

7. POINTERS AND ARRAYS

[Hengfeng Wei \(魏恒峰\)](#)

hfwei@nju.edu.cn



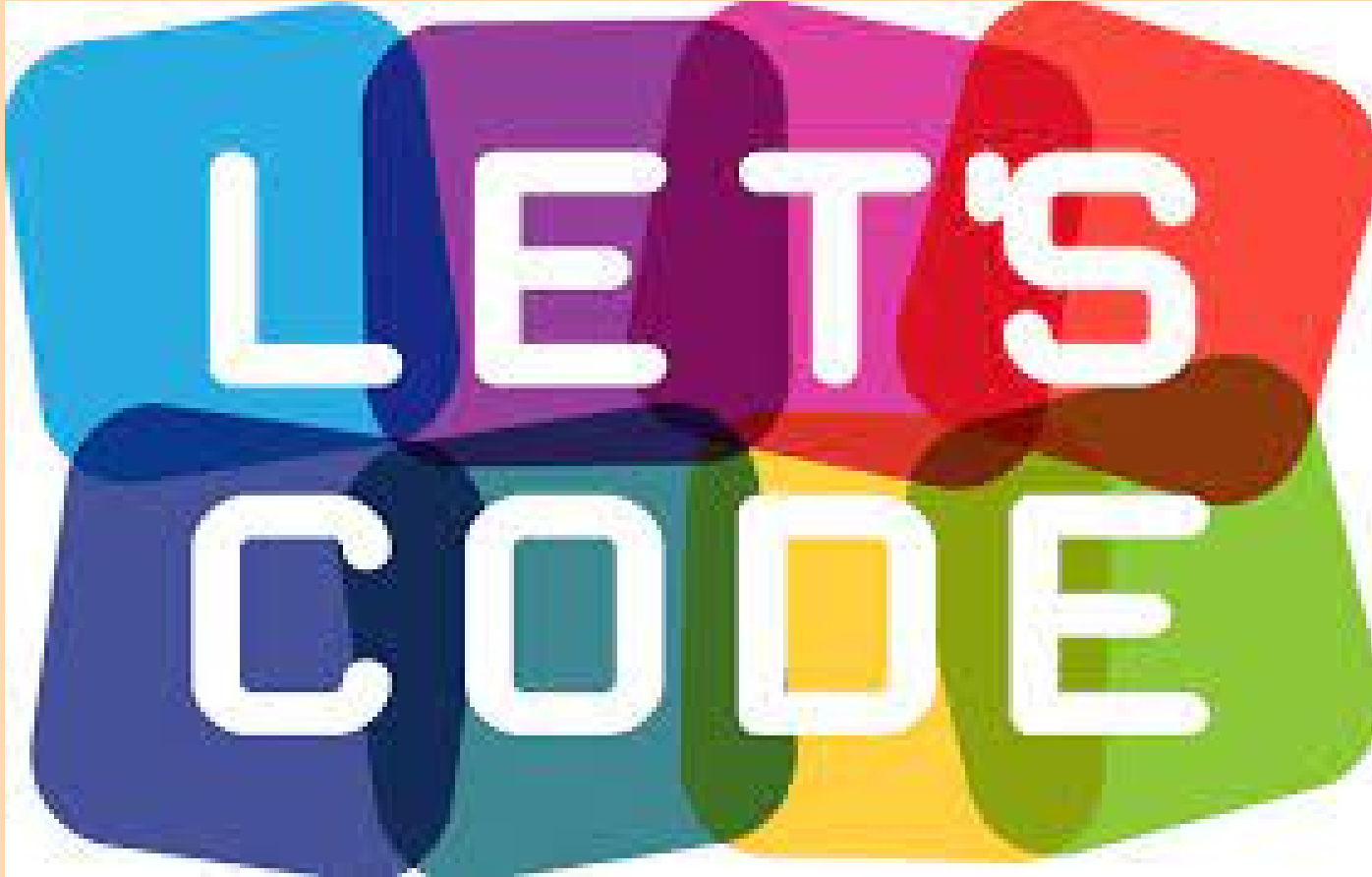
Nov. 04, 2023

Overview



Pointers and Arrays (7 sentences = 4 + 3)

Dynamic Memory Management



pointer.c

selection-sort.c

pointer-array.c

pointer-const.c

Pointers provide an **abstraction** to **memory**.



Memory Address

Object

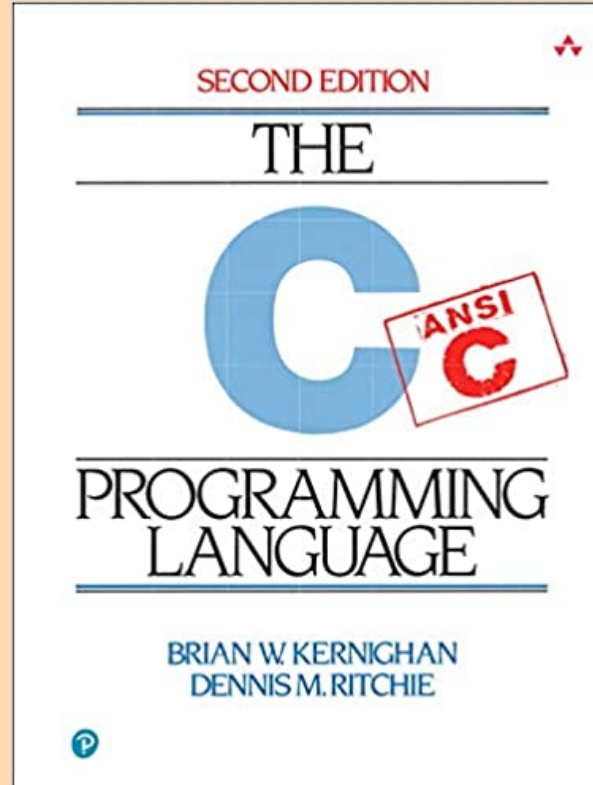
Variable (*identifier*)

Variables (**pointer.c**)

A **variable** has its *type*, *value*, and *address*.

