# Initializing Pointers

**A pointer can be in one of four states:**[1]

1. It can point to a **valid object**.
2. It can point **one-past** a valid object (in an array or `vector` for instance.
3. It can contain the value `nullptr` to indicate it points to "nothing", or is unused.
4. It can **be invalid**, such as an uninitialized pointer.

You can **initialize a pointer** in several ways.

- With the address of another object obtained from the **address operator.**
- With the address of an object **created on the heap** with the `new` operator.
- With the **name of a previously defined array**.
- By using **pointer assignment** to copy the address from another pointer

If you don't initialize a pointer, **it is invalid**. Here are examples of each of these:

```
int x{42}, y{0}, a[10]; // x->int, y->int, a->array

int *p1{&y};          // points to y
int *p2{&x};          // points to x
int *p3{new int{3}}; // points to int on heap
int *p4{a};           // points to first element of a
int *p5{a+10};        // points "one past" the array a
int *p6{nullptr};     // points to "nothing"
int *p7;              // uninitialzed (invalid)
```

[1] Lippmann, C++ Primer, 5th Edition, Page 52, Section 3.3.2