Assignment 1 : Strategy pattern

Mark Ehrhart  
Johri van Eerd

# Wat is de opdracht:

In de opdracht passen we het Strategy pattern toe op een probleem. Het probleem is dat een operating system uit verschillende zoek algoritmes wilt kunnen kiezen voor het opzoeken van data.

Om het probleem en opdracht te simuleren hebben we in C# een command line interface programma geschreven. In de GUI kan de gebruiker uit verschillende zoek algoritmes kiezen, Shortest Seek Time First, Circular-scan, First Come First Serve. De gebruiker kan ook diverse zoek acties invoeren en uitvoeren.

# Je eigen implementatie:

# Discuss the 3 “problems”:

**Reusability**Als basis is het pattern zonder de “command line interface” klasse eenvoudig opnieuw te gebruiken. Dit heeft wel als voorwaarde dat een vervangende user interface voldoende heeft aan de data die vanuit onze “context” klasse geleverd wordt. Ofdat een vervangend strategy pattern data in hetzelfde formaat aan kan leveren.

**Extensibility**

Na onze eerste implementatie met twee patterns nog een pattern eenvoudig kunnen toevoegen. Er bestaan geen harde afhankelijkheden zodat alles wat aan IStrategy voldoet door de CLI aan context toegekend kan worden. Dan is het alleen een kwestie van het algoritme implementeren en zorgen dat de nieuw strategie aan onze output voorschriften voldoet.

Voor het toevoegen van een nieuwe strategie hoeven geen klasses uit het pattern aan gepast te worden hiermee voldoet onze implementatie aan het Open/Closed Principle.

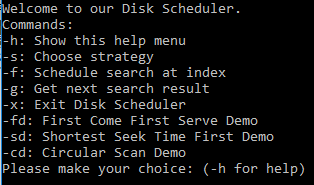
**Maintainability**

Elke strategy is individueel aan te passen van elkaar, hierdoor hebben wijzigingen in een pattern geen gevolg of invloed op een andere strategy. Omdat we, zoals voorgeschreven voor het strategy pattern, gebruik gemaakt hebben van aggregaties in plaats van inheritance van klasses, kan elk stukje gedrag aangepast worden zonder dat klasses die gebruik maken van het gedrag kapot gaan, en andersom kunnen ook stukken gedrag gewijzigd worden terwijl de klasses die daar gebruik van maken daardoor onaangeroerd blijven.

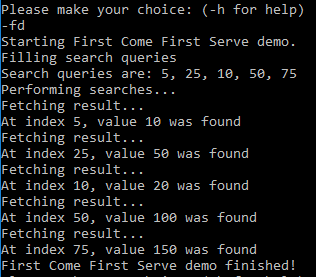
# Tests die je hebt gedaan:

Voor elk zoek algoritme dat we hebben geïmplementeerd hebben we een demo optie in het programma gemaakt. Voor elk zoek algoritme hieronder een screenshot van het resultaat.  
Onze data bestaande uit 100 indexen zijn ieder gevuld met een **int** van (index\*2) zo is makkelijk te verifiëren of ons zoek resultaat ook daadwerkelijk klopt.

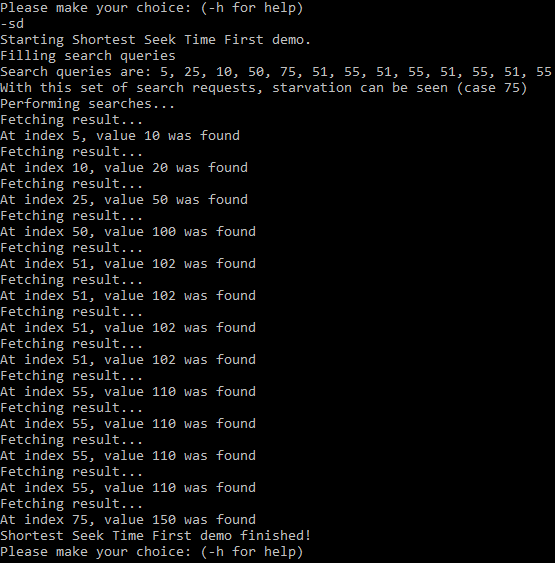
Application start:



First Come First Serve



Shortest Seek Time First:



Circular scan:

