1 Auslesen der Konstituentenstruktur (4 + 8 + 8 Punkte)

a) Der in der Vorlesung vorgestellte Chartparser steht unter `http://www.dfki.de/~becker/lispkurs/vorlesungen.ss00/chart.lisp` Analysiere den Satz "the student saw the man with the scope." Wieviele Lesarten gibt es? Erkläre genau, welche Kanten in der chart welche Lesart(en) repräsentieren.

b) Die Funktion `edge2trees` aus `http://www.dfki.de/~becker/lispkurs/vorlesungen.ss00/edge2trees.lisp` soll (passive) Kanten aus der chart in die zugehörigen Konstituentenstrukturen (Bäume in Listennotation) übersetzen. Schreibe die fehlende Funktion `find-edges` entsprechend des angegebenen Funktionsprototyps; die Funktion soll abhängig von ihren Aufrufargumenten alle Kanten einer chart auswählen, die die angegebenen Eigenschaften haben. Wird ein Argument nicht übergeben, kann das korrespondierende Merkmal der Kanten einen beliebigen Wert haben.

   Hinweis: Die Funktion muß unterscheiden können, ob ein Argument (möglicherweise mit dem Wert nil) angegeben wurde, oder im Funktionsaufruf gar nicht vorkam. Lies die genaue Syntax der `&key` Parameter in den Folienkopien des Einführungskurses nach.

c) In der Funktion `complete` fehlen einige Teile (markiert als [...]); ergänze sie so, daß `complete` gegeben eine Liste von Konstituentenkategorien (z.B. '(NP VP)') die Menge (Liste) aller möglichen Folgen von passiven Kanten (als Listen) berechnet, die adjazent zueinander der gesuchten Kategorienfolge entsprechen und die angegebenen Positionen und Abdeckungen haben.

   Hinweis: Die Funkion ist in ihrer Grundstruktur der Hilfsfunktion `cross-product` sehr ähnlich.

2 Schmökaufgabe: Laufzeit und Speicherverbrauch (5 Punkte)

a) Messungen mit time zeigen, daß der chart Parser im Laufzeitverhalten in etwa dem vorgestellten bottom-up Parser mit Gedächtnis entspricht. Beim Vergleich des Speicherbedarfs verbraucht der aktive chart Parser ('chart.lisp') allerdings deutlich mehr Hauptspeicher.

   Verwende `trace` auf Funktionsaufrufe, von denen zu vermuten ist, daß sie Speicher anfordern (d.h. neue Objekte erzeugen), und vergleiche das Ergebnis mit den Objekten, die nach Abschluß einer Analyse in der chart stehen. Was genau ist das Hauptproblem des aktuellen Verfahrens? Wann werden in großer Zahl Objekte erzeugt, die gar nicht zum Ergebnis beitragen?

   Hinweis: Für die Bearbeitung dieser Aufgabe reicht es, die Verarbeitung eines sehr kurzen Satzes zu studieren; das Problem ist so grundsätzlich, daß es (fast) sofort seine Auswirkung zeigt.

b) Modifiziere die Funktionen in 'chart.lisp' so, daß deutlich weniger unnötige Objekte erzeugt werden. Mit zwei vergleichsweise kleinen Änderungen läßt sich eine signifikante Verringerung des Speicherverbrauchs erzielen.