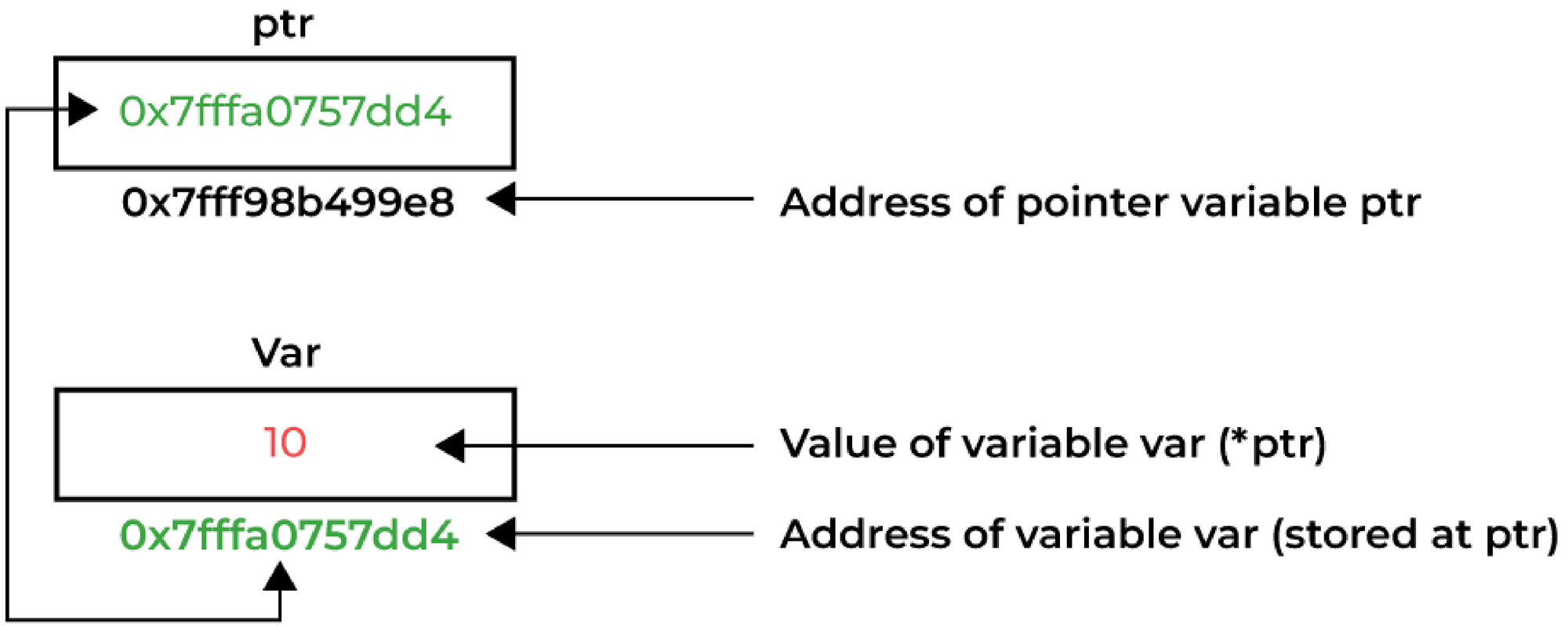
Session 8: Pointers in C

TeachToTech

What is a Pointer?

A pointer is a variable that stores the address of another variable.
Instead of holding data, it holds where the data lives.

