Understanding measure and comparative sentences through Prolog

Bengt Sigurd

Measure sentences are e.g."Sven is 12 years old",Swedish: "Sven är 12 år gammal","The boat is 12 feet long", Swedish: "Båten är 12 fot lång", "The stone has a weight of 12 kilos", Swedish:"Stenen har en vikt av 12 kilo", "How much is the car?", Swedish:"Hur mycket kostar bilen?". Such sentences are part of the core grammar and play an important role in the use of natural language in communication. They have attracted the interest of linguists because they indicate how languages express fundamental cognitive dimensions and many linguists have noted that "old" in "How old is Sven?" does not presuppose that Sven is old, while "How young is Sven?" presupposes that Sven is young. It has also been noted that the sentence "Sven is old " means that Sven is old for a man and that even the absolute use of adjectives implies a comparison with a standard. Comparative constructions such as "Sven is five years older than Lisa " have added to the interest in measure adjectives. There is a large body of literature on adjectives and comparatives(cf.Rusiecki,1985). These works will not be reviewed in this paper, which aims at illustrating how such facts appear in a computer program which can "understand" measure and comparative sentences and answer corresponding questions.

This paper will present the basic syntactic and semantic features of Swedish measure and comparative sentences. The Swedish expressions will be translated and comparison with the equivalent English expressions will thus be possible. The computer program, written in Prolog, is constructed with this background. Various limitations have been introduced in order to get a program which can be used for demonstration purposes.

The program can parse a wide range of measure and comparative sentences. If it is asked to accept a sentence it adds the fact to its data base and returns "Jaha" (OK). If it is asked to answer a question it returns an answer if one is available.The program knows the equivalence of sentences such as "The boat is 30 feet long","The boat has a length of 30 feet","The length of the boat is 30 feet" and "The boat measures 30 feet" and is able to use accepted information in answers to several types of questions. It understands that it is reasonable to call a person who is over 60 old. The program understands that Sven is older than Bill if it has learnt that Sven is older than Lisa and Lisa is older than Bill. It understands that Lisa is 5 years younger than Sven if it has learnt that Sven is 5 years older than Lisa. It understands that Sven is double as old as Lisa if it has learnt that Lisa is half as old as Sven. In some variants of the program explanations of the conclusions are also added.

The program may be of interest to those who construct expert systems, where variants of measure sentences often play an important role. The fragment discussed can perhaps be embedded in a more comprehensive system. Some theoretical and psycholinguistic issues will be touched upon as the program forces the designer to decide where different types of linguistic and encyclopedic knowledge is to be put. The choice of logical representations is also an interesting theoretical issue. Several have to be chosen in order to handle the problems and pecularities discussed above. One might suggest that e.g. only the unmarked representation is stored in the memory ("Sven is bigger than Lisa" not "Lisa is smaller than Sven") and that inferences always, or preferably, are done on unmarked (positive) representations. This idea can easily be implemented in a version of the program to get a model which can explain why subjects can verbalize the unmarked adjective faster than the marked one(big before small) as has been found (Schriefers,1985).

The differences between Swedish and English evoke some comments and raise questions about measure and comparative sentences in the languages of the world - typological questions which will not be treated in this paper.

The syntax of measure sentences

Four main types of measure sentences can be distinguished: The adjectival type (Sven är fem år gammal),the nominal type (Sven har en ålder av fem år),the genitive type (Svens ålder är fem år) and the verbal type (Sven väger trettio kilo;lacking for the dimension age).

The syntactic structure of the first type can be illustrated by the following representation:

np(Sven),cop(är),pf(mf(num(fem),unit(år)), a(gammal)), where pf means predicative phrase, mf means measure phrase, a means (measure) adjective, and the other abbreviations should be evident . The characteristic feature is the copula and the adjective and in the Prolog program only the present tense "är" (is) and some common adjectives are used. The sample subject nouns are chosen not to create any variation in agreement.If e.g."barnet"(the child) is also included the ending -t would have to be added to the predicative adjective as "barnet " has neuter gender.("Barnet är tre år gammalt").It is quite possible to handle such problems in the Prolog definite clause grammar formalism used, but for our purpose it is important not to clutter up the program by the additional machinery.

One equivalent question is:

pf(mf(hur), a(gammal)), cop(är),np(sven)?(How old is Sven?), and an answer could be:"(Sven är) fem år (gammal)." It would also be possible to answer such a question without a measure phrase by saying:"Han är mycket gammal."(He is very old.) This case is not covered by our program. Another equivalent question is:

pf(mf(num(hur många),unit(år)), a(gammal)),cop(är),np(sven), but the only acceptable answer in that case is a numerical answer substituting for "hur_många"(how many), e.g."(Sven är) fem(år gammal)". The usual way of putting this quesions is in fact only:"Hur många år är Sven?" where "gammal" is deleted, probably as being redundant when "år"(years) is mentioned.

Sentences without measure phrases are similar to this type. But as many linguists have noted the meaning of some measure adjectives is quite different if used with a measure phrase. "Sven is five years old" does not mean that he is old, and that is why it is not too strange to continue:"He is not old". For basic dimensions there is an adjective which can be used in this neutral way;an exception is the dimension weight. As in English one cannot say:"Sven är fem kilo tung, nor "fem kilo lätt."(Sven is five kilos heavy/light). One has to say that:"Sven väger mycket"(Sven weighs a lot) or "Svens vikt är hög"(Sven's weight is great). "Sven är fem kilo tung" is strange, but it has an interpretation, which we might want our program to be able to handle.(A version does).

The second type of measure sentences is the nominal type, which may be illustrated by:

np(Sven),vp(har),np(en,mn(ålder),av,mf(fem,år))

where the characteristic feature is "har" plus the measure noun (mn) followed by a measure phrase.

The equivalent question is "Vilken ålder har Sven?" (What age is Sven?). This construction is slightly formal.

The third type of measure sentence is characterized by a subject where the bearer of the property is in the genitive and the property is denoted by a measure noun. It can be illustrated by:

np(Svens,mn(ålder)), cop(är),mf(fem,år). The equivalent question is "Vilken är Svens ålder?" It is not possible to say "Vilken ålder är Svens?" which would rather be an equivalent to the strange English sentence "Which age belongs to Sven?". It is, however, also possible to say (more informally) "Vad är Svens ålder?" A native is hardly aware of which kind of measure construction he uses.

There is a related nominal construction:"Sven har hög ålder"(Sven has high age) and a related genitive construction:"Svens ålder är hög"(Sven's age is high). We note that hög(high),låg(low) can be used with some dimensions e.g.ålder,kvalitet,kvantitet,höjd,hastighet(speed).

The fourth type of measure construction is characterized by the fact that the verb denotes the dimension, but there are only a few such verbs available. The construction can be illustrated by:

np(Bilen),vp(kostar),mf(8000,kronor)

The verb "kostar"(cost) denotes the price dimension, "väger" denotes the weight dimension (not "tynger", which means press down)."Rymmer","innehåller"(contains) denote the volume dimension, but other measure verbs are hard to come by. Some that come to mind e.g."mäter"(measure),"räknar"(count) seem often to have less acceptability and they are not associated with only one dimension. It seems possible to answer the question "Hur stor är salen"(How big is the hall?) by "Den mäter 1000 kvadratmeter", but also by e.g. "Den räknar 1000 stolar". It is also possible to take "stor" to indicate volume and answer "Den rymmer 500 personer".(It can take 500 persons, It can seat 500 persons). The verb "omfattar" would fit in a sentence such as "Den omfattar 10 sektioner"(It includes 10 sections). It is possible to answer the question "Hur lång är båten?" (How long is the boat?" by "Den är 30 fot lång", but also by "Den mäter 30 fot". The question "Hur bred är båten?" (How wide is the boat?" can possibly also be answered by a phrase containing the verb "mäter" e.g. "Den mäter två meter" (It measures two meters). The English verb measure seems to carry similar stylistic values.

There is a related construction without a measure phrase:"Bilen kostar mycket" (The car costs much), where the word "mycket" is an equivalent of "a lot". It is sometimes possible to use the verb alone (in a pregnant sense):"Det kostar", meaning "That costs a lot". In the same way some dimensional nouns can be used in a pregnant sense, e.g."Sven har tyngd" (Sven has weight), but in that case the meaning is abstract:Sven is of importance/carries weight.

The price dimension can also be denoted by "betingar". It may also combine with the noun:"Bilen betingar ett högt pris /5000 kronor"(The car costs a lot/5000 kronor).

Comparative sentences

Superlative sentences cannot normally take measure phrases as illustrated by the following sentences and ungrammatical expansions: "Sven är äldst av pojkarna"(Sven is (the) oldest of the boys), *"Sven är 5 år äldst av pojkarna"(Sven is 5 years (the) oldest of the boys). It is, however, possible to use a non-numerical modifier as e.g in "Sven är klart äldst av pojkarna"(Sven is clearly/by far the oldest of the boys). Superlative sentences are not included in the fragment of Swedish covered by our program, but it is interesting to speculate about the reasons why natural language(s) - at least not Swedish and English - do not offer superlatives for numerical quantification of the difference between an object and the group of objects it is singled out from on the basis of a certain dimension. If Sven is 5 years older than a certain group of boys this fact is rendered by a comparative construction where he is not considered to be a member of a group:"Sven är 5 år äldre än pojkarna"(Sven is 5 years older than the boys).

There are different types of comparative construction. Some types will be discussed below, but all are not covered by the Prolog program. We may distinguish between adjectival types, where the comparative forms discussed by so many linguists appear, e.g. "Sven är 5 kilo tyngre än lisa" (Sven is 5 kilos heavier than Lisa), verbal types, e.g. "Sven väger 5 kilo mer än Lisa" (Sven weighs 5 kilos more than Lisa), and nominal types e.g. "Svens vikt är 5 kilo högre än Lisas" (Sven's weight is 5 kilos greater than Lisa's), "Sven har en fem kilo högre vikt än Lisa" (Sven has a 5 kilos greater weight than Lisa).

Comparative sentences may also be divided into equality (comparative) sentences e.g. "Sven är lika tung som Per" (Sven is equally heavy as Per, Sven is as heavy as Per) and difference (comparative) sentences. Difference sentences may be divided into additional (and subtractional) sentences, e.g. "Sven är 5 kilo tyngre än Lisa" (Sven is 5 kilos heavier than Lisa) and multiplicational (divisional) sentences, e.g. "Sven är två gånger så gammal som Tim, Sven är dubbelt så gammal som Tim,Sven är två gånger (dubbelt) äldre än Tim" (Sven is two times (twice) as old as Tim). The divisional comparative type is illustrated by "Tim är hälften så gammal som Sven" (Tim is half as old as Sven). It is also possible to combine these types and verbalise an euqation as in "Sven är dubbelt så gammalsom Rut, minus 2 år" (Sven is double as old as Rut, minus 2 years), but such complex constructions will not be handled by our demonstration program.

The syntactic structure of an evaluative comparative construction without a measure phrase quantifying the difference can be illustrated by the following sentence.

```
np(Sven),cop(är),   compar(äldre),pp( prep(än),np(Lisa))
   Sven      is            older          than    Lisa
```

Equivalent questions are e.g. "Är Sven äldre än Lisa?" (Is Sven older than Lisa?), "Vem är äldre än Lisa?" (Who is older than Lisa?).

The syntactic structure of a related construction where a measure phrase is included can be illustrated by the following sentence.

```
np(Sven),cop(är),mf(num(5),unit(år)), compar(äldre)
   Sven      is         5     years          older
   pp(prep(än) np(Lisa))
          than    Lisa
```

There are several ways of asking equivalent questions depending on how many words are moved to the front of the sentence:"Hur många år är Sven äldre än Lisa?"(How many years is Sven older than Lisa?), "Hur många år äldre är Sven än Lisa?" (How many years older is Sven than Lisa?), "Hur många år äldre än Lisa är Sven?" (How many years older than Lisa is Sven?). English seems to have the same stylistic options. The unit(år) has to go with the question word (hur många), which supports the measure phrase (mf) as a constituent. The preposition(än) has to go with the noun(Lisa), which supports the prespoitional phrase as a constituent. But the comparative(äldre) can be moved with the measure phrase or be stranded with the prepositional phrase and this fact speaks against assuming a constituent including the comparative form and the prepositional phrase.

