ARRAY 1:
AN `operator[]` EXAMPLE

Overloading the index operator

Delroy A. Brinkerhoff

AN ARRAY WITH SETTABLE BOUNDS

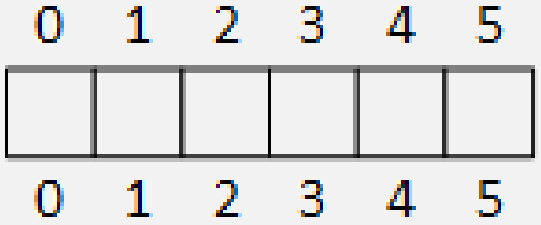
```
class Array
{
    private:
        int    lower;
        int    upper;
        char*  array;

    public:
        Array(int l, int u);
        ~Array() { if (array != nullptr) delete[] array; }
        char& operator[](int index);
};
```

MEMORY ALLOCATION: SIMPLE CASE

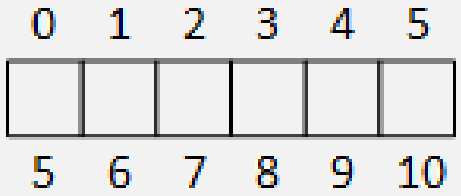
- `upper – lower + 1`

- `Array a(0, 5);`
- `5 – 0 + 1 = 6`

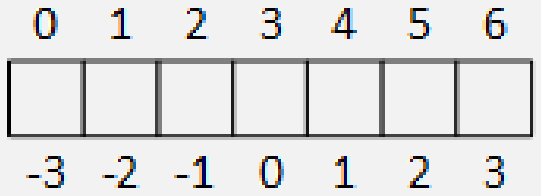


MEMORY ALLOCATION: GENERAL CASE

- `Array b(5, 10);`
- `10 – 5 + 1 = 6`



- `Array c(-3, 3);`
- `3 - -3 + 1 = 7`



THE Array CONSTRUCTOR

```
Array::Array(int l, int u) : lower(l), upper(u)
{
    if (upper < lower)
        throw "upper must be >= lower";

    array = new char[upper - lower + 1];
}
```

OVERLOADING THE INDEX OPERATOR

- Let `index` be the parameter/operand
  - `index – lower`
  - `array[index – lower]`

```
char& Array::operator[](int index)
{
    if (index < lower || index > upper)
        throw "index out of bounds";

    return array[index - lower];
}
```

DEMONSTRATING operator[]

```
Array c(-3, 3);

for (int i = -3; i <= 3; i++)
    c[i] = char(i + 'D');    // l-value

for (int i = -3; i <= 3; i++)
    cout << c[i] << endl;    // r-value
```