Programmeringsuppgift, ver 1.2

Vi föredrar om uppgiften görs i Java men det är helt okej att göra uppgiften i det språk du känner dig mest bekväm i. Om du väljer Java skall uppgiften göras i en av Javas LTS versioner (8, 11 eller 17). Programmet skall vara exekverbart när du lämnat in det.

Oavsett språk så tänk objektorientering och generalisering. Man kanske vill ha flera, nya validatorer. Man vill kunna sätta ihop godtyckliga sekvenser av validatorer.
Det är tillåtet att använda färdiga klassbibliotek

Validitetskontroll

ValidityChecker

Skriv en ValidityChecker vilken skall kunna konfigureras med olika ValidityChecks att utföras på data i någon ordning. Resultatet av en validitetskontroll m.h.a. ValidityChecker skall vara att indatat passerar validitetskontrollen eller ej. Dessutom skall den validitetskontroll som inte passeras loggas.

ValidityCheck

En ValidityCheck är en specifik validitetskontroll. Dessa ValidityChecks skall vara atomära och oberoende av varandra kunna kombineras i en ValidityChecker. Exempel på ValidityChecks är exempelvis; ”NotNull”, ”NotEmpty” och komplexare så som ”ÄrPersonnummer” etc.

1. Skriv en ValidityCheck som kontrollerar att data inte är null.
2. Skriv en ValidityCheck som kontrollerar huruvida datat representerar ett personnummer eller ej.
3. Skriv en ValidityCheck som kontrollerar om datat representerar ett ”normalt” svenskt bil registreringsnummer.

Personnummeralgoritmen

Algoritmen för att räkna ut ett personnummers kontrollsiffra är som följer:

Exempel 1

Pnr 19780202-2389										
19	**7**	**8**	**0**	**2**	**0**	**2**	**2**	**3**	**8**	**9**
	* 2	* 1	* 2	* 1	* 2	* 1	* 2	* 1	* 2	
	14	**8**	**0**	**2**	**0**	**2**	**4**	**3**	**16**	
	1 + 4	8	0	2	0	2	4	3	1 + 6	
	5	**8**	**0**	**2**	**0**	**2**	**4**	**3**	**7**	
	5 +	8 +	0 +	2 +	0 +	2 +	4 +	3 +	7	
									31	
									mod 10	
									1	
									10 - 1	
									9	
									mod 10	
									9	**9**

Exempel 2

Pnr 19820411-2380										
19	**8**	**2**	**0**	**4**	**1**	**1**	**2**	**3**	**8**	**0**
	* 2	* 1	* 2	* 1	* 2	* 1	* 2	* 1	* 2	
	16	**2**	**0**	**4**	**2**	**1**	**4**	**3**	**16**	
	1 + 6	2	0	4	2	1	4	3	1 + 6	
	7	**2**	**0**	**4**	**2**	**1**	**4**	**3**	**7**	
	7 +	2 +	0 +	4 +	2 +	1 +	4 +	3 +	7	
									30	
									mod 10	
									0	
									10 - 0	
									10	
									mod 10	
									0	**0**