Multiplicational comparative constructions can be given a syntactic representation as illustrated by the following sentence.

```
np(Sven),cop(är),gf(num(5),gånger),adj(äldre)
   Sven      is          5  times       older
   pp(prep(än) np(Tim))
          than    Tim
```

With dubbelt (double) the construction illustrated by "Sven är dubbelt så gammal som Tim" is generally used, but in other cases both the construction with the comparative form and the construction "så adj som" are grammatical. This paper does not attempt to pinpoint all such idiosyncratic facts, however.

Equivalent questions support the constituent analysis suggested above. One may ask:"Hur många gånger är Sven äldre än Tim?"(How many times is Sven older than Tim?), "Hur många gånger äldre är Sven än Tim?"(How many times older is Sven than Tim?), "Hur många gånger äldre än Tim är Sven?"(How many times older than Tim is Sven?). An answer could be:"Dubbelt så gammal" (Double as old), "Tre gånger så gammal" (Three times as old) or "Tre gånger äldre" (Three times older). The wording of the answer is not dependent on the wording of the question, only on the meaning of the question.

Synonymy relations

As noted above there are often (almost) equivalent adjectival, nominal and verbal expressions. The following table lists the most important cases, where there are at least two word classes to choose from when a dimension is to be denoted together with a measure phrase.

Verb	Adjective	Noun
-	gammal(old)	ålder(age)
mäter(measures)	lång(long)	längd(length)
rymmer(contains)	stor(big)	volym(volume)
räknar(counts)	stor(big)	storlek(size)
väger(weighs)	-	vikt(weight)
-	hög(high)	höjd(height)
-	djup(deep)	djup(depth)
-	bred(wide)	bredd(depth)
-	tjock(thick)	tjocklek(thickness)
-	varm(warm)	temperatur
kostar(costs)	-	kostnad,pris(cost)
betingar(costs)	-	kostnad,pris(cost)

There are very few verbs which have a dimensional meaning and can be combined with a measure phrase. And some are ambiguous as noted. There exist only a few adjectives which can occur with a measure phrase and have a dimensional meaning in ordinary language. Dimensional nouns can always be formed, especially in more specific or scientific language. We might say that an object has flexibility 5, the colour resistence 3 etc.This is done in consumers tests. But even if scalar dimensions are thus devised one can only use the noun, not the equivalent adjective. One may say that something has a flexibility of 5 flex units, but hardly that it is 5 (flex) units flexible or that it flexes 5 units.

Logical representations

The logical representation of measure sentences with measure phrases is l(X,D,N,U) or l(X,D,E) , where X,D,E,N,U are variables. X is the object having the property. D is e.g.age, the name of the property(dimension). E indicates an evaluation and takes on the value more(+),indicating more than the standard for such objects or less(-), indicating less than the standard for such objects. The variable N covers numbers and the variable U covers units of different type. e.g. year, meter. A typical sentence with a measure phrase e.g. Sven is 5 years old would be represented by l(sven,age ,5,years). Time is disregarded.
The sentence Sven is old would be represented by the formula l(sven,age,more),where the evaluation variable is set at more meaning that Sven has a high age (but the number and unit variables are unknown). Different formats are thus used for the two types of sentences, and normally evaluation is incompatible with a specification of number and units. It is sometimes possible,although a bit strange,to give both an evaluation and a number of units and say e.g. Sven is fifty years young, indicating that Sven is young for his age and such cases can be handled if two facts are added to the data base, which can be done in various ways. But if this way of construction is permitted and considered as grammatical the grammar will be overgenerating.

The following table indicates how some common sentences will be represented in our logical format.

Sentences	Logical form
Sven är 2 meter lång.	l(sven,height,2,meter)
Sven är 50 år gammal.	l(sven,age,50,år)
Sven väger 70 kilo.	l(sven,weight,70,kilo)
Bilen kostar 5000 kronor.	l(bilen,cost,5000,kronor)
Lisa har en ålder av 40 år.	l(lisa,age,40,år)
Bilen är hög.	l(bilen,height,more)
Bilen är lätt.	l(bilen,weight,less)
Per är 70 år ung.	l(per,age,70,år),l(per,age,less) (in some versions of program)
Pers ålder är 70 år.	l(per,age,70,år)

Certain questions ask for numerical details, e.g."Hur många år(gammal) är Sven?" and they have to answered by calling the longer logical form and giving the values of the last two variables. Even the question "Hur gammal är Sven?" has to be answered by giving the values of the last two variables. The typical cases when the evaluation variable is to be looked for are yes/no questions such as "Är Sven gammal?". An answer could be "Ja" and if numerical values are available the answer could be expanded into "Ja, han är 75 år".(Not done in this program).Other questions of this type are:"Är bilen dyr?" (Is the car expensive?), "Kostar bilen mycket?" (Is the car expensive?).

Logical representations of comparative sentences are established similarly. We will represent "Sven är äldre än Lisa" by 1(sven,age,more,lisa) or more generally 1(X,D,E,Y), where X and Y are the two objects compared (X is in focus), D is the dimension of comparison, E is the evaluative variable, which may be more(with age representing older), or less (with age representing younger). This representation could,in fact, also be used for absolute cases, such as "Sven är gammal"(Sven is old), if we assign a standard to Y.

For the sentence "X är lika gammal som Y" (X is as old as Y) the representation 1(X,age,more,like,Y) is used and for the sentence "X är lika ung som Y" (X is as young as Y) the representation 1(X,age,less,like,Y) is used. The difference between those two sentences is that with "ung"(young) the persons are pressupposed to be young, while with "gammal" (old) no evaluation is implied (gammal being the neutral unmarked term). Maybe this presupposition should rather be represented as a separate fact, however.

The quantified comparative sentences need more complex representations. For the additional case illustrated by "Sven är 5 år äldre än Lisa" (Sven is 5 years older than Lisa) we write 1(sven,age,more,5,years,lisa), where the forth and fifth arguments are the numbers and the units, respectively. Changing more into less allows us to represent the antonyms (older:younger,heavier:lighter,etc.). The case where the difference is given in multiplicational terms is illustrated by the following case:
1(sven,age,more,5,times,lisa), which is the logical translation of "Sven är 5 gånger så gammal som/äldre än Tim" (Sven is 5 times as old as/older than Tim).

