

Es soll eine Protokolldatei einer Autofahrt analysiert werden und die maximale Geschwindigkeit während der Fahrt berechnet werden.

Beispiel einer Protokolldatei:

Anfahren Beschleunigen 10 Abbremsen 5 Beschleunigen 10
Beschleunigen 5 Abbremsen 10 Stehenbleiben

Vorgehen:

- 1) Grammatik der Protokolldatei in EBNF beschreiben
Autofahrt = "Anfahren" Fahren "Stehenbleiben".
Fahren = { ("Beschleunigen" | "Abbremsen") number }.
- 2) Schreiben des Parsers (nur Textanalyse)
- 3) "Attributieren" des Parsers
 - Einfügen von Berechnungen
 - Aufbau von Datenstrukturen (z.B. "Syntaxbäume")
 - etc.

Autofahrt = "Anfahren" Fahren "Stehenbleiben".

Fahren = { ("Beschleunigen" | "Abbremsen") number }.

```
public class AutofahrtParser {  
    // parses "Autofahrt"  
    public static void parseAutofahrt() {  
        String ident = In.readIdentifier();  
        if (!ident.equals("Anfahren"))  
            SyntaxError("'Anfahren' expected, '" +  
                ident + "' found.");  
        parseFahren();  
        ident = In.readIdentifier();  
        if (!ident.equals("Stehenbleiben"))  
            SyntaxError("'Stehenbleiben' expected, '" +  
                ident + "' found.");  
    } // Parse done  
    ...  
}
```

Autofahrt = "Anfahren" Fahren "Stehenbleiben".

Fahren = { ("Beschleunigen" | "Abbremsen") number }.

```
public class AutofahrtParser {  
    // parses "Autofahrt" and calculates maximum speed  
    public static int parseAutofahrt() {  
        String ident = In.readIdentifier();  
        if (!ident.equals("Anfahren"))  
            SyntaxError("'Anfahren' expected, '" +  
                ident + "' found.");  
        int maxSpeed = parseFahren();  
        ident = In.readIdentifier();  
        if (!ident.equals("Stehenbleiben"))  
            SyntaxError("'Stehenbleiben' expected, '" +  
                ident + "' found.");  
        return maxSpeed;  
    } // Parse done  
    ...  
}
```

Autofahrt = "Anfahren" Fahren "Stehenbleiben".

Fahren = { ("Beschleunigen" | "Abbremsen") number }.

```
private static void parseFahren() {  
    while (In.peek() == 'B' || In.peek() == 'A') {  
        String ident = In.readIdentifier();  
        if (ident.equals("Beschleunigen"))  
            ;  
        else if (ident.equals("Abbremsen"))  
            ;  
        else SyntaxError("Illegal ident " + ident);  
        In.readInt();  
    } // while  
} // parseFahren  
} // class AutofahrtParser
```

Autofahrt = "Anfahren" Fahren "Stehenbleiben".

Fahren = { ("Beschleunigen" | "Abbremsen") number }.

```
private static int parseFahren() {
    int curSpeed = 0, dir, maxSpeed = 0;
    while (In.peek() == 'B' || In.peek() == 'A') {
        String ident = In.readIdentifier();
        if (ident.equals("Beschleunigen"))
            dir = 1;
        else if (ident.equals("Abbremsen"))
            dir = -1;
        else SyntaxError("Illegal ident " + ident);
        curSpeed = curSpeed + dir * In.readInt();
        if (curSpeed > maxSpeed)
            maxSpeed = curSpeed;
    } // while
    return maxSpeed;
} // parseFahren
} // class AutofahrtParser
```