# Übungen Compilerphasen

## Übung 1

Welche Compilephasen gibt es?

## Übung 2

Was ist der Input und Output der lexikalischen Analyse?

## Übung 3

Ordnen sie die folgenden Fehlermeldungen einer Compilephase zu

* „Variable myVar wurde bereits definiert“
* „‘)‘ missing“
* „unterminated string literal“
* „Expression of type int cannot be assigned to variable to type string”

## Übung 4

Welche Vorteile bietet die Trennung der Codegenerierung in Zwischencodegenerierung und Maschinencodegenerierung?

## Lösung 1

Lexikalische Analyse

Syntaktische Analyse

Semantische Analyse (Symboltabelle, Typeinference)

Zwischencodegenerierung

Optimierung

Maschinencodegenerierung

## Lösung 2

Input: Zeichenstrom  
Output: Token

## Lösung 3

* Semantische Analyse (Symboltabelle)
* Syntaktische Analyse
* Lexikalische Analyse
* Semantische Analyse (Typeinference)

## Lösung 4

* Der Zwischencode eignet sich gut für Optimierungen
* Aus dem Zwischencode kann Maschinencode für verschiedene Hardwarearchitekturen erzeugt werden

Lösung 2

FIRST(PC1) = {“c“}  
FIRST(PC2) = {ɛ}  
FIRST(C) = FIRST(PC1) u FIRST(PC2) = {“c“ , ɛ}  
FIRST(PB1) = {“b“}  
FIRST(PB2) = {ɛ}  
FIRST(B) = FIRST(PB1) u FIRST(PB2) = {“b“, ɛ}  
FIRST(PA1) = FIRST(B)/{ɛ} u FIRST(C) = {“b“, “c“, ɛ}  
FIRST(PA2) = {“a“}  
FIRST(A) = FIRST(PA1) u FIRST(PA2) = {“a“, “b“, “c“, ɛ}

da A das Startsymbol ist   
FOLLOW(A) = {#}   
  
da C nur am Ende einer Produktion von A vorkommt   
FOLLOW(C) = FOLLOW(A) = {#}   
  
da FIRST(C) ɛ enthält  
FOLLOW(B) = FIRST(C) u FOLLOW(C) = {“c“, #}  
  
Die First Sets von PC1, PB1 und PA2 enthalten nicht ɛ und sind daher identisch mit den Selection Sets.  
SELECT(PA1) = FIRST(PA1) /{ɛ} u FOLLOW(A) = {“b“, “c“, #}  
SELECT(PB2) = FIRST(PB2)/{ɛ} u FOLLOW(B) = {“c“, #}  
SELECT(PC2) = FIRST(PC2)/{ɛ} u FOLLOW(C) = {#}  
  
Die Selection Sets  
PA1, PA2  
PB1, PB2  
PC1, PC2  
sind jeweils disjunkt, daher ist die Grammatik LL(1).