The logical representations used are ad hoc and it is not clear how the should be integrated in a full semantics.

Some implications and inferences

Speakers of English draw the conclusion that Sven weighs 50 kilos, that his weight is 50 kilos and that he has a weight of 50 kilos if they have learnt that he weighs 50 kilos. They do not, in fact, consider these conclusions as consclusions rather as synonymy relations. This is a reason for treating them in the way it is done in our program, where all these expressions are represented by (translated into) the same logical representation. Once one of the expressions has been mentioned the others are automatically true and questions based on the other formulations are answered in the affirmative. The choice between adjectival, nominal or verbal expression does not involve any logical inferences in our program. The borderline between verbal synonymy and logical inferences is of theoretical interest, but it will not be discussed in this paper.

A genuine inference may be illustrated by deriving "Lisa is younger than Sven" from "Sven is older than Lisa". We may call this inference the "antonym comparative inference". In our Prolog program such inferences are handled by variants of the following rules.

```
l(X,D,more,Y) :- l(Y,D,less,X)
l(X,D,less,Y) :- l(Y,D,more,X)
```

Similarly the inference Lisa is 5 years younger than Sven can be drawn from the fact that Sven is 5 years older than Lisa, if the following rules are included in the program.

```
l(X,D,more,N,U,Y) :- l(Y,D,less,N,U,X)
l(X,D,less,N,U,Y) :- l(Y,D,more,N,U,X)
```

These implications and inferences are handled in a special section of the program which can be extended further. Presently this section also includes an equivalence inference

```
l(X,D,more,like,Y) :- l(Y,D,more,like,X)
```

This rule states that Sven is as tall as Tim if Tim is as tall as Sven. If we want to cover that Sven is as small as Tim if it is known that Tim is as big as Sven we could add a rule with less instead of more. A special variable ranging over more,less could also be used.

Multiplicational inferences can be handled by the following rule.

```
l(X,D,more,1/N,times,Y) :- l(Y,D,more,N,times,X).
```

This inference is illustrated by:Tim is 1/5 as old as Sven if Sven is 5 times as old as Tim. It is to be noted that only the numbers (N) are inverted in such cases, not the antonyms. It is not correct to conclude that Tim is 1/5 as young as Sven as that would imply that Sven was known to be young.

An inference based on the transitivity of the dimensions involved may be expressed in the following way. `l(X,D,more,Z) :- l(X,D,more,Y),l(Y,D,more,Z)`

This simply means that if X has the dimension in a higher degree than Y and Y has it in a higher degree than Z than X (also) has it in a higher degree(more) than Z. The case where numerical values are included is a bit more complicated. The following rules covers e.g. the following case:If X is 5 years older than Y and Y is 5 years older than Z than X is 10 years older than Z. `l(X,D,more,N,U,Z) :- l(X,D,more,M,U,Y),l(Y,D,more,P,U,Z),N=M+P`

We are now entering algebraic operations and further elaborations could imply solving equations, which would definitely take us inte the world of Artificial Intelligence. The age of Sven may for instance be computed if it is known that he is twice as old as Per, who is 6 years older than Bo, who is half as old as Bill, who is 60. But we do not want to prepare our program for such computations. They can hardly be included in the set of "natural implications".

Certain inferences dealing with evaluative sentences may be established although they are "fuzzy". If a person is over 60 he may be called old, and if he is under 20 he may be called young, but this varies with cultures and times. Similarly a price may be called high and the object expensive if it is above a certain figure, but this figure also varies with cultures, times - and, of course, with the type of object.

Comments on the program

The program is designed to be short and simple.It is meant to demonstrate one way of treating the scalar and comparative constructions commented on by so many linguists. The program runs on a VAX/VMS Version V4.2 and uses a Prolog version which the Department of Linguistics Lund has obtained by courtesy of the department of Cognitive Sciences Sussex (Sussex POPLOG Prolog (Version 10),1985). I am indebted to Robin Cooper for teaching me basic Prolog and to Mats Andersson for improving my program.

The first part of the program handles the parsing of declarative and interrogative sentences of the types discussed above. The predicate sent takes four arguments: a variable which is d for declarative and q for questions, a syntactic tree structure, the sentence(list) to be parsed and the empty list. The rules show which types of declarative and interrogative clauses the program can handle. The rules are written in the definite clause grammar formalism(DCG).The syntactic representation is simplified and many of the usual categorial labels have been left out in this restricted grammar.

The lexicon rules allow the insertion of certain definite nouns, which are non-neuter not to complicate inflexion by agreement (neuter and plural). The genitive form of nouns is constructed by adding an -s ('s) and that is handled by some rules below nps. Certain dimensional nouns are allowed, and some dimensional verbs. Some numerals and all integers are allowed as num in the measure phrases and a number of scalar units illustrate what can be accepted by the grammar.

The meaning of the words are given by the predicate b (for betydelse,meaning). The meaning is given by English words. We distinguish the dimensional meaning of e.g. gammal, which is age ,b(gammal,age), from the evaluative meaning of gammal,which is b(gammal,age,more). Note that absolute and comparative forms are assigned the same meaning here.

The syntactic structure derived through the parsing is translated into logical representations,e.g. for comparative 1(X,D,E,N,U,Y) where X is the object which has the property, D is the name of the property,E is the evaluation variable, N is the numeral value and U is the unit.Y is the compared object. The values of D and E are derived by the meaning relations (b).

The program cannot accept strange sentences such as "Sven är 10 meter gammal"(Sven is 10 meters old). The knowledge that certain units are associated with certain dimensions is considered to be encyclopedic and not to be included in the grammar rules. The natural place for these constraints seem to be in the translation into logical form, thus prohibiting surrealistic sentences as the one above.

Various implications are included in a special section.The program handles the fact that a person who is over 60 years old may be called old. The predicate exp inserts the value more as the E if the dimension is age and the number of years exceeds 60 (an approximation).

The interactive predicates allow adding of information to the data base. The predicate accept(X) parses the sentence, derives the equivalent logical clause and stores it. It returns the word "Jaha"(OK) as a signal. The lexical and syntactic details of the sentences are thus not stored, which is possibly a feature of psycholinguistic reality. The facts are stored without checking inconsisistences. The inference rules are used only in answering questions.

