Coding and texts
A "distant reading" approach

Stefano Penge, may 2022

Part I
Introduction

What is this lesson about

- This is a lesson that simulates a didactic project
- There are some learning objectives for the interdisciplinary part and some learning objectives for the CS part
- The contents of the lessons are interleaved with comments on why and how the teacher should introduce a concept or a tool, or to how the activities should be organized
- The typical target for this project is a class of 13-17 years old students
- The duration of the projects depends on the previous experiences and on the competences of the students; if fully extended, it could be a one year project
- The coding environment chosen for this simulation in Snap! but the majority of the concepts could be taught also in any other language equally expressive

Distant reading

- What is distant reading? "Distant reading is an approach in literary studies that applies computational methods to literary data, usually derived from large digital libraries, for the purposes of literary history and theory."
- Franco Moretti in his article "Conjectures on World Literature" fosters the use of of samples, statistics, paratexts, and other features not often considered within the ambit of literary analysis.
- https://en.wikipedia.org/wiki/Distant_reading
- DR is nowadays considered as a part of Digital Humanities ("Informatica Umanistica")

Goals

- Computer are good as crunching numbers, but they are also good at counting words
- Father Roberto Busa SJ was one of the first researcher to use computer (an IBM Mainframe) to analyze a corpus of text, namely the work of Saint Thomas Aquinas, and to lemmatize the entire Summa Theologiae.

  An archaeological work of reconstruction of this work of half a century can be found here: http://www.digitalhumanities.org/dhq/vol/14/3/000456/000456.html
- The simple feasibility is not a reason for taking such an approach: once we have measured a text, once we have counted the times a verb appears or the characters that are introduced by author, what are our earnings? Should one have in mind some hypotheses before the reading, or should one simply wait for a form to appear from data?
- There are a number of reason to do this kind of analysis: an enquiry on the author's style, on the text readability, on the occurrence of some topics
- We will take here a public text: not a story or an essay, but the Italian Constitution
- This lesson is based on Chapter 2 of "Dati, cittadinanza e coding", Anicia, 2022

The text

- The Italian Constitution was written by a team of 75 men and women during several months after the end of the World War II (1946-1947)
- The goal was not only to fix the principles on which all other laws would be based in the newly born republic, but also to lay them down in a plain language, so that every Italian citizen could read and understand them
- Readability was a central issue for a rising democracy
- In the same time, since the Constitution is a small text, the team should have been sure that all the fundamental topics of the common life would have been covered by its first 54 items (12 Principles + 42 Article) in an balanced way

Learning objectives /1

- The learning objectives, at the disciplinary level, will be to understand what is readability, and how can it be calculated
- The students will also learn to lemmatize and index a text by extracting roots from words
- They also will learn how to thematize a text

Questions

- Our questions will be:
  - which is the longest article? and the shortest?
  - which is the average length of articles?
  - which is the most frequent word? and the least frequent?
  - how many different words are present?
  - how many simple/complex words are present? which is the percentage ?
  - which are the overall themes? among these, which are the most important?

Learning Objectives

- The learning objectives at the Computer Science level will be:
  - to deal with text and tables
  - order, search, compare words and sentences
  - functions as first grade objects
  - map, reduce, filter

Prepare the data

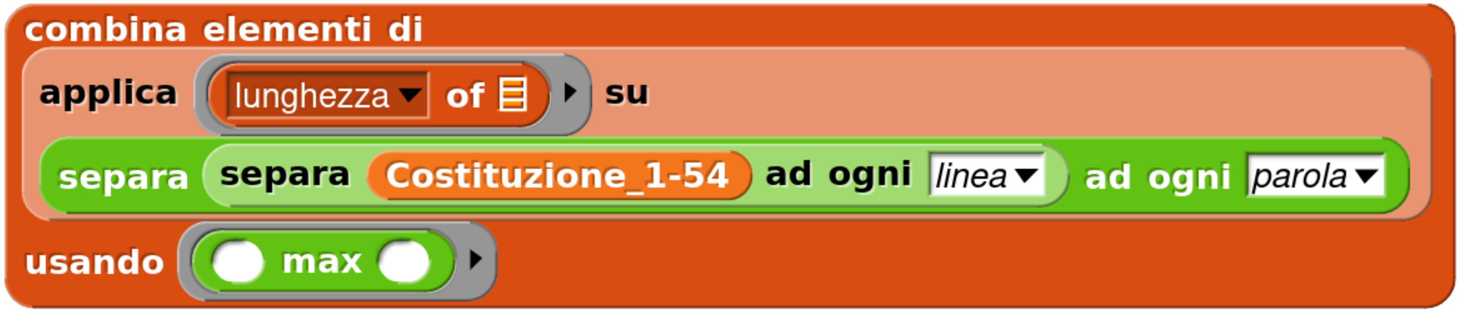
- There is no official text-only edition of the Italian Constitution; we have to start from one of the different PDF versions available and exclude all that is not useful for our treatment (font, dimension, color)
- We have to eliminate notes, comments, numbers and to transform the text in a list of articles
- We have to trim also strange characters like apostrophe, quotation marks – but this leads to a problem: how we will treat elision? Should we register "L'Italia" as an expression composed by two lemmas or as a single lemma?
- The last problem arises because apostrophe is an important sign in written Italian. Clearly using a English text would greatly simplify the job. This will be clearer when facing the problem of conjugations