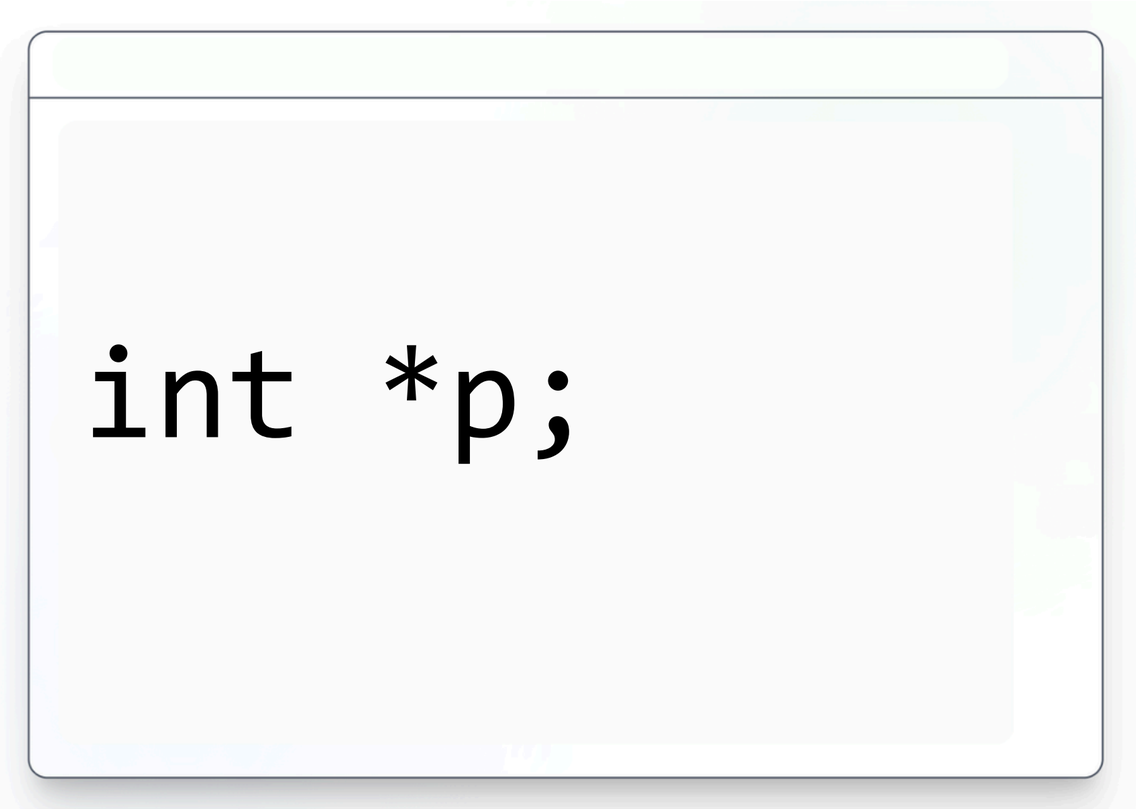

ptr
0x7fffa0757dd4
0x7fff98b499e8
Address of pointer variable ptr
Var
10
Value of variable var (*ptr)
0x7fffa0757dd4
Address of variable var (stored at ptr)

Why Pointers are Important

- Enable efficient memory usage
- Allow pass-by-reference
- Essential for arrays, strings, structures
- Foundation of dynamic memory allocation
- Used in linked lists, trees, and graphs

Pointer Declaration

- int → type of data it points to
- * → pointer symbol
- p → pointer variable

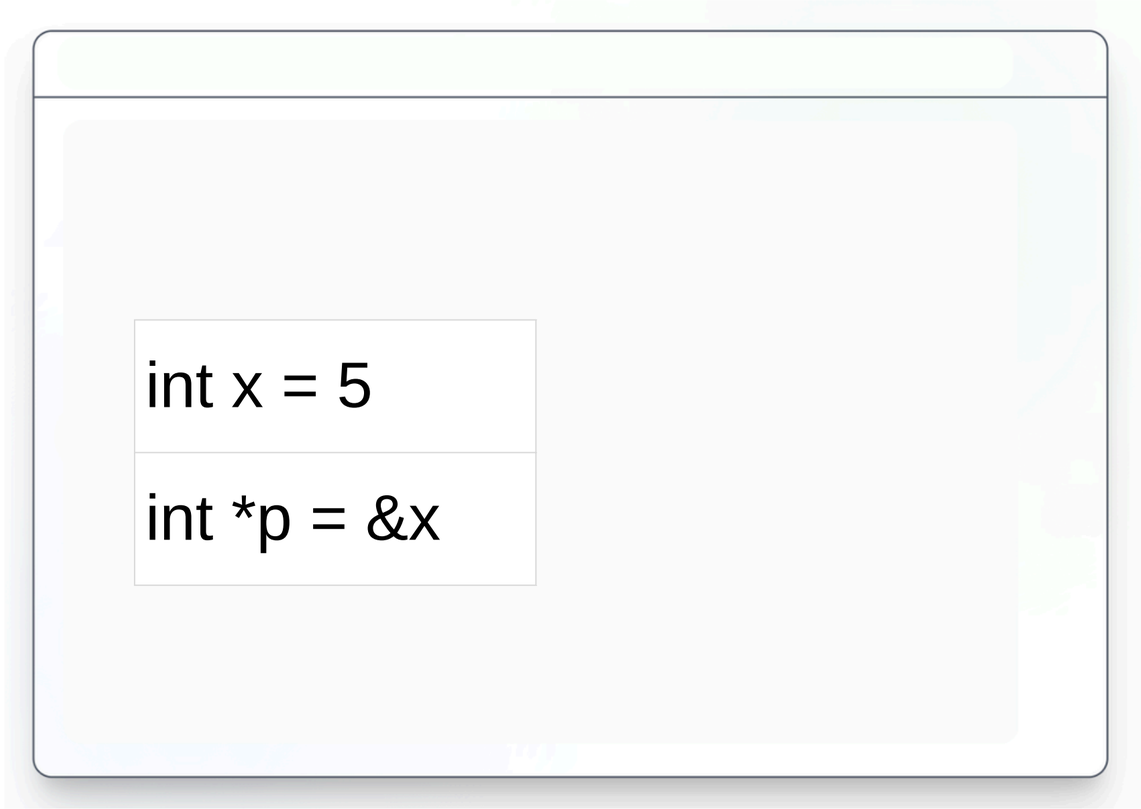


Address-of (&) and Dereference (*)

