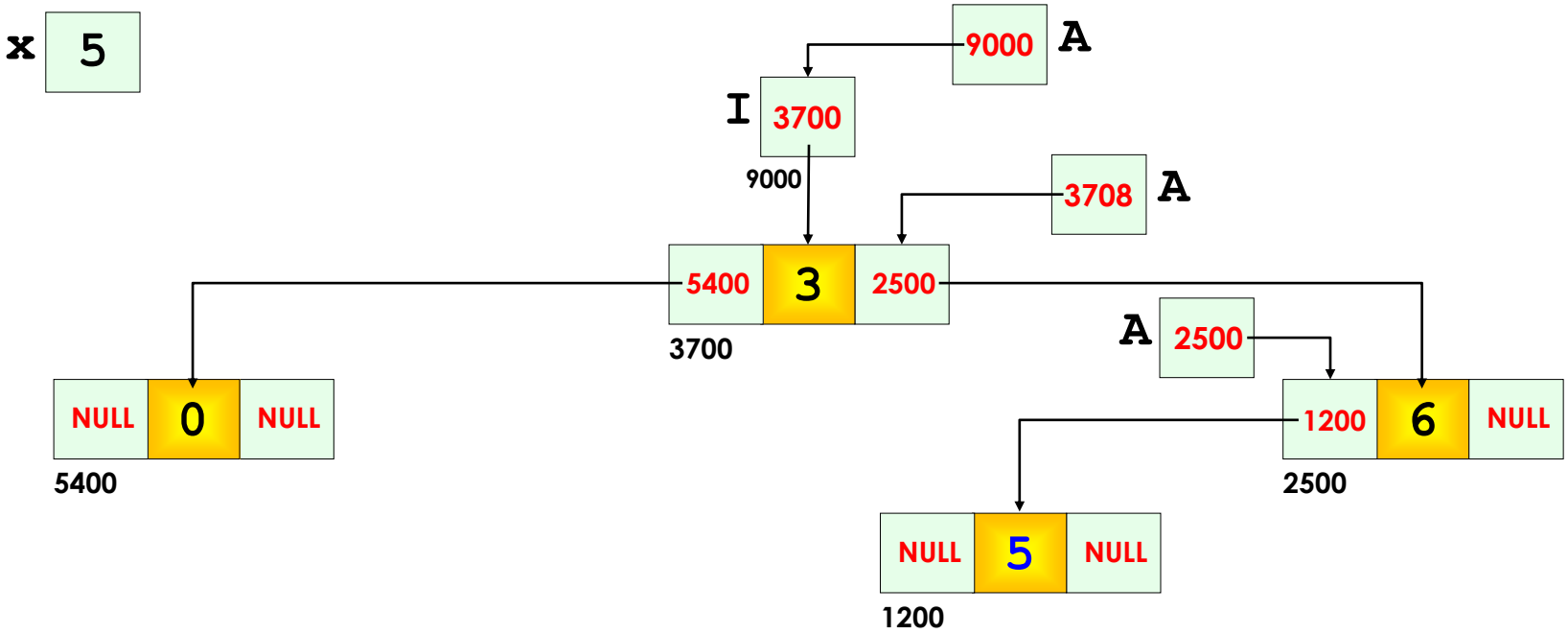


Árvores de busca binária

Exemplo 4. Inserção em árvore de busca binária

```
void ins(Item x, Arv *A) {
    if( *A == NULL ) *A = arv(NULL,x,NULL);
    else if( x <= (*A)->item ) ins(x,&(*A)->esq);
    else ins(x,&(*A)->dir);
}
```

```
ins(5,&I);
```



Árvores de busca binária

Exercício 2. Inserção de itens específicos

Complete e execute o programa a seguir.

```
#include <stdio.h>
#include <stdlib.h>
...
int main(void) {
    Arv I = NULL;
    ins(7,&I);
    ins(4,&I);
    ins(6,&I);
    ins(8,&I);
    ins(2,&I);
    ins(5,&I);
    ins(3,&I);
    ins(1,&I);
    exibe(I,0);
    return 0;
}
```

Árvores de busca binária

Exercício 3. Inserção de itens digitados pelo usuário

Complete e execute o programa a seguir.