There are many different ways of asking, which can be seen in the different definitions of answer(X). A typical question is "Hur gammal är Sven?" and this is handled if answer([hur,gammal,aer,sven,?]) is typed to the computer. The program will then parse the sentence according to the syntactic rules and derive the value q, and the tree structure s(sven ,[aer],mf([hur]), gammal). The logical form 1(sven,age,N,U) will be called,and at last the values of N and U will be printed. In some cases a longer answer is generated on the basis of the logical representation. If there is no information available, if nothing has been said which allows any conclusions, the answer will be "vet inte". If there is a contradictory fact the answer will be "nej".

The interactive module of the program includes a function TEXT(X) which prints a text (series of sentences) telling about X. It will give all possible, even synonymous, verbalisations (with adjective,noun,verb) of the facts stored in the data base. The text often gets a bit verbose, but the function TEXT allows checking the program and what is stored currently in the data base. The command LOGIC(X) prints the logical (semantic) representation of the sentence X. Questions do not have logical representations in the present system (See Engdahl, 1986, for suggestions).

The command ANSWER(X) parses the sentence and gives an answer.The command SAY generates a sentence according to the grammar, prints it and adds the corresponding fact to the data base. This command simulates the situation where a speaker says something and a listener stores it. The command ACCEPT(X) is somewhat more complex as the storing is followed by a backchannel item, which is "Jaha"(OK) if the fact is new and "Jag vet" (I know) if the fact is known already. The command ASK generates a question which is also printed. These commands can be combined in different ways in order to simulate conversation. One way of doing this is shown by the command DIAL. This command calles the other commands in order to get new declarative sentences, questions, backchannel items and answers. These commands can be used in more advacnced simulations of communication, where two parts really exchange information.

Conclusion. A 7-modular model of the language user

The programming language Prolog offers a fairly convenient means for writing programs which parse and "understand" sentences. It is also very suitable for inverting the process and generating sentences on the basis of the same rules.It forces the linguist to make his ideas about syntax, lexicon, morphology, semantics, pragmatics and logic more precise. The working of the program is a proof that the analysis is correct, but not that the solution is the only possible, that it is the best, that it reflects the psycholinguistic processes of a speaker/hearer or that it would be suitable for the rest of the language. As can be seen from our program even such a small fraction as the measure and comparative sentences requires a big program.

The construction of this program handling measure and comparative sentences has lead to the establishment of seven integrated modules. The whole model may be called Integrative grammar, because the grammar is integrated among the 7 modules needed in a full model of a language user.
1.Categorial combination rules. These rules tell which categories or individual words can combine. They have to be expanded with agreement conditions in order to handle concord phenomena, which the rules in the program do not. These rules and the handling of transforms of sentences, gaps etc have been the focus of much linguistic discussion during the last decades. The categorial rules of combination also build trees in which the structure may be made more regular ("deep structures") and they give information about sentences mode (declarative and question) in a special variable. These rules can be parsed by an builtin mechanism in Prolog, which is not discussed at all here. This module is traditionally called Syntax.

2.Word category rules.
These rules tell which words fit into the categories mentioned in the categorial combination rules. This module is traditionally called lexicon and is mainly a list in the program.

In addition to the lists there are rules which may be called category derivation rules. These rules state that if a certain word belongs to a certain category,a certain variant of it belongs to a certain other category. Thus the addition of an -s makes a noun which belongs to the category genitive, and the addition of an -are to an adjective makes a word which belongs to the category comparative. These general rules do not cover all cases and a number of words have therefore to be listed separately, e.g. the irregular comparatives tyngre, mindre etc.). The contents of this module has traditionally been treated under the title Word formation or (Derivational) morphology. They get a natural place in integrative grammar.

3. Word meaning rules. These rules state that a certain word (form) has a certain meaning, may decomposed as in the dimenional adjectives under discussion. Our program gives statements such as for gammal b(gammal,age,more), where b is short for betydelse/bedeutung. A traditional entry in a dictionary would give the information about an entry (a form) at one place. These pieces of information are separated in our program.

Just as word category rules are supplemented with word category derivational rules, the word meaning rules are supplemented with meaning derivational rules. The meaning of a comparative form is thus derived from the meaning of the corresponding adjectival form by special rules.

4. Natural language-to-Logic rules (NLL rules) One rationale for translating the syntactic and lexical information into logic representation is that inferring can be expressed much more economically, if it is done with logical representations instead of natural language (surface) sentences. We will not discuss all the problems of semantic representations here.

5. Inference rules

Inference rules can be divided into encyclopedic rules and (purely) logical rules. The encyclopedic rules state that a person who is above 60 may be called old, that a town (generally) has streets, that a car (generally) has wheels etc. Logical rules are illustrated by the transitivity rules in the program, rules telling that X has more of something than Z, if X has more than Y, and Y has more than Z, or that X has more than Z if X has more than Y and Y the same as Z.

It seems the inference rules can be elaborated much if the converse terms are taken into account fully. The relations between parent and child, sell and buy are often mentioned, but there are many other predicates of this type in languages and they can also be used in inference rules (See further Sigurd,1976), e.g. teacher-student, doctor-patient, lawyer-client, like-please, know-familar.

6. Data base (memory) All linguistic interaction presupposes that the speaker has a data base (memory) in which he stores what he learns and which is being consulted when he answers questions. In the present system facts are stored by means of the built-in predicate assert. How the memory is structured and accessed is a well-known psycholinguistic problem. Some reaction time experiments (Schriefers,1985) indicate that unmarked adjectives can be produced faster than marked (negative) ones, and this interesting facts can easily be reflected in the design of the data base. Another problem is whether negative facts also should be stored, as indicated by ideas in situation semantics (Cooper,1984).

7. Interactive (pragmatic) rules The interactive commands of the last section of the program simulates interactive actions, which are triggered by the mode signals in speech. A declarative sentence triggers a back-channel item (OK etc.) and a question triggers an answer. The structure of this section is elaborated in Sigurd(1986) and has been the object of much research under the heading speech act theory. The present system has a place for these linguistic features as well.

The conclusion to be drawn from the present Prolog program for a fragment of natural language is that all the 7 modules and their interaction have to be elaborated and studied taking into account theoretical,typological and psycholinguistics facts. Sample interaction with the program is illustrated in an appendix. Extracts of the program is also given in an appendix.(The whole program is available on request.)

