Pointers and Class

Class	Names		
	Public variables:		
	string	first_name	
	string	last_name	
	int	salary[10]	

value						
index	0	1	..	…	..	9

```
class Names{
        public:
                string first_name;
                string last_name;
                int salary[10];
};
```

Pointers and Class

obj ect	n1		
	first_name	mary	
	last_name	smith	
	int	salary[10]	

value	100	90	85			
index	0	1	..	…	..	9

```
class Names{
        public:
                string first_name;
                string last_name;
                int salary[10];
};
```

Pointers and Class

```
A x;
x.a = '7';
x.b = 'b';
x.c = 'a';

A *p; // declare a pointer named p with the type A.
p = &x;
(*p).a = '8';
(*p).b = 'b';
p->b = 'a';
p->r[6] = 5;
cout << x.r[6] << endl;
```

```
int num;

num = 78;

p = &num;

A y;
y.a = '9';
y.b = 'b';
```

Pointers and Class

```
A x;
x.a = '7';
x.b = 'b';
x.c = 'a';

A *p;
p = &x;
(*p).a = '8';
(*p).b = 'b';
p->b = 'a';  ⇔ (*p).b = 'a'
cout << p << endl;
p -> a = 't';
p -> c = 'n';
p -> r[0] = 10;
p->r[6] = 5;
cout << x.r[6] << endl;
cout << p->r[6] << endl;
```

Object	x	With address	21
	Public variables:		
		char	a = '7'
		char	b = 'a'
		char	c = 'a'
		int	r[7]

value	10					5
index	0	1	..	...	..	6

Pointer p

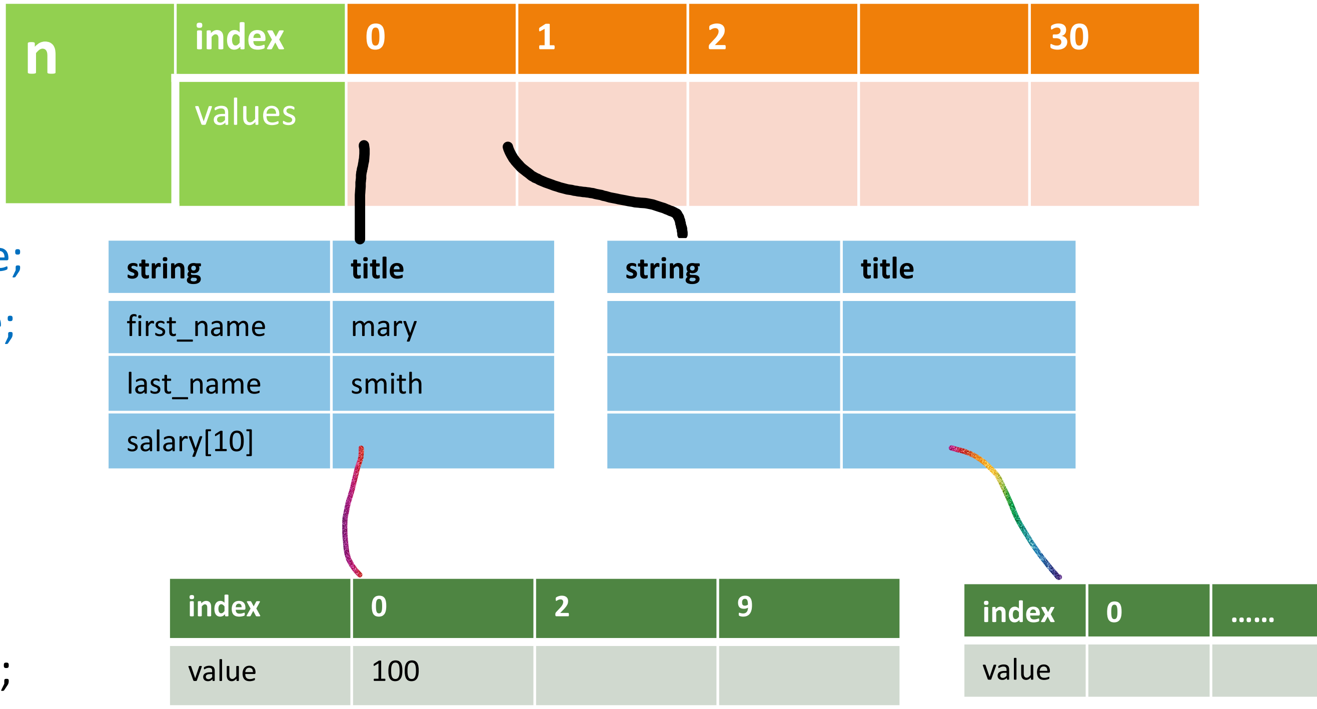
Address	value
100000	21

Pointers and Classes

- A pointer to a class object is no different than a pointer to any other type of variable
- Given:

```
class Names{
    public:
        string first_name;
        string last_name;
        int salary[10];
};

Names n[31];
n[0].first_name = “mary”;
n[0].last_name = “smith”;
n[0].salary[0] = 100;
```



