EINDOPDRACHT C++ 1

Maak een text based game in de vorm van een console application. De output is dus enkel ascii characters die via een console getoond worden aan de gebruiker. Het gaat hier om een handelsspel in de tijd van eind 16de, begin 17de eeuw. Vaar met je schip van haven tot haven, neem lading in, verkoop die in een andere haven, en verdedig je op zee tegen piraten.

HAVENS

Ben je in een haven, dan kun je de volgende acties ondernemen:

1. ***Inkopen/Verkopen goederen.***
   De prijzen en hoeveelheden worden bij het binnengaan van de haven vastgesteld. De prijs die op dat moment geldt voor ieder handelsgoed kun je bepalen aan de hand van het bereik wat gegeven wordt in de *goederen prijzen.*csv file. De beschikbare hoeveelheid per handelsgoed dat op dat moment aanwezig is kun je bepalen aan de hand van het bereik wat gegeven wordt in de *goederen hoeveelheid.*csv file. Het is uiteraard niet mogelijk voor een schip om meer lading te vervoeren als dat er laadruimte is.
2. ***Kopen/Verkopen kanonnen.***
   Er zijn drie soorten kanonnen: lichte kanonnen, middelgrote kanonnen, en zware kanonnen. De hoeveelheden van ieder die worden aangeboden in een haven wordt bij het binnengaan vastgesteld volgens de volgende verdeling:
   a. Lichte kanonnen: 0 – 5 stuks, kosten: 50 goudstukken per kanon
   b. Middelgrote kanonnen: 0 – 3 stuks, kosten: 200 goudstukken per kanon
   c. Zware kanonnen: 0 – 2 stuks, kosten: 1000 goudstukken per kanon

   Je kunt uiteraard niet meer kanonnen kopen dan dat er ruimte is voor kanonnen op je schip. Kleine schepen zijn niet in staat om zware kanonnen aan boord te hebben (zie ook bijzonderheden in de paragraaf Schepen). Je kunt kanonnen altijd verkopen. Je krijgt dan de helft van de koop prijs.
3. ***Kopen/Verkopen van een nieuw schip.***
   Je mag altijd maar één schip hebben, dus als je een nieuw schip koopt wordt je oude schip automatisch verkocht. Voor het oude schip krijg je 50% van de aankoop prijs terug. Je kunt niet alleen je schip verkopen (want dan zou je zonder schip blijven zitten). Welke schepen er beschikbaar zijn om te kopen wordt willekeurig bepaald. De prijs van schepen kun je vinden in de *schepen.csv* file.
4. ***Wegvaren***.
   De speler krijgt een overzicht van welke havens er nog meer zijn en hoeveel beurten varen die van de huidige haven vandaan liggen. Hier kan dan een selectie uit worden gemaakt waarna het schip zich niet meer in de haven bevindt, maar op zee. De afstanden tussen verschillende steden kun je vinden in de *afstanden tussen steden.csv* file.
5. ***Repareren***.
   Je kunt voor 1 goudstuk per 10 schadepunten je schip laten repareren.
6. ***Quit***.
   Je kunt stoppen met het spel. (De console application dient dan netjes afgesloten te worden zodat eventuele memory leaks zichtbaar worden in de development omgeving.)

OP ZEE

Afhankelijk van waar je naar toe vaart duurt het een aantal beurten voordat je bent aangekomen bij de haven waar je naar toe op weg bent. Elke beurt worden de volgende dingen random bepaald:

Er is 20% kans dat er een piratenschip verschijnt. Indien dat het geval is breekt er een gevecht uit (zie paragraafVechten). Wat voor soort piratenschip, hoeveel en wat voor soort kanonnen wordt random bepaald.

1. Bepaal een random getal van 1 – 20 en check de volgende tabel:

Random Getal	**Wind**	**Gevolg**
1 – 2	Geen	Het schip heeft niet bewogen. Je bent dus even ver van de haven verwijdert als vorige beurt.
3 – 4	Briesje	Heeft het schip als bijzonderheid “licht” dan geldt hetzelfde effect als bij normale wind. Zo niet, dan geldt hetzelfde effect als bij geen wind.
5 – 7	Zwak	Heeft het schip als bijzonderheid “log” dan geldt hetzelfde effect als bij geen wind. Zo niet, dan geldt hetzelfde effect als bij normale wind.
8 – 17	Normaal	Het schip heeft gevaren en is daarom één beurt dichter genaderd tot de haven die het tot doel heeft.
18 – 19	Sterk	Er staat zo veel wind dat het schip twee beurten dichter genaderd is tot de haven die het tot doel heeft.
20	Storm	Er is 40% kans dat het schip van de koers is geblazen en er één beurt langer over doet om bij de haven te komen die het tot doel heeft. Er is 40% kans dat het schip niet bewogen heeft (hetzelfde resultaat dus als geen wind). Er is 20% kans dat het schip de goede richting in geblazen is (hetzelfde resultaat als normale wind). Welk van voorgaande drie resultaten ook, het schip heeft sowieso averij opgelopen en verliest 1 – 100% van zijn schadepunten. Indien het schip eindigt met 0 of minder schadepunten is het dus in de storm gezonken.