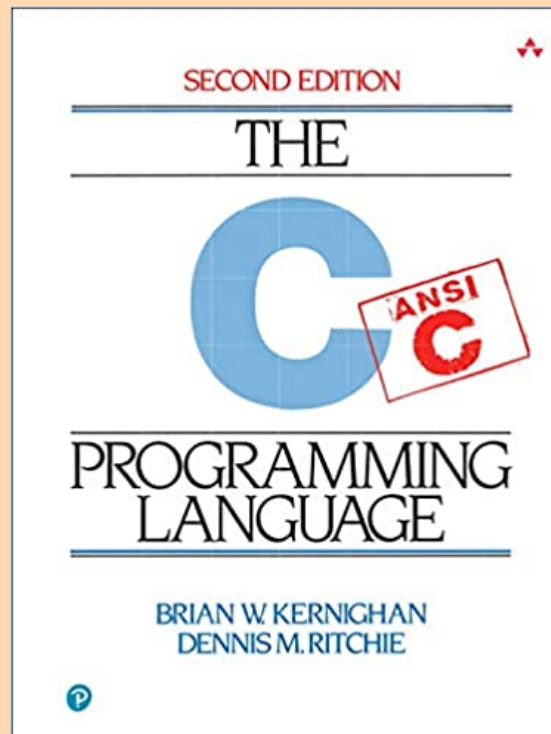
A **variable** can be used as a *lvalue* or a *rvalue*.

"A *pointer* is a *variable* that contains the *address* of a variable."

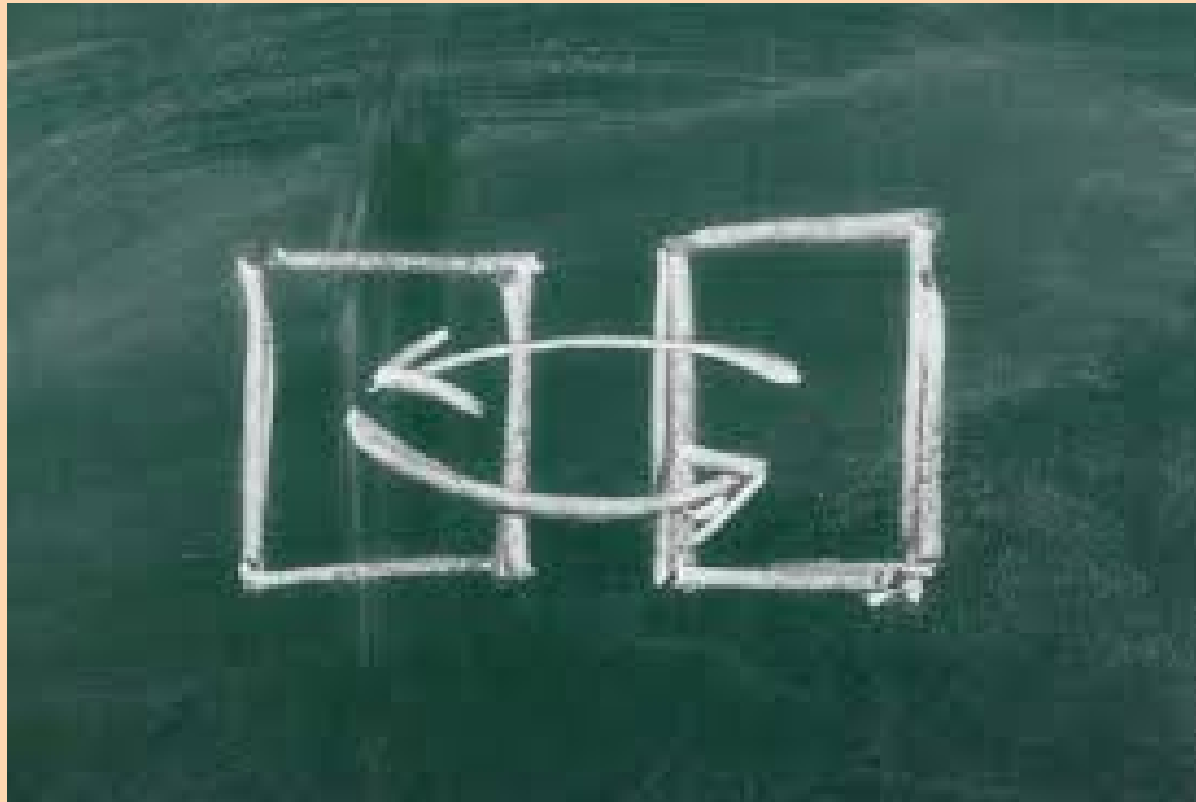


```
int *ptr = &var;
```

"**ptr* can occur in any context where *var* could"



Swap (**selection-sort.c**)



Pointers and Arrays (**selection-sort.c**)

In ***expressions***, the ***name*** of an array is a synonym for the ***address of its first element***.

Pointers and Arrays (**selection-sort.c**)

arr[i] is an ***lvalue***.

Pointers and Arrays (**selection-sort.c**)

But an *array name* is **NOT** a *variable*.

(*unmodifiable lvalue*)

Dynamic Memory Management (selection-sort.c)

void *malloc(size_t size);

void free(void *ptr);

