

# Recap

session 1 t/m 3

De constructor van een class wordt aangeroepen wanneer je een object van die class maakt. Welke functie wordt aangeroepen wanneer een object *out of scope* raakt?

- a. Destructor
- b. Deconstructor
- c. Decostructor

De relatie tussen een class met een property, met als type een andere class noem je

- a. Association
- b. Composition
- c. 'Has a' relationship
- d. Alle bovenstaande antwoorden

# Wat betekent het als je een functie override?

- a. Dat je de functie nog een keer alloceert en implementeert, maar dan met andere argumenten.
- b. Dat je in de derived class de functie nog een keer hetzelfde alloceert en implementeert.

# Waarom is constructor delegation verstandig?

- a. Minder typwerk.
- b. Alle initialisatie code op één plek.
- c. Dan kun je meer opties bieden voor hoe je een object aanmaakt.

# Wat is een destructor?

- a. Een destructor wordt aangeroepen wanneer je een object van een class maakt.
- b. Een destructor vernietigt het object.
- c. Een destructor wordt aangeroepen wanneer een object vernietigd wordt.

# Wanneer heb je een virtuele functie nodig?

- a. Dat is altijd handig, alle functies in je base class kun je het beste virtual maken.
- b. Een virtuele functie is handig wanneer je een functie niet wilt implementeren in je base class.
- c. Wanneer je de optie wilt bieden dat een derived class de functie kan *overloaden*.
- d. Wanneer je de optie wilt bieden dat een derived class de functie kan *overriden*.

# Waarom zou je een initializer list gebruiken?

- a. Dan hoef je de waarden niet te initialiseren in de constructor zelf.
- b. Scheelt typwerk.
- c. De waarde wordt geïnitieerd nog voordat je in de constructor komt.
- d. Is overzichtelijk.

# Wat betekent het als je een functie overload?

- a. Dat je de functie nog een keer alloceert en implementeert, maar dan met andere argumenten.
- b. Dat je in de derived class de functie nog een keer hetzelfde alloceert en implementeert.

# Waarom zou je een abstracte class willen?

- a. Wanneer het niet mogelijk zou moeten zijn dat je een object van een base class aan kunt maken.
- b. Wanneer het niet zinvol is om een functie in de base class te implementeren.
- c. Beide bovenstaande antwoorden zijn correct.

```
virtual void getName() {};
```

- a. Deze functie is een pure virtual functie.
- b. Deze functie is een method.
- c. Deze functie is een virtual functie.

*Welke twee uitspraken zijn juist? ... en ... zijn juist.*

`virtual void getName() = 0;`

- a. Deze functie is een pure virtual functie.
- b. Deze functie is een method.
- c. Deze functie is een virtual functie.

*Welke twee uitspraken zijn juist? ... en ... zijn juist.*

Een audio callback krijgt een pointer naar de outputbuffer. Hoezo?

- a. Dat is handig, dan kun je er direct bij.
- b. Dan hoeft er niet een kopie gemaakt te worden van de buffer.
- c. Een array doorgeven als parameter is altijd *pass by reference*, net zoals in Python.