VECHTEN

Een gevecht bestaat uit één of meer ronden. Elke ronde kan de speler bepalen welke van de volgende acties hij onderneemt:

1. Schieten.

   Eerst schiet jouw schip, is het piratenschip nog niet gezonken dan schiet het piratenschip. Ieder kanon doet random schade afhankelijk van zijn grootte:

   a. Ieder licht kanon: 0 – 2 schade

   b. Ieder middelgroot kanon: 0 – 4 schade

   c. Ieder zwaar kanon: 0 – 6 schade

   Schade gaat direct af van je schadepunten. Indien een schip wordt gereduceerd tot 0 of minder punten, zinkt het. Indien het spelerschip zinkt, is het spel voorbij.

2. Vluchten.

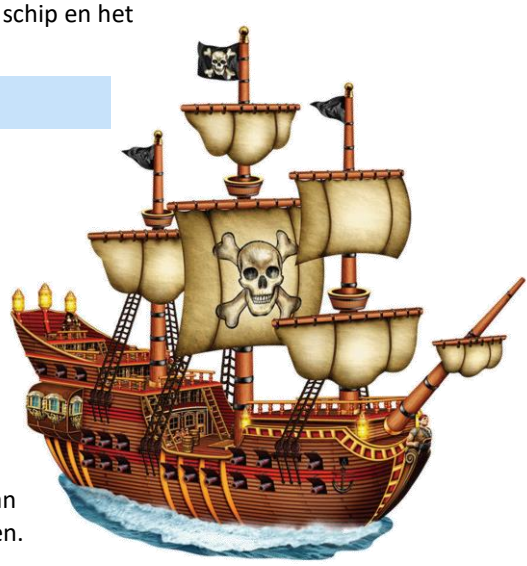
   Afhankelijk van de bijzonderheden van je schip heb je een kans dat je vlucht lukt. Zie hiervoor de onderstaande tabel. Of dit nou lukt of niet, de piraten schieten deze ronde nog wel op je. Slaagt de vlucht, dan is het gevecht voorbij. Zo niet gaat het gevecht gewoon door.

	Licht Piraten Schip	**Normaal Piraten Schip**	**Log Piraten Schip**
Jouw Schip is Licht	Vluchtkans 50%	Vluchtkans 60%	Vluchtkans 75%
Jouw Schip is Normaal	Vluchtkans 30%	Vluchtkans 40%	Vluchtkans 55%
Jouw Schip is Log	Vluchtkans 5%	Vluchtkans 15%	Vluchtkans 30%

3. Overgave.

   Je kunt besluiten je over te geven aan de piraten. Dan stelen de piraten alle lading die er in hun schip past, en gooien de rest overboord (zo zijn piraten). Ze laten je daarna gaan met een leeg schip en het gevecht is voorbij.

SCHEPEN



Elk schip heeft de volgende eigenschappen:

Type: De naam van het soort schip.

Prijs: Het aantal goudstukken dat een speler moet betalen om dit schip te kopen.

Laadruimte: Hoeveel ton lading een schip kan vervoeren.

Kanonnen: Het aantal kanonnen waarvoor ruimte is.

Schadepunten: Hoeveel schade een schip kan incasseren voordat het zinkt (en het spel voorbij is).

Bijzonderheden: Een schip kan *licht* of *log* zijn (daar heeft hij voor of nadeel mee bij het bewegen over zee). Een schip kan *klein* zijn (dan kan hij geen zware kanonnen aan boord nemen). Het is ook mogelijk voor een schip om geen bijzonderheden te hebben.

De eigenschappen van schepen kunnen gehaald worden uit de *schepen.csv* file die bij deze opdracht zit.

WINNEN/VERLIEZEN

Het spel is verloren indien het schip van de speler zinkt (het is al zijn schadepunten kwijt).

Het spel is gewonnen indien de speler er in slaagt om 1.000.000 goudstukken te verzamelen.