References

Bierwisch,M.1976. Some semantic universals of German adjectives.Foundations of Language 3

Bolinger,D.1972. Degree words. The Hague:Mouton

Cooper,R.1984. Sentence negation in situation semantics. Proc. 20th annual meeting of the Chicago linguistic society

Engdahl,E.1985. The syntax and semantics of questions with special reference to Swedish.Dordrecht:Reidel

Gnutzman,C.1974. Zur Graduierbarkeit von Adjektiven im Englischen. Linguistische Berichte 31

Lafrenz,P.G.1983. Zu den semantischen Struktur den Dimensionsadjektive in der Deutschen Gegenwartssprache. Acta Universitatis Gothoburgensis. Göteborg

Quirk,R, S,Greenbaum, G.Leech and J.Svartvik.1985. A Comprehensive Grammar of the English Language.London:Longman

Rusiecki,J.1985. Adjectives and comparison in English. A semantic study. London:Longman

Schriefers,H.J.1985.On semantic markedness in language production and verification. Diss.Nijmegen

Sigurd,B.1967. Adjektivkomparationens morfologi. Arkiv för Nordisk Filologi 82:1-4 (1967)

Sigurd,B.1976. Converse terms in Swedish.In: N.E.Enkvist and V.Kohonen (eds),Meddelanden från stiftelsen för Abo Akademi forskningsinstitut nr 8.

Sigurd,B.1986. Computer simulation of dialogue and communication.In: F.Karlson (ed), Proc. of the 3rd conf. of Scand. comptational linguistics. Helsinki:Dept of Linguistics.

APPENDIX I
SAMPLE DIALOG (SIMPLIFIED, IDEALIZED)

U(ser): SVEN HAR EN ÅLDER AV 70 ÅR. (SVEN HAS AN AGE OF 70 YEARS)
C(omputer): JAHA (OK)
U: ÄR SVEN GAMMAL? (IS SVEN OLD?)
C: JA. (YES)
U: HUR GAMMAL ÄR SVEN? (HOW OLD IS SVEN?)
C: 70 ÅR. (70 YEARS)
U: VAD ÄR SVENS ÅLDER? (WHAT IS SVEN'S AGE?)
C: 70 ÅR. (70 YEARS)
U: SVEN ÄR ÄLDRE ÄN LISA. (SVEN IS OLDER THAN LISA)
C: JAHA. (OK)
U: LISA ÄR ÄLDRE ÄN PER. (LISA IS OLDER THAN PER)
C: JAHA. (OK)
U: LISA ÄR LIKA GAMMAL SOM HANS. (LISA IS AS OLD AS HANS)
C: JAHA. (OK)
U: ÄR SVEN ÄLDRE ÄN PER? (IS SVEN OLDER THAN PER?)
C: JA. (YES)
U: ÄR SVEN ÄLDRE ÄN HANS? (IS SVEN OLDER THAN HANS?)
C: JA. (YES)
U: ÄR PER YNGRE ÄN HANS? (IS SVEN YONGER THAN HANS?)
C: JA. (YES)
U: ÄR HANS LIKA GAMMAL SOM PER? (IS HANS AS OLD AS PER?)
C: NEJ. (NO)
U: HUR MYCKET VÄGER HANS? (HOW MUCH WEIGHS HANS?)
C: VET INTE. (DON'T KNOW)
U: LISA ÄR 35 ÅR GAMMAL. (LISA IS 35 YEARS OLD)
C: JAHA. (OK)
U: HUR MÅNGA ÅR ÄR SVEN ÄLDRE ÄN LISA? (HOW MANY YEARS IS SVEN
C: 35 år. (35 YEARS) OLDER THAN LISA?)
U: HUR MÅNGA GÅNGER ÄLDRE ÄR LISA ÄN SVEN? (HOW MANY TIMES IS LISA
C: 1/2 GÅNGER. (1/2 TIMES) OLDER THAN SVEN?)
U: PER ÄR HÄLFTEN SÅ GAMMAL SOM HANS. (PER IS HALF AS OLD AS HANS)
C: JAHA. (OK)
U: HUR GAMMAL ÄR PER? (HOW OLD IS PER?)
C: 17,5 ÅR. (17.5 YEARS)
U: ÄR PER GAMMAL? (IS PER OLD?)
C: NEJ. (NO)

U: HANS VÄGER MER ÄN LISA. (HANS WEIGHS MORE THAN LISA)
C: JAHA. (OK)
U: VILKEN VIKT HAR LISA? (WHAT WEIGHT HAS LISA?)
C: VET INTE. (DON'T KNOW)
U: ÄR LISA TYNGRE ÄN HANS? (IS LISA HEAVIER THAN HANS?)
C: NEJ. (NO)

U: TEXT(SVEN).
C: SVEN ÄR 70 ÅR GAMMAL. SVEN HAR EN ÅLDER AV 70 ÅR. SVENS ÅLDER ÄR 70 ÅR. SVEN ÄR ÄLDRE ÄN LISA. SVEN HAR EN HÖGRE ÅLDER ÄN LISA. SVENS ÅLDER ÄR HÖGRE ÄN LISAS.

U: SAY.
C: SVEN ÄR STARK. (SVEN IS STRONG)
U: MSAY.
C: SVEN ÄR FEM ÅR GAMMAL. (SVEN IS FIVE YEARS OLD)
U: CSAY.
C: SVEN ÄR FEM ÅR ÄLDRE ÄN PER. (SVEN IS FIVE YEARS OLDER THAN PER)

