Sprache B

Die Sprache B besteht aus Aussagen, wie sie unten beschrieben und erklärt werden. Diese Aussagen sind aus Schlüsselwörtern und den Namen der Individuen und Typen der entsprechenden Mini-Welt aufgebaut. Die verwendeten Schlüsselwörter sind „HasType“, „SubTypeOf“, „DisjointWith“, „EquivalentTo“, „not“, „and“ und „or“.

Aussagen

Jede Aussage kann entweder wahr oder falsch sein. Jede Aussage der Sprache B hat die Form eines der vier Schemas, die hier beschrieben werden. Es ist zu beachten, dass Typen komplex sein können (siehe nächster Abschnitt).

HasType-Aussagen	
Schema:	*Individuum* ***HasType*** *Typ*
Beispiel:	John **HasType** golfer
Erklärung:	Eine HasType-Aussage verlangt ein Individuum und einen Typ und sagt aus, dass das gegebene Individuum zum gegebenen Typ gehört. Das obige Beispiel sagt aus, dass John ein Golfer ist.

SubTypeOf-Aussagen	
Schema:	*Typ1* **SubTypeOf** *Typ2*
Beispiel:	golfer **SubTypeOf** man
Erklärung:	Eine SubTypeOf-Aussage verlangt zwei Typen und sagt aus, dass jedes Individuum, das zum ersten Typ gehört, auch zum zweiten Typ gehört (aber nicht zwingenderweise umgekehrt). Das obige Beispiel sagt aus, dass jedes Individuum, das ein Golfer ist, auch ein Mann ist.

DisjointWith-Aussagen	
Schema:	*Typ1* **DisjointWith** *Typ2*
Beispiel:	woman **DisjointWith** golfer
Erklärung:	Eine DisjointWith-Aussage verlangt zwei Typen und sagt aus, dass kein Individuum sowohl zum ersten Typ als auch zum zweiten Typ gehört. Das obige Beispiel sagt aus, dass kein Individuum sowohl eine Frau als auch ein Golfer ist.

EquivalentTo-Aussagen	
Schema:	*Typ1* **EquivalentTo** *Typ2*
Beispiel:	golfer **EquivalentTo** man
Erklärung:	Eine EquivalentTo-Aussage verlangt zwei Typen und sagt aus, dass jedes Individuum, das zum ersten Typ gehört, auch zum zweiten Typ gehört und umgekehrt. Das obige Beispiel sagt aus, dass jedes Individuum, das ein Golfer ist, auch ein Mann ist und umgekehrt.

Typ-Operatoren

Jeder Typ (einfach oder komplex) steht für eine gewisse Gruppe von Individuen. Einfache Typen sind zum Beispiel „woman“ oder „golfer“. Neben einfachen Typen gibt es aber auch komplexe Typen, die durch die hier beschriebenen Typ-Operatoren zusammengesetzt werden. „woman **or** golfer“ ist zum Beispiel ein komplexer Typ. Es ist zu beachten, dass solche komplexe Typen auch ineinander verschachtelt sein können. In diesem Fall werden Klammern verwendet um die Struktur zu verdeutlichen, zum Beispiel „**not** (golfer **and** man)“.

not-Operator	
Schema:	**not** *Typ*
Beispiel:	**not** golfer
Erklärung:	Der not-Operator verlangt nur einen Typ. Der entstehende komplexe Typ steht für alle Individuen, die nicht zum gegebenen Typ gehören. Das obige Beispiel steht für alle Individuen, die keine Golfer sind.

and-Operator	
Schema:	*Typ1* **and** *Typ2*
Beispiel:	golfer **and** man
Erklärung:	Der and-Operator verlangt zwei Typen. Der entstehende komplexe Typ steht für alle Individuen, die sowohl zum ersten Typ als auch zum zweiten Typ gehören. Das obige Beispiel steht für alle Individuen, die sowohl Golfer als auch Männer sind.

or-Operator	
Schema:	*Typ1* **or** *Typ2*
Beispiel:	woman **or** golfer
Erklärung:	Der or-Operator verlangt zwei Typen. Der entstehende komplexe Typ steht für alle Individuen, die zum ersten Typ oder zum zweiten Typ oder zu beiden gehören. Das obige Beispiel steht für alle Individuen, die Frauen oder Golfer oder beides sind.