Bestudeer de video van de volgende les geheel:

Klassen, objecten en modules

Experimenteer zelfstandig met de daarin behandelde voorbeeld-programma's. Een link naar de broncode daarvan staat onder de video. In de volgende les kun je hierover vragen stellen.

Opgaven bij de huidige les:

In te leveren uiterlijk 2 dagen vóór de volgende les.

Het programma heeft een voorgeschreven opbouw. Verbeteringen hierop zijn toegestaan in overleg met de docent. Begin je bron-bestand weer met requirements, testspecs en design.

Werk nog niet met classes en methods. Het doel van deze opgave is namelijk om te laten zijn hoe klein eigenlijk de stap is van functie-gestructureerde talen zoals C naar object georienteerde talen zoals C++.

Maak een programma dat een struct type *Vec3D* declareert. Variabelen van dit type zijn vectoren met 3 componenten. Zo'n declaratie ziet er in C++ als volgt uit:

```
struct Vec3D {float x, y, z;};
```

Maak hierbij de volgende functies:

Een functie met signature `Vec3D vec3D (float x, float y, float z)` die variabelen van dit type creeert, initialiseert en returnt.

Een functie met signature `void show (st::string label, Vec3D const &self)` die variablen van dit type afdrukt in een console window, gelabeld met hun naam en afgesloten met een newline.

Een functie met signature `void show (st::string label, float scalar)` die floating point scalars afdrukt in een console window, gelabeld met hun naam en afgesloten met een newline.

Een functie met signature `void show ()` die alleen een newline afdrukt.

Een functie met signature `Vec3D minus (Vec3D const &self)` die een vector returnt die de andere kant op wijst als *self*.

Een functie met signature `Vec3D add (Vec3D const &self, Vec3D const &other)` die de som van *self* en *other* returnt.

Een functie met signature `Vec3D sub (Vec3D const &self, Vec3D const &other)` die het verschil van *self* en *other* returnt.

Een functie met signature `Vec3D mul (Vec3D const &self, float scalar)` die het product van *self* en *scalar* returnt.

Een functie met signature `Vec3D div (Vec3D const &self, float scalar)` die het quotient van *self* en *scalar* returnt.

Een functie met signature float norm (Vec3D const &self) die de norm (lengte) van *self* returnt.

Een functie met signature Vec3D unit (Vec3D const &self) die een vector returnt met dezelfde richting als z'n *self*, maar met lengte 1.

Een functie met signature float dot (Vec3D const &self, Vec3D const &other) die het inproduct (dot product) van *self* en *other* returnt.

Een functie met signature Vec3D cross (Vec3D const &self, Vec3D const &other) die het uitproduct (cross product) van *self* en *other* returnt.

De hoofdfunctie met signature int main () die al deze functies test in overeenstemming met de testpecs die je hebt opgesteld.

Ook deze opgave is weer een stap richting de eindopgave. Termen zoals *som, verschil, product* en *quotient* worden uitgelegd in de les *Getallen* van de module *Toegepaste wiskunde*. De overige benodigde wiskunde wordt uitgelegd in de les *Vergelijkingen, matrices en vectoren* uit diezelfde module, inclusief de begrippen *vector*, *inproduct* en *uitproduct*. Dezelfde zaken worden ook behandeld in het vak Lineaire Algebra. Je kunt de video's echter zelfstandig bestuderen.

De beoordeling van deze huiswerkopgave maakt deel uit van je eindcijfer. Wat beoordeeld wordt is niet alleen een al dan niet werkend programma, maar vooral of je zowel in grote lijnen als in details begrepen hebt wat je aan het doen bent en de gemaakte keuzen kunt motiveren. Programma en verstandig gedoseerd commentaar weer in het engels.

--//--