- & → gets address
- * → accesses value at address

```
int x = 10;
int *p = &x;

printf("%d", x);   // 10
printf("%d", *p);  // 10
printf("%p", p);   // address of x
```

Pointer Initialization



⚠️ **Uninitialized pointers are dangerous.**

Pointer Types

a. Data Pointer

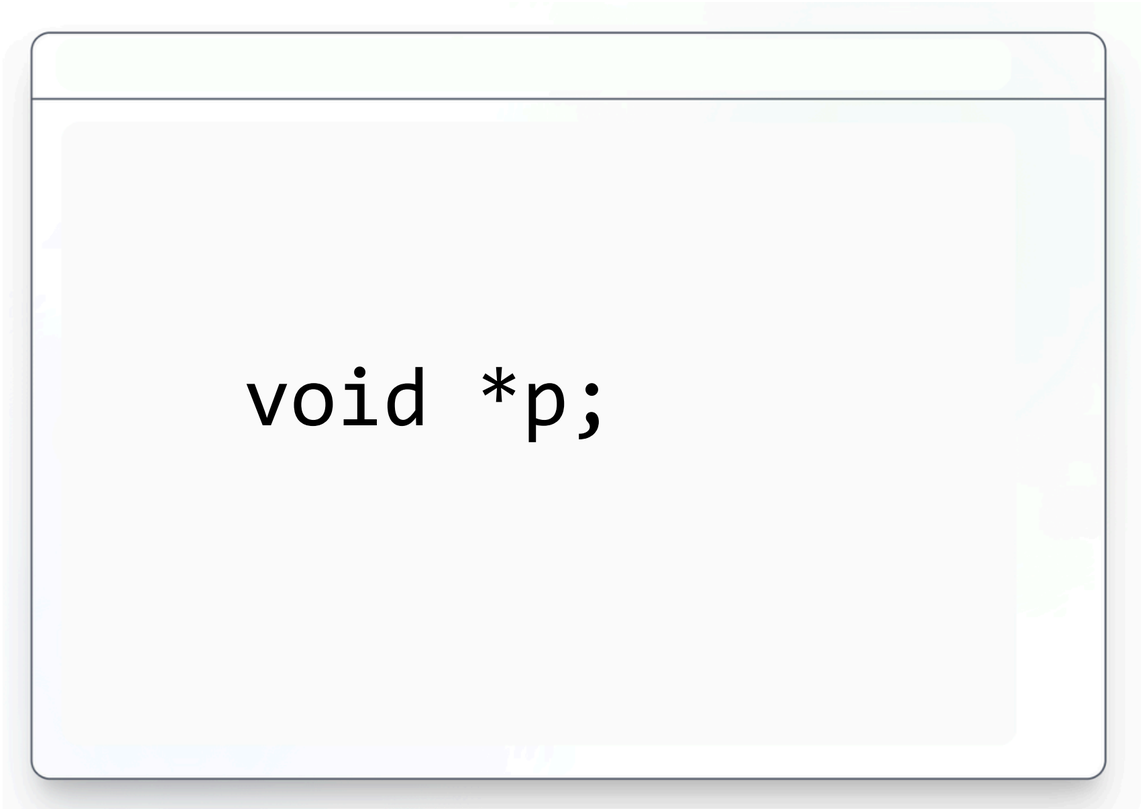
```
int *p;

char *c;
```

b. Null Pointer

```
int *p = NULL;
```

Void Pointer

- Can point to any data type
- Must be type-cast before dereferencing



Wild Pointer

- Points to random memory
- Causes undefined behavior

```
int *p;
// uninitialized
```

Dangling Pointer

Occurs when:

- Memory is freed
- Pointer still points to it

```
free(p);
p = NULL;
```

Pointer Arithmetic

Moves pointer by:

- `sizeof(data_type)`

```
p++;
p--;
```

Pointers and Arrays

```
int arr[3] = {10,20,30};
int *p = arr;

printf("%d", *(p+1)); // 20
```

Constant Pointers

Pointer to Constant

```
const int *p;
```

Constant Pointer

```
int *const p;
```

Constant Pointer to Constant

```
const int *const p;
```

Function Using Pointers (Swap)

```
void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}
```

Activity

Swap two numbers using pointers

Demonstrates:

- Pass by reference
- Memory access

Common Pointer Mistakes

- Using uninitialized pointers
- Forgetting * or &
- Dangling pointers
- Incorrect pointer arithmetic

Session Summary

- Pointers store addresses
- Enable memory-level control
- Essential for advanced C
- Backbone of data structures