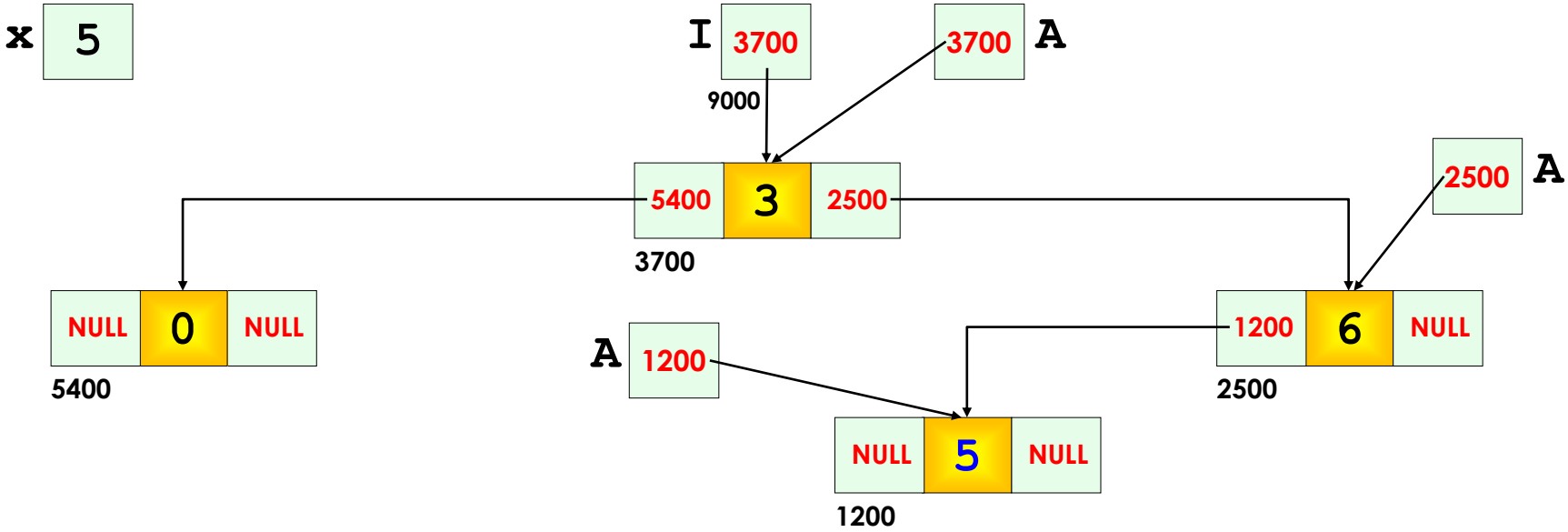
```
#include <stdio.h>
#include <stdlib.h>
...
int main(void) {
    Arv I = NULL;
    Item x;
    puts("\nPara sair, digite um inteiro negativo!\n");
    while( 1 ) {
        printf("Item a ser inserido? ");
        scanf("%d",&x);
        if( x<0 ) break;
        ins(x,&I);
    }
    exibe(I,0);
    return 0;
}
```

Árvores de busca binária

Exemplo 5. Busca em árvore de busca binária

```
int busca(Item x, Arv A) {
    if( A == NULL ) return 0;
    if( x == A->item ) return 1;
    if( x < A->item ) return busca(x,A->esq);
    else return busca(x,A->dir);
}
```

busca(5,I)



Árvores de busca binária

Exercício 4. Busca em árvore de busca binária

Complete e execute o programa a seguir.

```
#include <stdio.h>
#include <stdlib.h>
...
int main(void) {
   int v[9] = {71,43,64,92,80,27,58,35,16};
   Arv A = NULL;
   for(int i=0; i<9; i++) ins(v[i],&A);
   emordem(A);
   puts("\nPara sair, digite um inteiro negativo!");
   while( 1 ) {
      int x;
      printf("\nItem a ser buscado? ");
      scanf("%d",&x);
      if( x<0 ) break;
      if( busca(x,A) ) puts("Encontrado!");
      else puts("Inexistente!");
   }
   return 0;
}
```

