# Chapter 5: Pointers and Arrays

- A pointer is a variable that contains the address of a variable.
  - Usually lead to more compact and efficient code.
- Pointers and Arrays are closely related.
- When used carelessly they can create impossible to understand programs.

## 5.1 Pointers and Addresses

- A typical machine has an array of consecutively numbered or addressed memory cells.
  - Can be manipulated individually or in contiguous groups.
- A pointer is a group of cells that can hold an address.
  - The pointer holds the address of the information we want.
- The memory address is stored in the pointer using the & operator.
  - '"p = &c"' assigns the address of c to the pointer p.
  - The & operator only applies to objects in memory.
    - variables and array elements.
- The memory is accessed using the * operator.
  - Called the *indirection* or *dereferencing* operator.
  - When applied to a pointer it accesses the object at the memory location held by the pointer.
- A pointer is constrained to point to a particular kind of object.
  - Void pointers are a different story but have special considerations.

## 5.2 Pointers and Function Arguments

- The only way for a function to change the arguments passed to it are via pointers.

- We can access the variables by passing a pointer to the variables.
- Pointer arguments allow a function to access and change objects in the function that calls it.

## 5.3 Pointers and Arrays

- There is a strong relationship between pointers and arrays in C
- Any operation that can be done by array subscripting can be done by using pointers.
    - The pointer version will be faster in general.
    - if pa points to an element of an array then pa + i will point to the element i places after pa.
        - This happens regardless of element size of the array.
- By definition the value of a variable or expression of type array is the address of the first element of the array.
- An array-and-index expression is equivalent to one written as a pointer and offset.
    - One difference is that a pointer is a variable and an array name is not.
- When an array name is passed to a function the location of the initial element is passed.
    - An array name parameter is a pointer.
    - So as an argument we can increment the array name pointer.
- It is possible to pass part of an array to a function.
    - pass the pointer to the beginning of a sub-array.
    - The function doesn't know the difference.

## 5.4 Address Arithmetic

- If p is a pointer to some element of an array then p++ increments p to point to the next element.

    - p+=i increments it to point i elements beyond where it currently points.
- Example using storage allocator:
    - `alloc(n)` returns a pointer p to n-consecutive character positions.
        - they can be used by the caller to store characters.
    - `afree(p)` releases storage acquired so it can be reused later.
    - The calls to `afree()` must be made in reverse order to the ones made to `alloc()` because the storage resides on the stack.

# 5.5 Character Pointers and Functions

- A string constant such as "This is a string" is an array of characters.
    - In the internal representation it is terminated with a null character.
    - The null character makes the length of the storage one more than the number of characters.
- The most common occurrence of string constants is as arguments to functions.
- When a character string appears in a program access to it is through a character pointer.
- String Constants need not be function arguments.
    - Pointers can point to strings.
    - C does not provide any operators for processing an entire string of characters as a unit.
    - There is a difference between the following two definitions:

```
char amessage[] = "now is the time";    /*is an array*/
char *pmessage = "now is the time";   /*is a pointer*/
```

- amessage is an array and just big enough to hold the sequence of characters and '\0'.

    - individual characters may be changed but amessage will always refer to the same storage.
- pmessage is a pointer, initialized to point to a string constant.
    - The pointer may be modified to point elsewhere but it is undefined to change the string contents.

## 5.6 Pointer Arrays; Pointers to Pointers

- Pointers are variables so they may be stored in arrays as well.

## 5.7 Multi-dimensional Arrays

- C provides rectangular multi-dimensional arrays.

## 5.8 Initialization of Pointer Arrays

## 5.9 Pointers vs Multi-dimensional Arrays

- An advantage of pointer arrays is that the rows may have different length.

## 5.10 Command Line Arguments

- There is a way to pass command line arguments or parameters to a program when it begins executing.
- When main is called it is called with two arguments.
    - argc - the number of command line arguments invoked.
    - argv - a pointer to an array of character strings that contain the args.
        - one argument per string.
- Since argv is a pointer to an array of pointers we can manipulate the

pointer rather than index the array.

## 5.11 Pointers to Functions

- In C a function is not a variable itself.
- It is possible to define pointers to functions.
    - The pointers can:
        - be assigned.
        - placed in arrays
        - passed to functions.
        - returned by functions
        - etc.