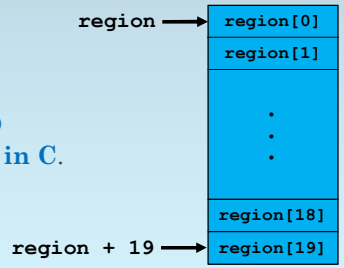
Brackets are Shorthand for Add and Dereference

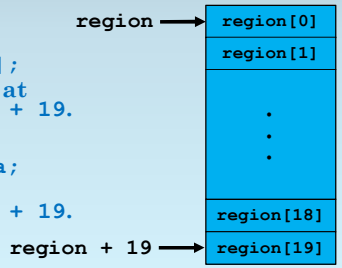
```
int region[20];
```



Thus,
- `region[N]` and
- `*(region + N)`
- are **equivalent in C**.

Use Brackets for Reading and Writing Array Elements

```
int region[20];
int a;
```



Thus
- **`a = region[19];`**
- **reads the value at address `region + 19`.**

And
- **`region[19] = a;`**
- **stores bits to address `region + 19`.**

Pointer Arithmetic Generally Involves Multiplication

```
int region[20];
```

Say that **`region` is address `0x12345000`**.

What is `region + 5`?

`0x12345005`? Not necessarily.

The answer depends on
- the size of an **`int`** and
- the addressability of memory.

The amount added is the number of addresses required for 5 `int`s.

Pass Array Arguments as Pointers

Let's do an **example**. Our task:
- write **a subroutine**
- to **find the minimum value**
- **in an array of `int32_t`s**.

How can we pass the array to the subroutine?

Copy it onto the stack?

Expensive!

Instead, **pass a pointer to the first element!**

