Release of Memory: delete

- The operator `delete` deallocates the memory allocated by the operator `new`.

```
Point* p = new Point(1.0, 2.0);
delete p;
```

- If the deleted object belongs to a type hierarchy, more destructors will automatically be called.
- If you deallocated the memory, further access to the object is undefined behavior.

If you deallocate a with `new` allocated object with `delete []`, you will get undefined behavior.

Release of Memory: delete[ ]

- A with `new[]` allocated C array has to be deallocated with `delete[].`

```
Point* p = new Point[15];
delete[] p;
```

- The call of `delete[]` calls in contrast to `delete` all destructors of the C array.

If you deallocate a with `new []` allocated object with `delete,` you will get undefined behavior.

overloadNewAndDelete.cpp
overloadNewAndDelete2.cpp