n	index	0	1	2		30
	values					

string	title
first_name	mary
last_name	smith
salary[10]	

string	title

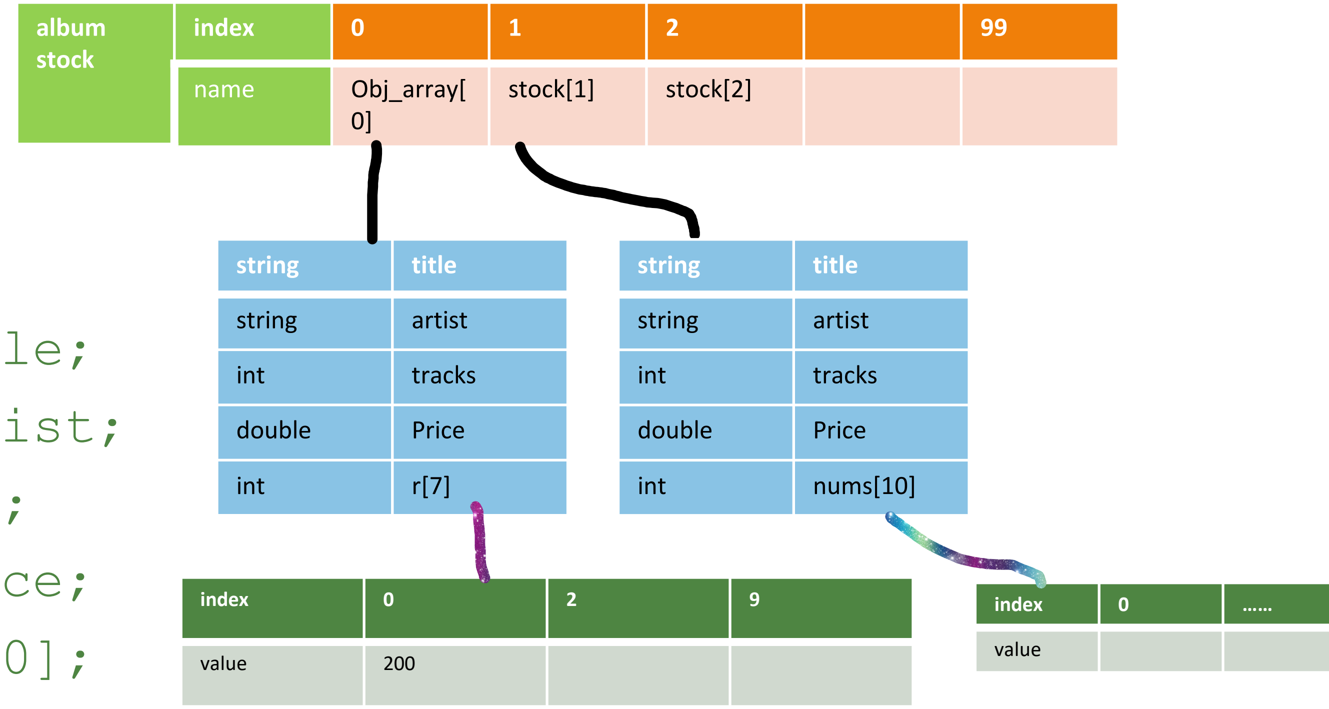
index	0	2	9
value	100		

index	0	……
value		

Pointers and Classes

- A pointer to a class object is no different than a pointer to any other type of variable
- Given:

```
class album
{
public:
   string title;
   string artist;
   int tracks;
   double price;
   int nums[10];
};
album stock[100];
album *pick;
```



Pointers and Classes

- A particular album can be selected by assignment:

```
pick = stock; // album a, *p; p = &a;
pick = pick + 49; // pick + 49 ⇔ stock[49]
or
pick = &(stock[49]);
```

- The members of that album are accessed by a combination of dereference (`*`) and membership (`.`):

```
(*pick).title = “Listener Supported”;
```

- There is also a syntactic shortcut:

```
pick->title = “Listener Supported”;
// pick is a pointer
// point to an object
// one element of the object is title
```

Pointers and Functions

- Pointers, like any variable, can be used as parameters and return values

```
void some_function( int x, album * p );
album * some_other_function( double y );
```

- They are passed by value by default
  - Involves copying the value (an address) into a local variable
  - Changes to a local copy *do not* change the pointer
  - But, changes to the memory the pointer points at are not limited to local variables!

```
void some_function( int x, album * p )
{
    p->artist = "The Black Crowes";        // non-local change
    p = NULL;                              // local change!
}
```