SOP Opdracht 7

Wrappers:
Voor de base datatypes wrapper classes gemaakt, en vervolgens voor ieder type property subclasses van de base wrapper gemaakt.
De properties zijn gewrapt op basis van hun doel, price zit bijvoorbeeld in een int pricewrapper. Dit is gedaan omdat price dan als het in de toekomst bijvoorbeeld als euro's geformatteerd moet worden alleen de tostring functie ge-override zou hoeven worden.

Instance variabelen per class:

Geinventariseerd welke properties elke class nodig had. Daaruit bleek dat elke part een price, title, en type nodig heeft. Door subclasses te gebruiken is type niet meer nodig, daardoor blijft er title en price over.
Daarna gekeken of er classes waren die nog meer properties sharen, dit bleek alleen te gelden voor de subtype property, dus er is een class die Part extend en de property subtype toevoegt. De rest van de classes hebben nu alleen properties die voor henzelf gelden.
Op deze manier heeft elke subclass alleen de data die nodig is, en geen data die alleen voor andere types relevant is.
De Part class is abstract omdat het niet de bedoeling is dat er een Part geinitialiseerd wordt.

Een niveau nesting per method:
De enige plek waar dit een probleem was was bij de computer.isComplete() method. Dit is opgelost door in plaats van een aantal booleans en checks te doen een lijst van vereiste classes op te bouwen en te kijken of die allemaal aanwezig zijn. Dit is nu mogelijk omdat elk type een eigen subclass heeft.

Geen else:
Dit was op 2 plekken het geval:

Bij computer.IsComplete(), daar is het opgelost door simpelweg list.isEmpty() te returnen.
In Main. Hier is het probleem opgelost door als er van de default flow (de computer is compleet) afgeweken wordt, een andere message te printen en dat te returnen. De else is nu onderdeel geworden van de default flow.

Geen afkortingen:

Opgelost door afkortingen te zoeken en die te vervangen door volledige variabelenamen.

Klasse en methode lengte:

De klasses en methodes zijn aan de hand van een definitie op internet gepromoveerd tot kort. Alle klasses zijn korten dan 50 regels, en alle klasses bevatten minder dan 7 methodes,

deze restricties zijn gevonden op
http://binstock.blogspot.nl/2008/04/perfecting-oos-small-classes-and-short.html

Dat gezegd hebbende vinden wij dat de lengte van een klasse niet keihard gelimiteerd moet zijn maar dat een klasse maar een ding hoort te doen, en dat zullen sommige klasses wat langer of korter zijn, maar dan zijn ze logisch gesorteerd in plaats van op een psuedo-willekeurige wijze om maar aan een codestijl te voldoen.