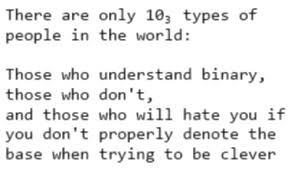
# Practicumopgaven week 2



Bestudeer de hoofstukken van de reader zoals aangeven in het weekoverzicht.

## Katten

Lees de onderstaande code met de class *cat* en de main functie die deze class gebruikt. Wat zijn de waarden van de attributen *pianocat*, *ceilingcat* en *happycat* na het uitvoeren van de code?

|  |
| --- |
| #include "cat.hpp"  int main(int argc, char \*\*argv){  cat pianocat(230,10,6, 1);  cat ceilingcat;  cat happycat = pianocat + ceilingcat;  pianocat \*= ceilingcat;  } |
| main.cpp |

|  |
| --- |
| #ifndef CAT\_HPP  #define CAT\_HPP  #include <algorithm> // std::min  #include <cstdint> // uint8\_t  class cat {  private:  int n\_lives;  // colour of the cat:  uint8\_t red;  uint8\_t green;  uint8\_t blue;  public:  cat( uint8\_t red, uint8\_t green, uint8\_t blue, int n\_lives = 9 ):  n\_lives( n\_lives ), red( red ), green( green ), blue( blue )  {}  cat():  n\_lives(9), red(7), green(4), blue(5)  {}  cat operator+( const cat& rhs ) const {  return cat(  (red + rhs.red ) / 2,  ( green + rhs.green ) / 2,  (blue + rhs.blue ) / 2,  n\_lives + rhs.n\_lives );  }  cat & operator\*=( const cat & rhs ){  red = std::min( red \* rhs.red, 255 );  green = std::min( green \* rhs.green, 255 );  blue = std::min( blue \* rhs.blue, 255 );  n\_lives \*= rhs.n\_lives;  return \*this;  }  };  #endif // CAT\_HPP |
| cat.hpp |

## Doxygen

Installeer indien nodig Doxygen (in het Ubuntu image zit het al). Copieer de inhoud van de week-2-rational directory (run **git pull** in de v1oopc-examples directory als je die week-2-rational directory nog niet hebt), run Doxygen in de directory, bekijk de documentatie die wordt aangemaakt. Er ontbreekt Doxygen documentatie voor één operator (bij het genereren geeft Doxygen een warning). Voeg die documentatie toe en genereer de documentatie opnieuw. Check in de Doxygen uitvoer dat die operator nu wel goed gedocumenteerd is. Zorg dat bij het inleveren de gegenereerde documentatie \*niet\* wordt meegnomen. (zie bv. .gitignore)

## Catch2

Installeer indien nodig Catch2 (in het Ubuntu image zit het al). De week-2-rational directory bevat een Catch2 test voor een library voor rationele getallen (breuken). Zoals je kan zien als je de test runt zitten er een aantal fouten in de library. Verbeter die fouten. Implementeer daartoe eerst een operator<< voor rationele getallen, die de beide getallen (teller en noemer) print in vier posities, hexadecimaal, met voorloopnullen, en 0x ervoor.

## ADT: vector

* Schrijf een vector klasse (hpp en cpp file) met de functionaliteit die aangegeven is in de volgende tabel. Je mag daarvoor code uit de reader en de voorbeelden kopieren.
* Documenteer je klasse met Doxygen commentaar en genereer de documentatie.
* Test je implementatie met Catch2. Schrijf hiertoe voor iedere operator (ten minste) 1 test.
* Gebruik je tests op de ADT implementatie van (ten minste) 1 andere student, en zorg dat (ten minste) 1 andere student zijn implementatie test met jouw tests. Wijzig zo nodig je implementatie en/of tests naar aanleiding van de resultaten. Noteer wie die andere student was, en wat de resultaten waren.

|  |  |  |
| --- | --- | --- |
| **Feature** | **Functie parameters** | **Functie resultaat** |
| atributes | x int |  |
| y int |  |
| constructors | - |  |
| int, int |  |
| + | v | v |
| v, v | v |
| += | v &, v | v & |
| \* | v, int | v |
| int, v | v |
| \*= | v &, int | v & |
| << | std::ostream &, v | std::ostream & |
| v = vector parameter | | |