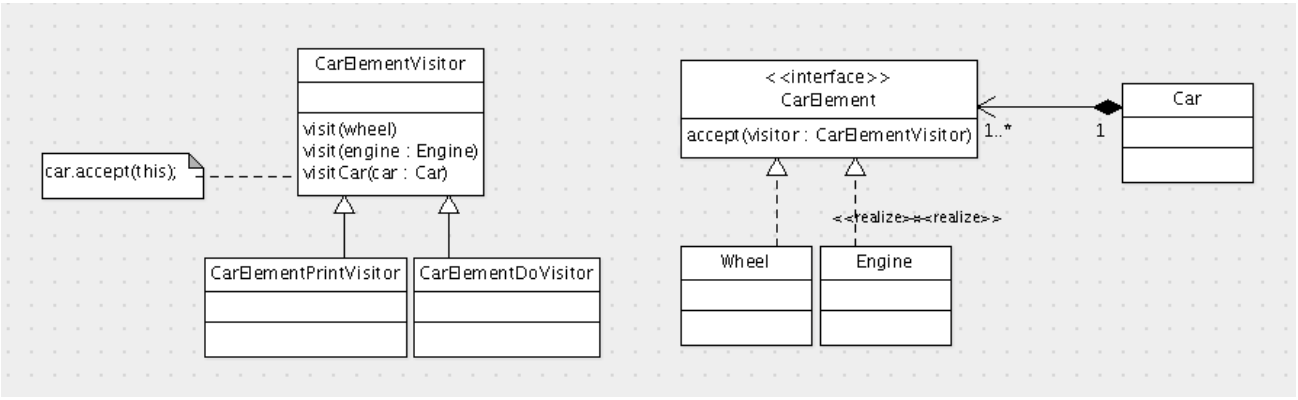
Dizajn programskih jezika - januar 2009.

1. Napisati JAVA kôd na osnovu UML dijagrama:



Funkcija `main` treba da izgleda kao:

```
public static void main(String[] args) {
   Car auto = new Car();
   CarElementVisitor printVisitor = new CarElementPrintVisitor();
   CarElementVisitor doVisitor = new CarElementDoVisitor();
   printVisitor.visit(car);
   doVisitor.visit(car);
}
```

Izlaz iz programa je:

```
Visiting: Engine
Visiting: Wheel 1
Visiting: Wheel 2
Visiting: Wheel 3
Visiting: Wheel 4
Starting engine...
Rotating the wheel...
Rotating the wheel...
Rotating the wheel...
Rotating the wheel...
```

2. U jeziku SCHEME definisati funkciju `(power_n a n)` logaritamske složenosti koja izračunava stepen broja a^n.

3. U jeziku HASKELL definisati tip podataka `Stablo a` za predstavljanje binarnog stabla čiji su elementi tipa `a` i funkciju

   `zipTreeWith :: (a -> b -> c) -> Stablo a -> Stablo b -> Stablo c`

   koja na dva stabla istog oblika primenjuje dati operator element po element i tako gradi novo stablo istog oblika (funkcija je analogna funkciji `zipWith` za liste).

4. Lista je uspravnim crtama podeljene na delove. Npr. `1 2 |3 4 5 |6 7 |8`. Ovakvu strukturu podataka je moguće predstaviti na sledeća dva načina:

```
((1, false) (2, false) (3, true) (4, false) (5, false) (6, true) (7, false) (8, true))

((1 2) (3 4 5) (6 7) (8))
```

   Napraviti funkciju koja prevodi iz prve u drugu (ili iz druge u prvu) reprezentaciju.

5. Korišćenjem tehnike lenjog izračunavanja dati efikasnu definiciju liste brojeva koji odgovaraju vrednostima funkcije definisane sa

```
f n | n == 0 = 1
    | n == 1 = 2
    | otherwise = 2*f(n-1) + 3*f(n-2)
```

   Uputstvo: Fibonačijevi brojevi se mogu definisati kao:

```
fibs = 0 : 1 : zipWith (+) fibs (tail fibs)
```