How's your HW5 so far?

done good

not bad

4

Today's plan

- → Review Ex 10-1
- → Recap questions
- → In-class Ex 10-2

Mentimeter

Ex 10-1: **mean**

- std::vector<double> grades, bool is_sorted
- double mean = 0;
- for (size_t i = 0; i < grades.size(); ++i) {...}
- mean += grades[i];
- if (grades.empty()) return -1;
- else return mean / grades.size();

Mentimeter

Ex 10-1: **median**

- return percentile(50);
- if (grades.empty()) return -1;
- if (grades.size() % 2) return grades[grades.size() / 2];
- else {
- size_t mid_idx = grades.size() / 2;
- return (grades[mid_idx - 1] + gradex[mid_idx]) / 2;
- }

Ex 10-1: accessing **private** member fields

- the field grades is private
- we should use the getter to get its value as a const ref
- gl.grades[i] -> gl.get(i)

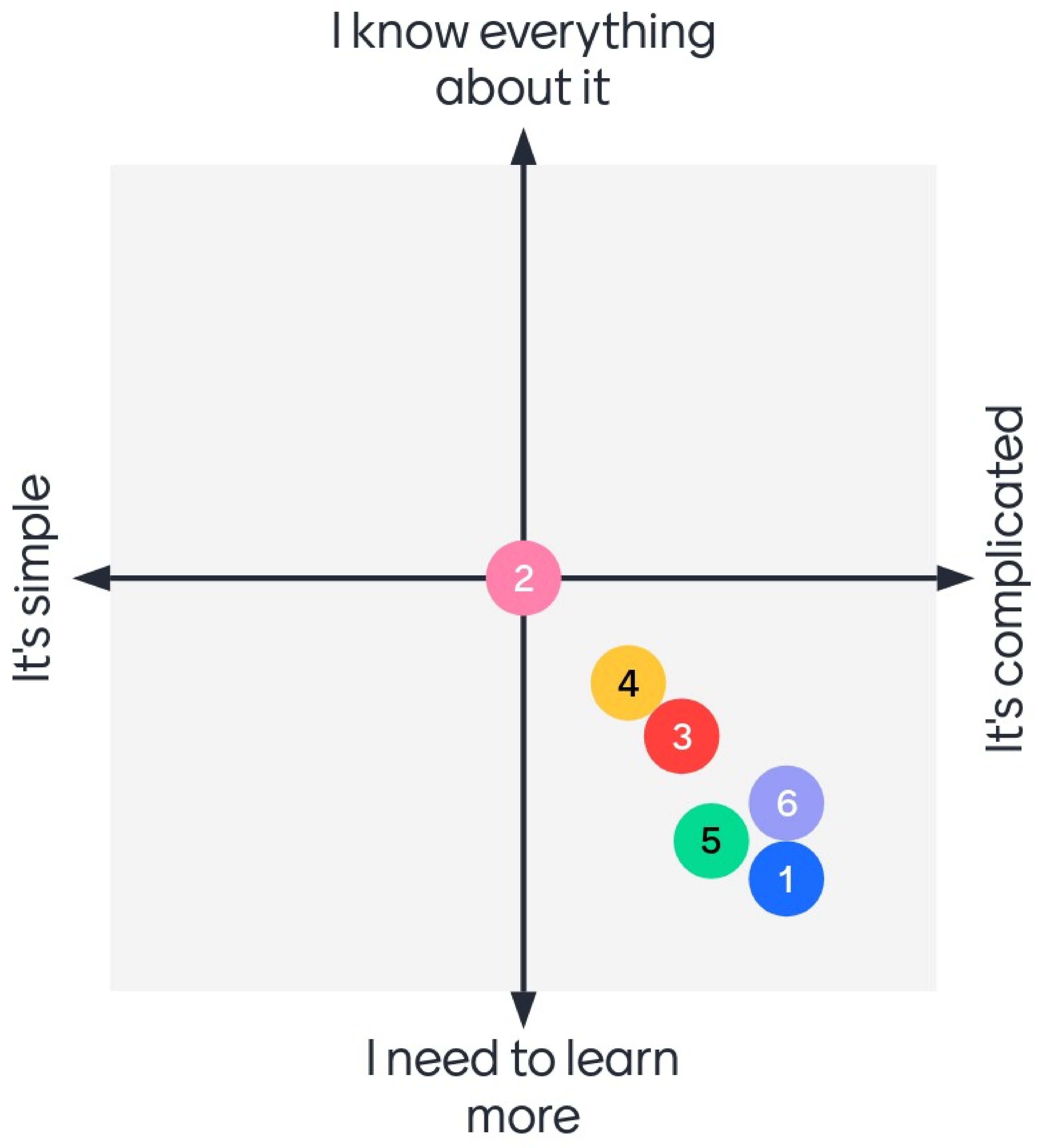
Mentimeter

Ex 10-1: add all even numbers 0-100

- Use the add function
- for (int i = 0; i <= 100; i += 2) gl.add(i);
- gl.mean() to get the mean
- gl.percentile(xx) to get the xxth percentile
- gl.median() to get the median

Self evaluation

Mentimeter



1 Non-default (alternate) constructors

2 Default arguments

3 Name conficts, and use of `this` pointer

4 When will the compiler generate a default constructor for you?

5 `new` and constructor

6 Destructors

2

What is a non-default (alternate) constructor?