U: CONV.
C: SVEN ÄR STARK.
JAHA.
ÄR SVEN ÄLDRE ÄN LISA?
VET INTE.

```
sent(d, s(X, [aer], Y)) --> np(X), [aer], a(Y), [.
].
sent(d, s(X, Y, i(Z))) --> np(X), mv(Y), i(Z), [.]
.
sent(d, s(X, [har], id(Y), Z)) --> np(X), [har], i
d(Y), mn(Z), [.].
sent(d, s(X, [aer], Y, Z)) --> np(X), [aer], mf(Y)
, a(Z), [.].
sent(d, s(X, [har], Y, Z)) --> np(X), [har], [en],

 mn(Y), [av], mf(Z), [.].
sent(d, s(X, Y, [aer], Z)) --> nps(X), mn(Y), [aer
], mf(Z), [.].
sent(d, s(X, Y, Z)) --> np(X), mv(Y),mf(Z), [.].
sent(d, s(X, [aer], Z, [aen], W)) --> np(X), [aer]
, compar(Z), [aen], np(W), [.].
sent(d, s(X, Y, Z, [aen], W)) --> np(X), mv(Y), co
mpar(Z), [aen], np(W), [.].
sent(d, s(X, [aer], Y, Z, [aen], W)) --> np(X), [a
er], mf(Y), compar(Z), [aen], np(W), [.].
sent(d, s(X, Y, [lika], [mycket], [som], Z)) --> n
p(X), mv(Y), [lika], [mycket], [som], np(Z), [.].
sent(d, s(X, [aer], [lika], Y, [som], Z)) --> np(X
), [aer], [lika], a(Y), [som], np(Z), [.].
sent(d, s(X, [aer], Y, [gg_saa], Z, [som], W)) -->
 np(X), [aer], num(Y), [gaanger], [saa], a(Z), [so
m], np(W), [.].
sent(d, s(X, [aer], Y, [gg], Z, [aen], W)) --> np(
X), [aer], num(Y), [gaanger], compar(Z), [aen], np
(W), [.].
sent(q, s(X, [aer], mf(hur), Y)) --> [hur], a(Y),
[aer], np(X), [?].
sent(q, s(X, [aer], mf(hur_maanga, Y), Z)) --> [hu
r_maanga], unit(Y), a(Z), [aer], np(X), [?].
sent(q, s(X, Y, mf(hur_maanga, Z))) --> [hur_maang
a], unit(Z), mv(Y), np(X), [?].
sent(q, s(Z, Y, X)) --> mf(X), mv(Y), np(Z), [?].
sent(q, s(X, Y, mf(hur_mycket))) --> [hur_mycket],
 mv(Y), np(X), [?].
sent(q, s(X, Y, mf(vad))) --> [vad], mv(Y), np(X),
 [?].
sent(q, s(X, [har], Y, mf(vilken))) --> [vilken],
mn(Y), [har], np(X), [?].
sent(q, s(X, Y, [aer], mf(vad))) --> [vad], [aer],
 nps(X), mn(Y), [?].
sent(q, s(X, Y, [aer], mf(vilken))) --> [vilken],
[aer], nps(X), mn(Y), [?].
sent(q, s(X, [aer], Y)) --> [aer], np(X), a(Y), [?
].
sent(q, s(X, Y, i(Z))) --> mv(Y), np(X), i(Z), [?]
.
sent(q, s(X, Y, Z)) --> mv(Y), np(X), mf(Z), [?].
sent(q, s(X, [aer], Y, Z)) --> [aer], np(X), mf(Y)
, a(Z), [?].
sent(q, s(X, [har], id(Y), Z)) --> [har], np(X), i
d(Y), mn(Z), [?].
sent(q, s(X, [aer], Z, [aen], W)) --> [aer], np(X)
, compar(Z), [aen], np(W), [?].
sent(q, s(X, Y, Z, [aen], W)) --> mv(Y), np(X), co
mpar(Z), [aen], np(W), [?].
```

```
/* lexikon */
np(X) --> [X], {isn(X)}.
isn(sven).
isn(lisa).
isn(stolen).
isn(bilen).
isn(stenen).
isn(mannen).
nps(X) --> [X], {isns(X)}.
isns(hans).
gen(hans, hans).
isns(Y) :- isn(X), gen(X, Y).
gen(X, V) :- name(X, Y), name(s, Z), append(Y, Z,
W), name(V, W).
mn(X) --> [X], {ismn(X)}.
ismn(bredd).
ismn(hoejd).
ismn(tjocklek).
ismn(storlek).
ismn(tyngd).
ismn(vidd).
ismn(laengd).
ismn(kvalitet).
ismn(styrka).
ismn(kvantitet).
ismn(vikt).
ismn(aalder).
ismn(kostnad).
ismn(volym).
ismn(temperatur).
a(X) --> [X], {isa(X)}.
isa(stark).
isa(djup).
isa(svag).
isa(klok).
isa(dum).
isa(tung).
isa(laett).
isa(billig).
isa(dyr).
isa(snabb).
isa(bred).
isa(smal).
isa(laang).
isa(kort).
isa(stor).
isa(liten).
isa(hoeg).
isa(laag).
isa(tjock).
isa(gammal).
isa(ung).
isa(varm).
isa(kall).
isa(bra).
isa(daalig).
i(X) --> [X], {isi(X)}.
isi(mycket).
isi(lite).
isi(foega).
id(X) --> [X], {isid(X)}.
isid(stor).
```

```
/* dimension-unit relations */
du(length, meter).
du(height, meter).
du(height, fot).
du(depth, meter).
du(temperature, grader).
du(cost, kronor).
du(weight, kilo).
du(age, aar).
du(volume, liter).
du(time, timmar).
du(time, minuter).
/* dimensional meaning of certain adjectives (tran
slations) */
b(hoeg, height).
b(gammal, age).
b(laang, length).
b(stor, size).
b(varm, temperature).
b(bred, width).
b(djup, depth).
b(tjock, thickness).
/* derivation of dimenional(unmarked) adjective */
b(X, D) :- b(X, D, more), isa(X).
/* dimenional meaning of certain verbs */
vb(kostar, cost).
vb(vaeger, weight).
vb(raeknar, size).
vb(rymmer, volume).
vb(maeter, length).
/* meaning of certain nouns */
b(kostnad, cost).
b(vikt, weight).
b(storlek, size).
b(laengd, length).
b(hoejd, height).
b(aalder, age).
/* derivation of form and meaning of comparatives
*/
cb(tyngre, weight, more).
cb(aeldre, age, more).
cb(yngre, age, less).
cb(laengre, length, more).
cb(stoerre, size, more).
cb(mindre, size, less).
cb(hoegre, height, more).
cb(mer, size, more).
cb(laegre, height, less).
cb(X, Y, Z) :- atocompar(W, X), b(W, Y, Z).
compar(X) --> [X], {iscompar(X)}.
atocompar(X, Y) :- isa(X), name(X, Z), name(are, W
), append(Z, W, V), name(Y, V).
iscompar(Y) :- isa(X), atocompar(X, Y).
iscompar(aeldre).
