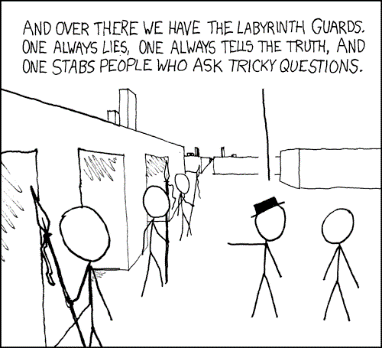
# Practicumopgaven week 3



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

## Auto

|  |
| --- |
| class course {  public:  const std::vector< char > name;  };  class mark {  public:  course & course;  uint8\_t number;  };  class student {  public:  const std::string name;  int number;  std::vector< mark > marks;  };  class klas {  public:  const std::array< char, 3 > name;  std::vector< student > students;  klas(){}  }; |
| class definitions |

Gegeven deze definities, wat is het type van de variabelen a .. x?

|  |
| --- |
| klas v1a;  auto a = v1a;  auto b = a.name;  auto c = a.students[ 3 ];  auto d = c.name;  auto e = c.number;  auto f = c.marks[ 7 ];  auto g = f.course;  auto h = f.number;  auto i = g.name; |
| variables |

## Oefening met pointers en overerving

Geef van de onderstaande code aan of de regels met een getal in het commentaar een error opleveren of niet, en waarom.

|  |
| --- |
| class vehicle { . . . };  class car : public vehicle { . . . };  class train : public vehicle { . . . };  class intercity : public train { . . . };  void travel( vehicle & v );  void ovchipkaart( train & t );  void file( car & c );  car batmobiel;  intercity utr\_amst;  train ICE;  file( batmobiel ); // 1  ovchipkaart( utr\_amst ); // 2  file( ICE ) // 3  ovchipkaart( batmobiel ) // 4  travel( utr\_amst ) // 5  travel( batmobiel ) // 6  intercity & ref7 = utr\_amst; // 7  car & ref8 = utr\_amst; // 8  train & ref9 = utr\_amst; // 9  train & ref10 = batmobiel; // 10  **intercity** & ref11 = batmobiel; // 11  **intercity** & ref12 = ICE; // 12 |
| Overerving, parameters en references |

## Dikkere muren

|  |
| --- |
| D:\2017-2018\vakken\V1OOPC\images\week3\wall.png |
| Een muur |

Neem de 12-10-bounce code als uitgangspunt. Voeg daar een muur subklasse (cpp en hpp files) aan toe. Een muur is een rectangle, maar hij heeft als extra een vlag die aangeeft of hij gevuld of ongevuld (zoals de huidige rechthoek) moet worden afgebeeld. Pas de main aan zodat de muren 4 pixels dik en (initieel) gevuld zijn. Het is niet erg als de muren in de hoeken overlappen. Een muur wisselt iedere toggle\_interval update calls tussen gevuld en ongevuld. Dit aantal is een constructor parameter van de muur.

## Stuiteren

|  |
| --- |
| D:\2017-2018\vakken\V1OOPC\images\week3\bounce.png |
| Een betere stuiter-richting |

Neem de code van de vorige opgave als uitgangspunt. Voeg daar in de drawable een stuiter vector toe, met als default waarde (1,1). Als de bal botst met een object, dan vermenigvuldigd hij zijn speed x en y met de x en y van deze vector in drawable. De default (1,1) heeft dus geen enkel effect. De waarden (-1,1) en (1,-1) zorgen voor een realistisch stuiter effect voor een horizontale resp. een vertikale muur.

Pas de interactie van de bal aan.

Pas de constructor van de muur aan zodat een stuiter-waarde wordt meegegeven. Geef in de main de muren een passende stuiter waarde, zodat de bal realistischer tegen een muur stuitert.

## Verdwijnen

Kopieer de code van je vorige opgave naar een nieuwe directory. Voeg een nieuwe klasse slachtoffer (victim) toe. Een victim is een (ongevulde) rechthoek die zodra de bal hem heeft aangeraakt bij iedere update stapje kleiner wordt. Hiervoor moet je zowel de plek als de omvang van het slachtoffer aanpassen. Teken (mag met een pen, in een afdruk van het UML diagram met de hiërargie van drawables) deze nieuwe klasse. De bal hoeft niet tegen een slachtoffer te stuiteren (dat is best lastig omdat je de richting niet van te voren weet), hij kan er gewoon overheen gaan.