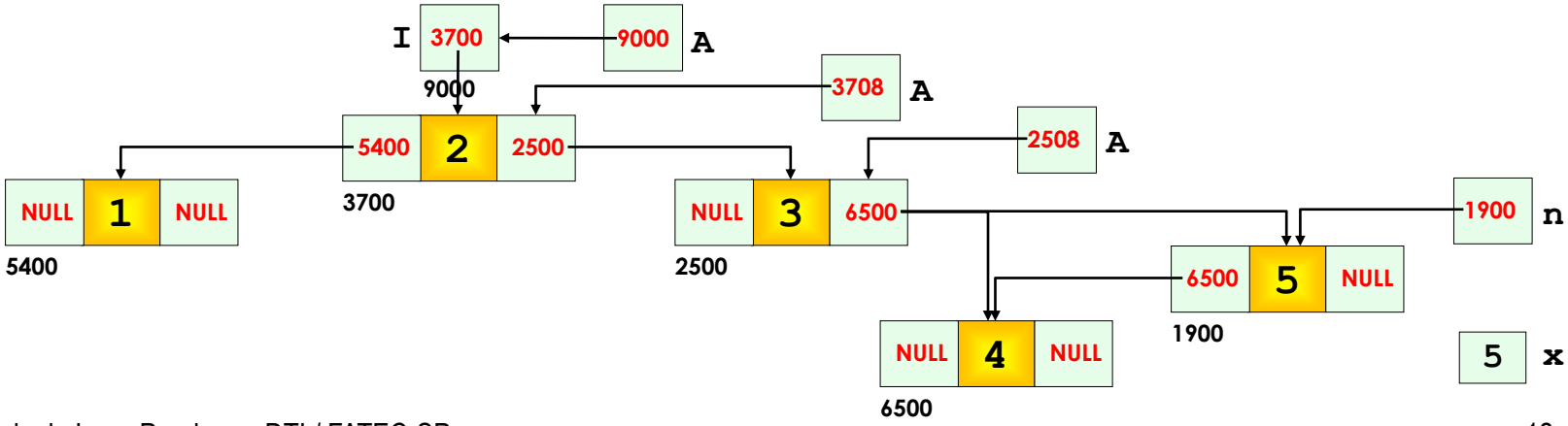
Árvores de busca binária

Exemplo 6. Remoção do item máximo

```
Item remmax(Arv *A) {
    if( *A == NULL ) abort();
    if( (*A)->dir == NULL ) {
        Arv n = *A;
        Item x = n->item;
        *A = n->esq;
        free(n);
        return x;
    }
    return remmax(&(*A)->dir);
}
```

O item máximo está na nó mais à direita possível!

```
remmax(&I);
```



Árvores de busca binária

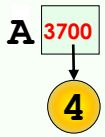
Exercício 5. Remoção de itens máximos

Complete e execute o programa a seguir.

```
#include <stdio.h>
#include <stdlib.h>
...
int main(void) {
    int v[9] = {71,43,64,92,80,27,58,35,16};
    Arv A = NULL;
    for(int i=0; i<9; i++) ins(v[i],&A);
    puts("Inicial");
    exibe(A,0);
    for(int i=0; i<9; i++) {
        puts("Depois de remover o maximo");
        remmax(&A);
        exibe(A,0);
        getchar();
    }
    return 0;
}
```