Mentimeter

constructor which accepts arguments ✗

not of the class name ✗

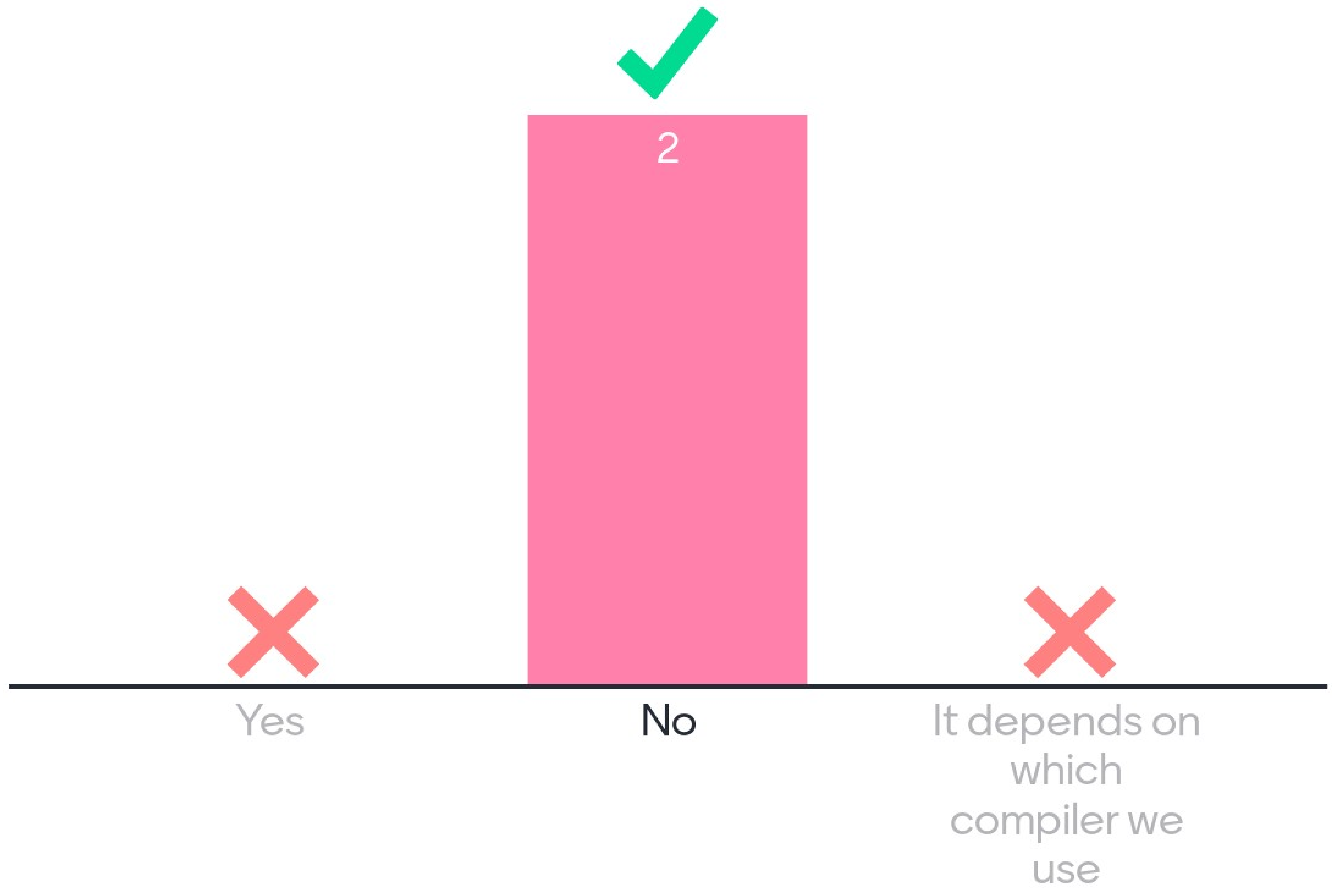
we pass in the value that we want to assign to the variables as parameters ✗

The correct answer is: A constructor that takes arguments for initializing member fields

3

If we define a non-default constructor, will C++ generate an implicitly defined default construtor?