MINIMUM EISEN

De volgende eisen zijn **minimaal** verplicht om te implementeren. **Wanneer één of meer van deze eisen niet geïmplementeerd zijn krijg je automatisch een 1 voor de eindopdracht.**

1. Vóór de deadline verstreken is moet de laatste versie van de opdracht zijn ingeleverd op blackboard voor beide studenten.
2. Een werkend spel, geprogrammeerd in C++, inclusief alle dingen die in de voorgaande paragrafen beschreven staan. **Een werkend spel is een spel dat niet crasht!**
3. Er moet een vorm van *memory leak detection* beschikbaar zijn ten tijde van het assessment
4. Er mogen geen memory leaks optreden in het programma*
5. Er mag alleen maar van **raw pointers** gebruik worden gemaakt (zoals beschreven in de eerste presentatie van week 2), niet van smart pointers (waar we in week 4 op vooruitgeblikt hebben).
6. Code is **const correct**
7. Alle code moet exception safe zijn. (notabene: dit is niet hetzelfde als minimum eis 8!)
8. Er moet aan exception handling gedaan worden.
9. Zodra een object niet meer gebruikt wordt, **moet** het object zo snel mogelijk uit het geheugen worden verwijdert.
10. Het is **niet** toegestaan *STL containers* te gebruiken. Dus geen vectors, maps, queues, STL arrays, etc. Het is **wel** toegestaan om "gewone" arrays te gebruiken (dus arrays waarvan je zelf de dimensies bepaald).
11. Het is **niet** toegestaan strings te gebruiken. In plaats daarvan zul je gebruik moeten maken van character arrays.
12. Als je casts gebruikt mag er ***alleen*** gebruik worden gemaakt van **C++ style casts** (static_cast, const_cast, dynamic_cast, en reinterpret_cast).
13. Voor het genereren van random nummers mag er *niet* gebruik worden gemaakt van rand(), maar moet er gebruik worden gemaakt van de **<random>** header. Zie de *main.cpp* file bij deze opdracht voor voorbeelden hoe deze te gebruiken.
14. Het gebruik van externe libraries zoals Boost of Poco is niet toegestaan

* Memory leaks waar jullie niets aan kunnen doen (mochten die in bijv. de Standard Library gesignaleerd worden) mogen wel voorkomen, maar dan moeten jullie wel kunnen beredeneren dat deze niet veroorzaakt worden door jullie code. "False positives" (de memory leak detection tool geeft een leak aan, maar dit is niet correct) mogen ook voorkomen. In geval van twijfel vraag tijdig even de mening van je practicum docent.

Er moet in duo's gewerkt worden. Alleen na nadrukkelijke toestemming van je practicum docent mag je in je eentje de eindopdracht doen. Het is niet toegestaan de eindopdracht met drie of meer mensen te doen.

ASSESSMENT

REPARATIE

Het is niet toegestaan om reparaties te doen *tijdens* het assessment. Ook is het niet toegestaan om een reparatie te doen en later op de dag het assessment te herhalen. In andere woorden: als je applicatie **crasht** of **memory leaks** vertoont, of op een andere manier niet voldoet aan de minimum eisen krijg je een onvoldoende voor de eindopdracht en zul je het assessment opnieuw moeten doen aan het eind van volgend blok. **Zorg dus dat jullie werk uitvoerig getest is voordat je bij het assessment verschijnt!**

BONUSPUNTEN

Er kunnen bonuspunten gehaald worden voor het implementeren van de volgende zaken:

1. Overal waar nodig maar ten minste één keer de *rule of five* geïmplementeerd in de code [+0,5 punt]
2. Er wordt gebruik gemaakt van move constructors en move assignment operators en jullie kunnen uitleggen waarom. (De move constructor/assignment operator zijn dus niet alleen maar aanwezig, maar deze worden daadwerkelijk gebruikt.) [+0,5 punt]
3. Alle bijzonderheden uit de C++ presentaties die terugkomen in de code. [in totaal tot +1 punt]
4. De applicatie is netjes en degelijk ontworpen. Jullie moeten kunnen uitleggen waarom dit het geval is. Vereiste hierbij is de aanwezigheid van UML diagrammen die representatief zijn voor de ingeleverde applicatie (dus geen oude versie van de diagrammen, maar geüpdate diagrammen) [tot +1 punt]

MALUSPUNTEN

1. Indien je tijdens het assessment niet voor ieder object correct uit kan leggen of het zich op de stack, de heap, of in de global memory space bevindt, en waarom. [-1 punt]
2. Iedere student van een duo moet *alle* code kunnen uitleggen. Ook de code die de ander geschreven heeft. Is dat niet het geval dan krijgt die student individueel maluspunten. [-0.5 tot -1 punt *per geval*]
3. Geen duidelijk concept van eigenaarschap van objecten in de code hebben of kunnen uitleggen (wie verantwoordelijk is voor het aanmaken en vrijgeven van objecten) [-0.5 punt]

BEOORDELING

Indien alle minimum eisen foutloos geïmplementeerd zijn wordt de eindopdracht met een 7 beoordeeld. Dit cijfer kan verder worden verhoogd en/of verlaagd door de zaken die genoemd staan bij Bonuspunten en Maluspunten.