Árvores de busca binária

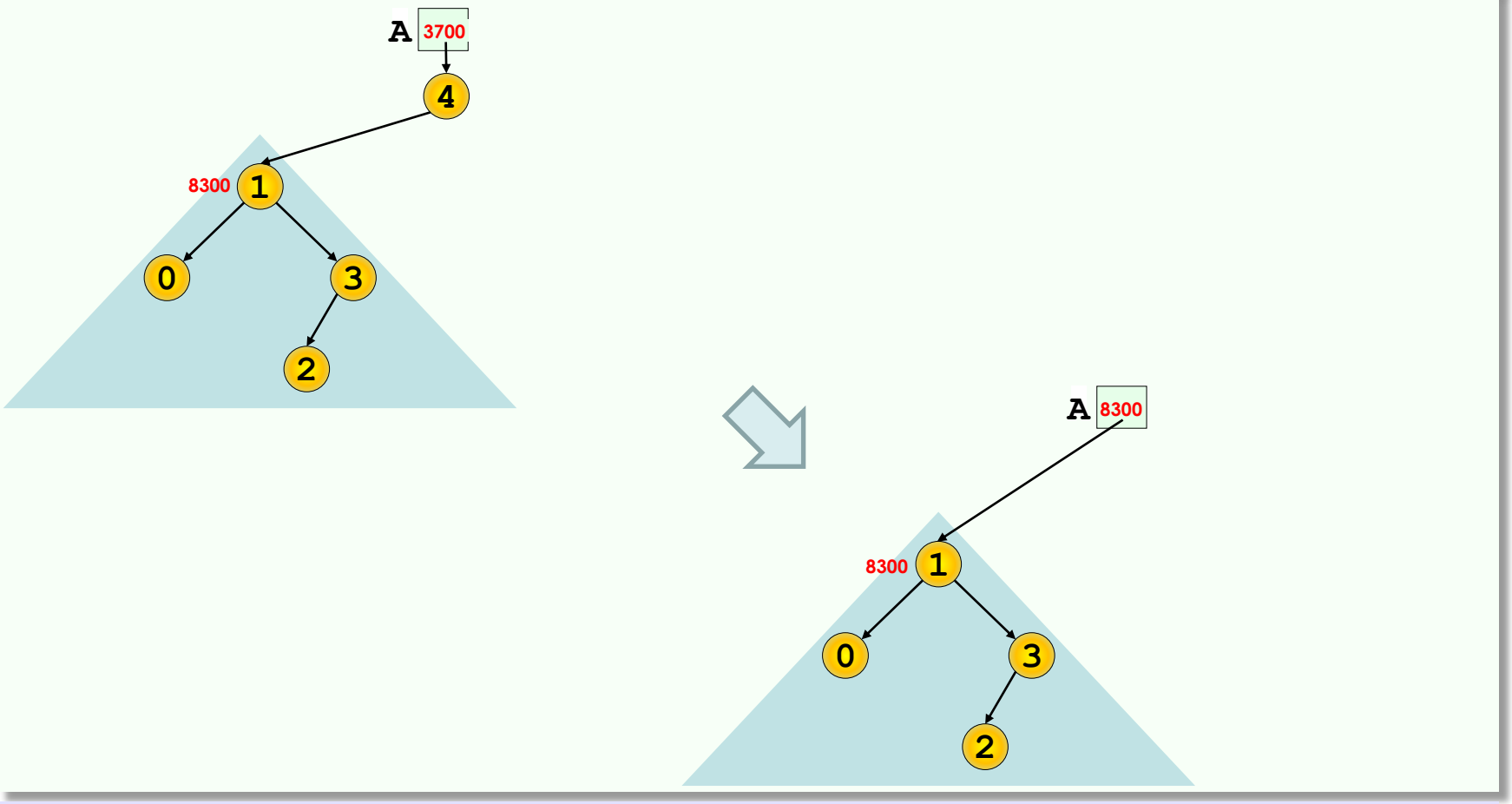
Exemplo 7. Remoção da raiz de uma árvore de busca binária: primeiro caso



O ponteiro que aponta a raiz fica nulo!

Árvores de busca binária

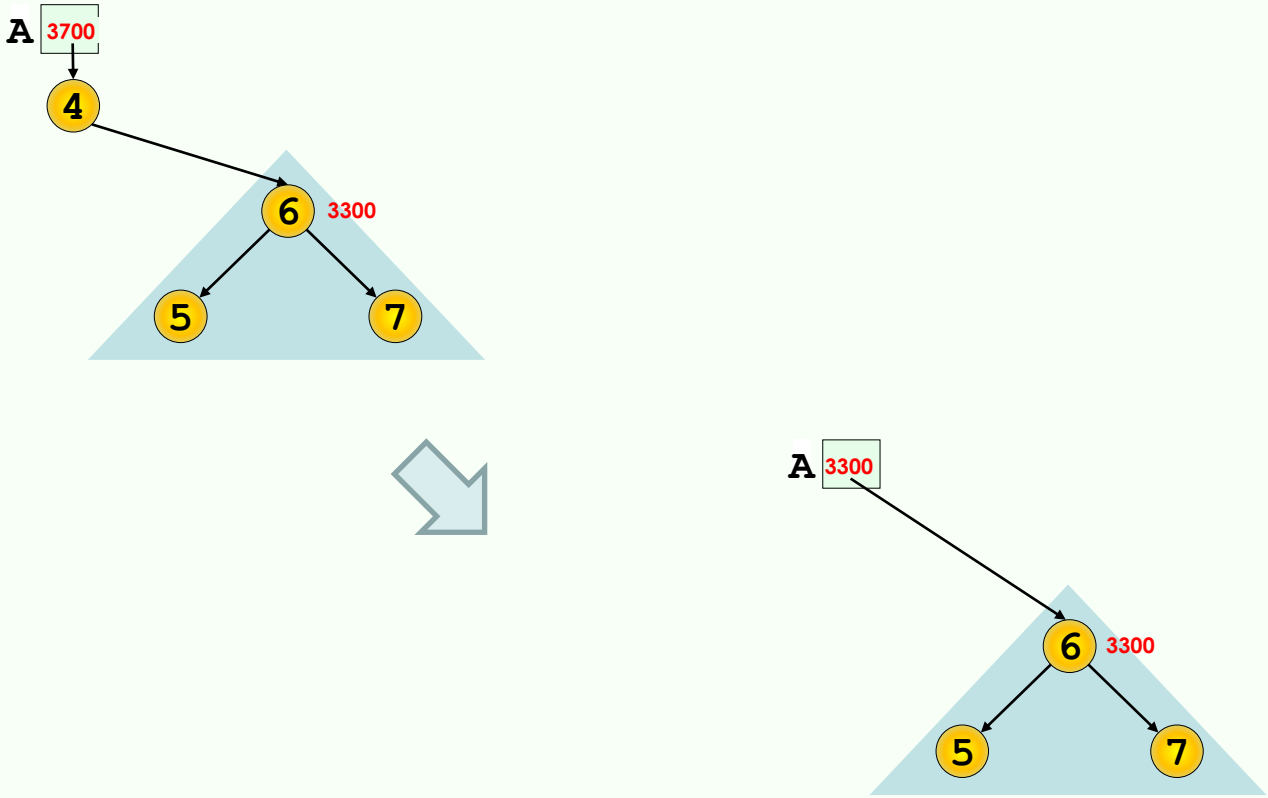
Exemplo 7. Remoção da raiz de uma árvore de busca binária: segundo caso



