Basics of dynamic classes
(classes that use dynamic memory)

Always need ① Copy constructor
② Assignment operator
③ destructor.

Why ①? `vector2(const vector2&);`

used to make copies of existing objects.
Called "behind the scenes" e.g. when you call by value with a vector:

```
Void f(vector2 w) {
    cout << "oh noooo";
}   <- w goes out of scope.
void main() {
    vector2 v;
    v.push_back(0);
    ...
    f(v);
    // X-X
}
```

Default copy constructor:
member-wise copying.
E.g.,
```
vector2(const vector2& v){
    size = v.size;
    capacity = v.capacity;
    data = v.data;
}
```
only copies the pointer

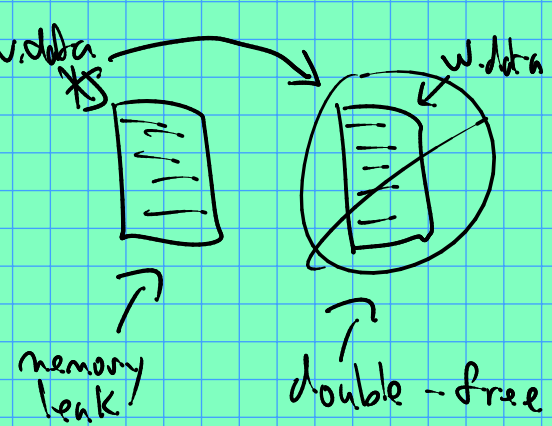
at end of f(...), w's destructor is called, which also deletes v.data :-

② Default assignment operator is similar to the default copy const.: member wise copy:

```
operator = (const vector2& W) {
    size = W.size;
    capacity = W.capacity;
    data = W.data;
}
```

This will almost always break:

```
void f() {
    vector2 V;
    vector2 W;
    ...
    V = W;
}
```




```
void f (vector2& W) {
    vector2 X;
    X = W;
}
```

Similar issues...

X ≠ V will have different scopes. So seg fault might happen before the double free...

```
main() {
    vector2 V;
    f(V);
}
```