Mentimeter



2

Mentimeter

When do we need to use the **this** keyword?

when comparing to another object's attributes to avoid confusion ✗

refer to field ✗

when the parameter name of a function which is not the constructor is the same as one of the member field ✗

The correct answer is: When the local variables hide the member fields (except constructor's initlializer list).

3

What is a destructor? When do we need to call it?

Mentimeter

destructor is to free dynamically allocated objects. don't need to explicitly call it ✕

invoked when the object goes out of scope ✕

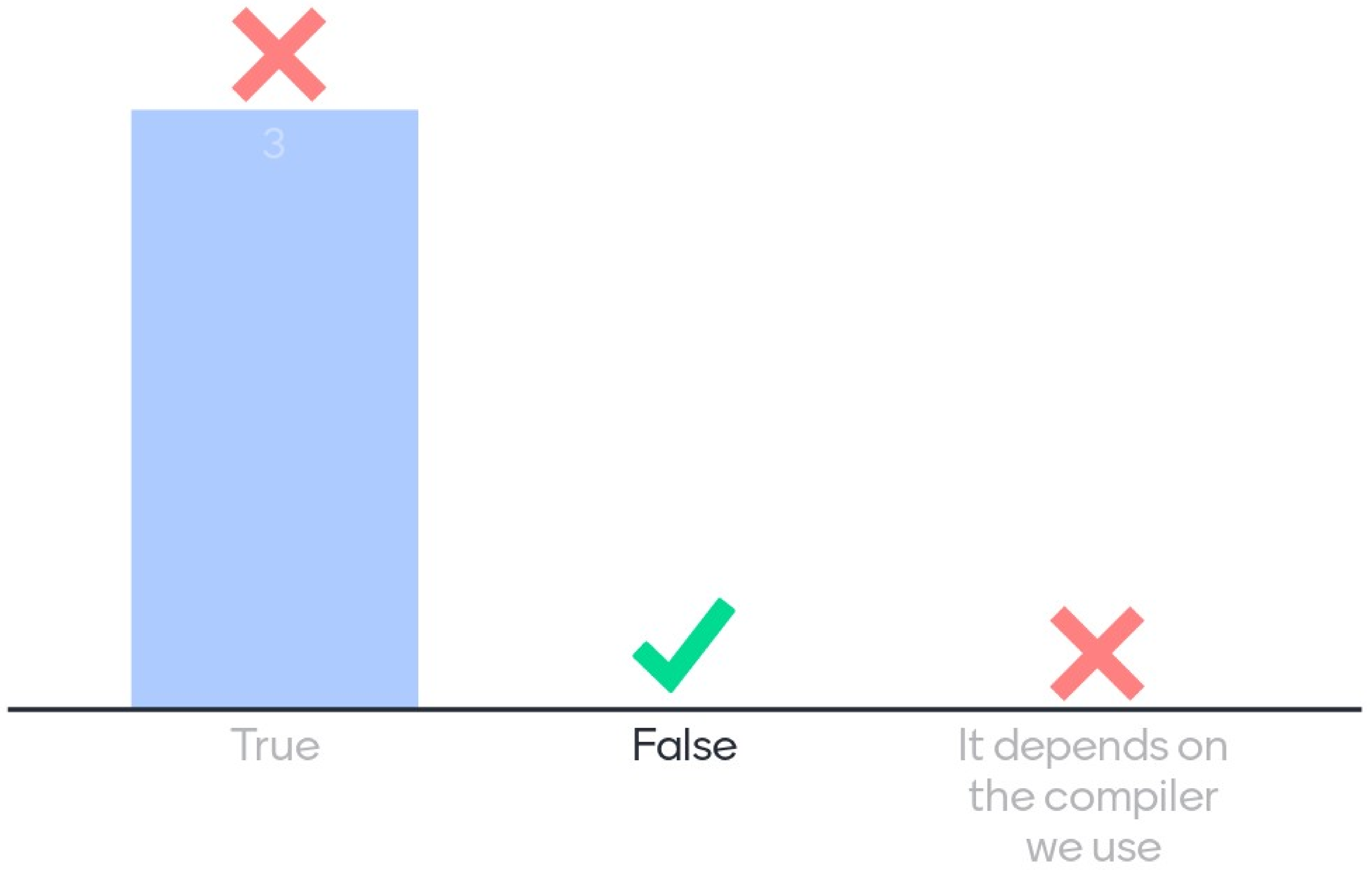
To release the memory when out of the life span of a variable; when we dynamically allocate a memory ✕

The correct answer is: It's a special function that will be called automatically when an object's lifetime ends, or it is deallocated. We don't need to call it explicitly.

3

Mentimeter

A destructor will automatically release memories/resources that are allocated in the constructor. True or False?



3

Ask me anything

Mentimeter

0 questions
0 upvotes