O ponteiro que aponta a raiz passa a apontar seu filho esquerdo!

Árvores de busca binária

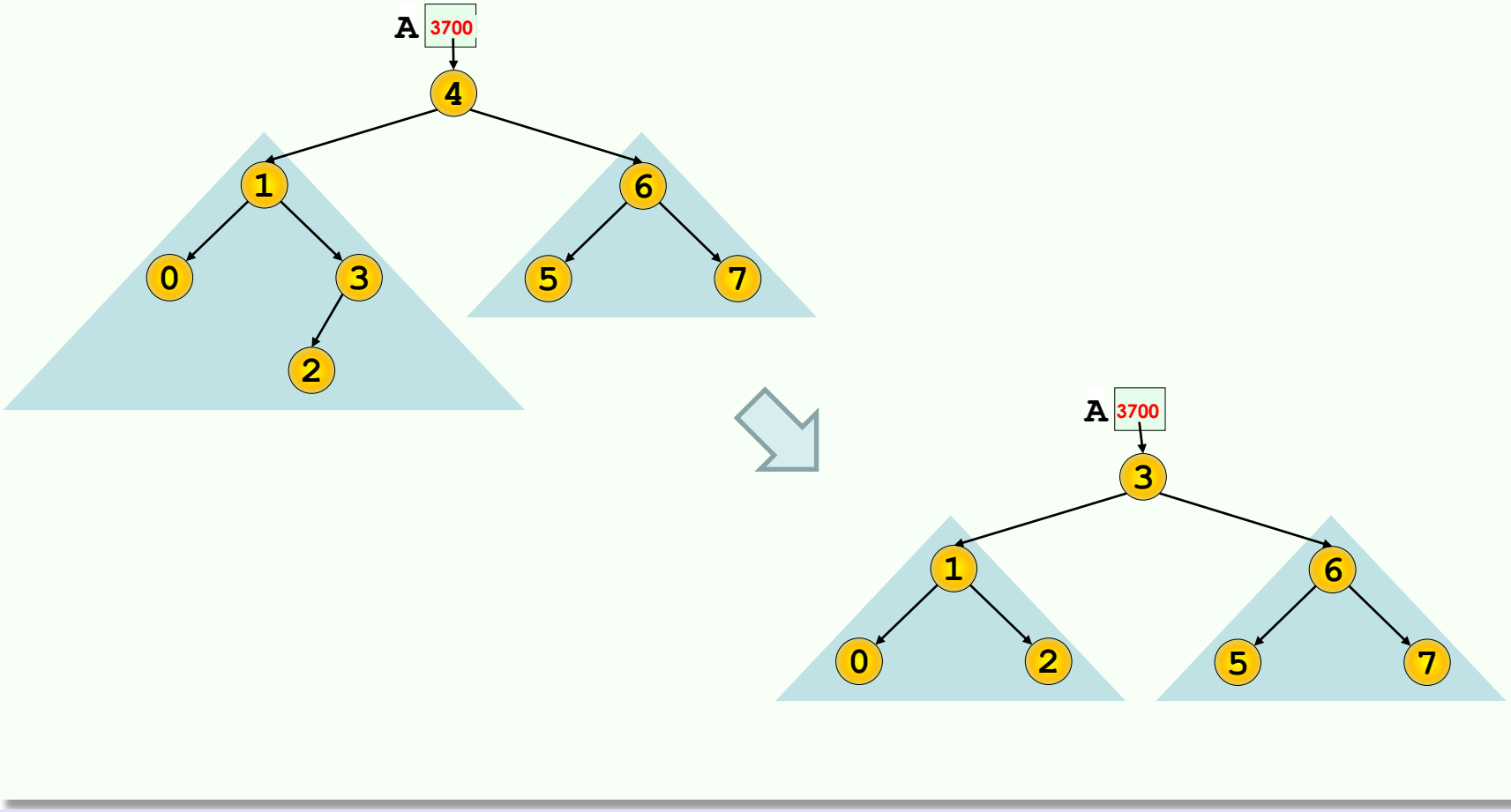
Exemplo 7. Remoção da raiz de uma árvore de busca binária: terceiro caso