iscompar(tyngre).
iscompar(stoerre).
iscompar(laengre).
iscompar(laegre).
iscompar(hoegre).
```

```
/* translation into logic */
to_log(F, L) :- F=s(X, [aer], Y), L=l(X, D, E), b(
Y, D, E).
to_log(F, L) :- F=s(X, Y, i(Z)), L=l(X, D, E), vb(
Y, D), ib(Z, E).
to_log(F, L) :- F=s(X, [har], id(Y), Z), L=l(X, V,
 E), ib(Y, E), b(Z, V).
to_log(F, L) :- F=s(X, [har], Y, mf(Z, M)), L=l(X,
 V, Z, M), b(Y, V), du(V, M).
to_log(F, L) :- F=s(X, [aer], mf(Z, M), U), L=l(X,
 Y, Z, M), b(U, Y), du(Y, M).
to_log(F, L) :- F=s(G, Y, [aer], mf(Z, M)), L=l(X,
 V, Z, M), gen(X, G), b(Y, V), du(V, M).
to_log(F, L) :- F=s(X, Y, mf(Z, U)), L=l(X, D, Z,
U), vb(Y, D), du(D, U).
to_log(F, L) :- F=s(X, [aer], Z, [aen], Y), L=l(X,
 D, E, Y), cb(Z, D, E).
to_log(F, L) :- F=s(X, Y, I, [aen], Z), L=l(X, D,
E, Z), vb(Y, D), cb(I, D, E).
to_log(F, L) :- F=s(X, [aer], mf(Y, M), Z, [aen],
W), L=l(X, D, E, Y, M, W), cb(Z, D, E), du(D, M).
to_log(F, L) :- F=s(X, [aer], [lika], Y, [som], Z)
, L=l(X, D, E, like, Z), b(Y, D, E).
to_log(F, L) :- F=s(X, [aer], [lika], Y, [som], Z)
, L=l(Z, D, E, like, X), b(Y, D, E).
to_log(F, L) :- F=s(X, Y, [lika], [mycket], [som],
 Z), L=l(X, D, more, like, Z), vb(Y, D).
to_log(F, L) :- F=s(X, [aer], Y, [gg_saa], Z, [som
], W), L=l(X, D, E, Y, times, W), b(Z, D, E).
to_log(F, L) :- F=s(X, [aer], Y, [gg], Z, [aen], W
), L=l(X, D, E, Y, times, W), cb(Z, D, E).
/* implicational rule about old and age for humans
 */
exp(X, D, E) :- l(X, D, E), !.
exp(X, D, E) :- l(X, age, Y, _), Y>60, human(X), E
=more.
human(sven).
human(hans).
human(lisa).
/* inferences */
/* transitivity */

exp(X, D, E, Z) :- l(X, D, E, Y), l(Y, D, E, Z), p
rint([as, X, has, E, D, than, Y, and, Y, E, than,
Z]).
exp(X, D, E, Z) :- l(X, D, E, Y), exp(Y, D, E, Z),
 print([as, X, has, E, D, than, Y, and, Y, E, than
, Z]).
exp(X, D, E, Z) :- l(X, D, E, Y), exp(Y, D, E, lik
e, Z), print([as, X, has, E, D, than, Y, and, Y, i
s, like, Z]).
exp(X, D, E, Z) :- exp(X, D, E, like, Y), l(Y, D,
E, Z), print([as, X, is, like, Y, and, Y, has, E,
D, than, Z]).
exp(X, D, E, like, Y) :- l(X, D, E, like, Y), !.
exp(X, D, E, like, Z) :- l(X, D, E, like, Y), l(Y,
 D, E, like, Z), print([as, X, is, like, Y, and, Y
, is, like, Z]).
exp(X, D, E, like, Z) :- l(X, D, E, like, Y), exp(
Y, D, E, like, Z), print([as, X, is, like, Y, and,
```

```
/* interactive commands */
logic(X) :- sent(T, F, X, []), to_log(F, L), print
(L), nl.
synt(X) :- sent(T, F, X, []), print(F), nl.
accept(X) :- sent(T, F, X, []), to_log(F, L), asse
rt(L), print(jaha), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, [aer],
 mf(hur_maanga, Z), W), l(Y, D, N, Z), b(W, D), pr
int([N, Z]), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, Z, mf(
hur_maanga, W)), l(Y, D, N, M), vb(Z, D), print([N
, M]), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, [har],
 Z, mf(vilken)), l(Y, V, N, U), b(Z, V), print([N,
 U]), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, Z, mf(
hur_mycket)), l(Y, D, N, U), vb(Z, D), print([N, U
]), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, Z, mf(
vad)), l(Y, D, N, U), vb(Z, D), print([N, U]), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, [aer],
 mf(hur), Z), l(Y, D, N, U), b(Z, D), print([N, U,
 Z]), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, Z, [ae
r], mf(vad)), l(G, D, N, U), b(Z, D), gen(G, Y), p
rint([N, U]), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, Z, [ae
r], mf(vilken)), l(G, D, N, U), b(Z, D), gen(G, Y)
, print([N, U]), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, [aer],
 Z), exp(Y, V, E), print(ja), b(Z, V, E), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, [aer],
 Z), exp(Y, V, P), P\=E, b(Z, V, E), print(nej), n
l.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, Z, i(W
)), vb(Z, V), ib(W, E), exp(Y, V, E), print(ja), n
l.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, Z, i(W
)), vb(Z, V), ib(W, more), exp(Y, V, less), print(
nej), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, [har],
 id(Z), V), b(V, D), ib(Z, E), exp(Y, D, E), print
(ja), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, Z, mf(
V, W)), l(Y, D, V, W), vb(Z, D), print(ja), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, [aer],
 mf(V, M), Z), l(Y, D, V, M), b(Z, D), print(ja),
nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, [aer],
 Z, [aen], W), exp(Y, D, E, W), cb(Z, D, E), print
(ja), nl.
answer(X) :- sent(T, F, X, []), T=q, F=s(Y, [aer],
 [hur_mycket], Z, [aen], W), l(Y, D, E, N, M, W),
b(Z, D, E), print(N, M), nl.
```