O ponteiro que aponta a raiz passa a apontar seu filho direito!

Árvores de busca binária

Exemplo 7. Remoção da raiz de uma árvore de busca binária: quarto caso



O item máximo da esquerda é removido e usado para substituir o valor da raiz!

Árvores de busca binária

Exemplo 7. Remoção da raiz de uma árvore de busca binária

```
void remraiz(Arv *A) {
    if( *A == NULL ) abort();
    Arv n = *A;
    if( n->esq == NULL ) *A = n->dir;
    else if( n->dir == NULL ) *A = n->esq;
    else n->item = remmax(&n->esq);
    if( n != *A ) free(n);
}
```

O nó apontado por `n` não pode ser desalocado se `*A` ainda o aponta!

Exercício 6. Teste da remoção de raiz

Para cada caso considerado no Exemplo 7 (slides 14 a 17), faça um programa que: cria a árvore de busca binária, exibe-a, remove sua raiz e exibe-a novamente.

Árvores de busca binária

Exemplo 8. Remoção em árvore de busca binária

```
void rem(Item x, Arv *A) {
    if( *A == NULL ) return;
    if( x == (*A)->item ) remraiz(A);
    else if( x < (*A)->item ) rem(x,&(*A)->esq);
    else rem(x,&(*A)->dir);
}
```

Se o item a ser removido não está na raiz, a função faz uma busca!

Se o item a ser removido estiver repetido, apenas a primeira ocorrência dele será removida!

Árvores de busca binária

Exercício 7. Remoção de itens arbitrários da árvore

Complete e execute o programa a seguir.

```
#include <stdio.h>
#include <stdlib.h>
...
int main(void) {
   int v[9] = {71,43,64,92,80,27,58,35,16};
   Arv A = NULL;
   for(int i=0; i<9; i++) ins(v[i],&A);
   puts("Inicial");
   exibe(A,0);
   for(int i=0; i<9; i++) {
      printf("Depois de remover o item %d\n",v[i]);
      rem(v[i],&A);
      exibe(A,0);
      getchar();
   }
   return 0;
}
```

Árvores de busca binária

Exercício 8. Inserção sem repetição

Crie a função recursiva **`ins_sr(x,A)`**, que insere o item **`x`** na árvore de busca binária **`A`** se, e somente se, esse item não estiver em **`A`**.

Exercício 9. Remoção de todas as ocorrências

Crie a função recursiva **`rem_todos(x,A)`**, que remove todas as ocorrências do item **`x`** na árvore de busca binária **`A`**.

Exercício 10. Exibição decrescente

Crie a função recursiva **`exibe_dec(A)`**, que exibe todos os itens da árvore de busca binária **`A`**, em ordem decrescente.

Exercício 11. Item máximo

Crie a função recursiva **`maximo(A)`**, que devolve o maior item da árvore de busca binária **`A`**.

Exercício 12. Item mínimo

Crie a função recursiva **`minimo(A)`**, que devolve o menor item da árvore de busca binária **`A`**.

Fim