Part II
The Lexicon

The longest article

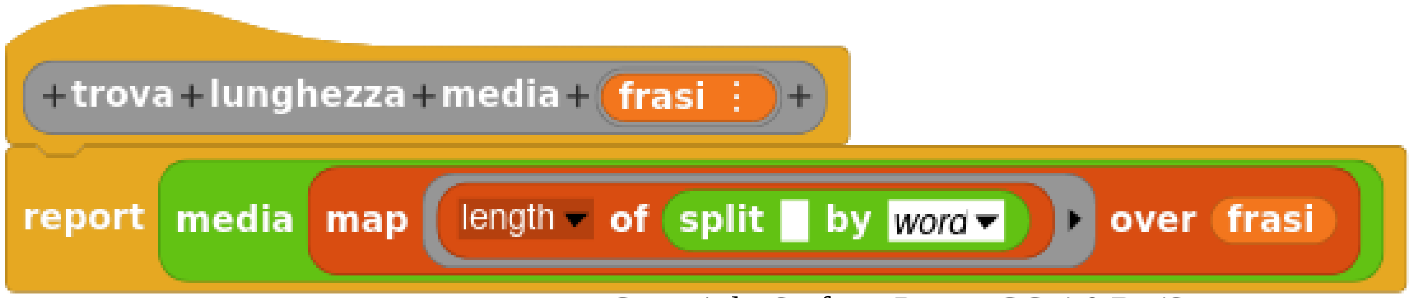
- An article too long (in terms of words, not of letters) is surely complex to read and understand. At the same time, certain topics surely need more details than others. Which is the relation between topic and length?
- So we may wonder which is the longest article. We have to:
  - create a list of articles
  - divide every article in words
  - count the number of words per article
  - order by length

- Split the text in lines, the split every line in words; apply the function "length of" to the resulting list and then reduce the result using the function "max"



```
combina elementi di
  applica (lunghezza of ☰) su
    separa (separa Costituzione_1-54 ad ogni linea) ad ogni parola
usando (( ) max ( ))
```

- To get the smaller article, it is sufficient to substitute the *max* function with the *min* function
- To calculate the average length of the Articles (which is a simple index of complexity) we could apply the "split by word" function onto the list of sentences



```
+trova+lunghezza+media+ frasi ⋮ +
report media map (length of split [ ] by word) over frasi
```

Blocks and concepts used

- In the blocks shown in the previous slide we have a lot of concepts typical of the functional approach:
  - Combine: returns a *single* element applying a function over the elements of a list
  - Map: returns a new *list* applying a function over a list
- Combine is a form of the classical "reduce" method of functional languages
- We also introduce methods to transform lists to strings and viceversa
  - Split: divides a string in list elements using a boundary separator
  - Join is the inverse method: returns a string from a list and a separator. It will be useful to "render" the results in the future

Count the lemmas

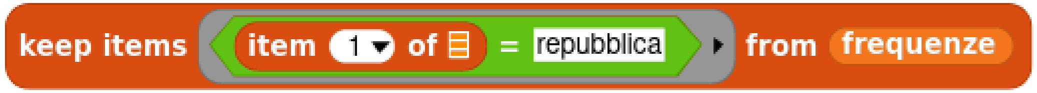
- Talking about readability, we could wonder how many *different* words the Constitution contains – because a text with to many different words is more difficult to read.
- We should:
  - build a list of all the words
  - order alphabetically the list
  - count the duplicates and build another list of list <word>,<frequency>
  - then exclude duplicates and build an index of the lemmas (words used at least one time)

Opacity

- The functions that create the index could be too much complex for the competences of the students; in this case the teacher may decide to give them to his/her students as opaque blocks to be just used – not analyzed and improved
- Some of these functions could run slowly, depending on the way they are implemented and on the speed of the computer; in this case, the teacher may decide to "hide" them and simply give the final result (the final list) pre-calculated
- Note that Snap! can save the content of variables with the program by declaring them "not transient"

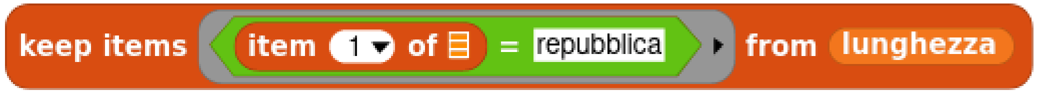
Frequency/1

- Home many times the Costituzione speaks about rights? How about searching the frequency of a specific word?
- We may use the block "keep items", a filter which returns all the elements for which the function used as first parameter returns TRUE. Since "frequenze" is a list of list, we should compare the first element of the list elements



```
keep items ( item 1 of ⌸ = repubblica ) from frequenze
```

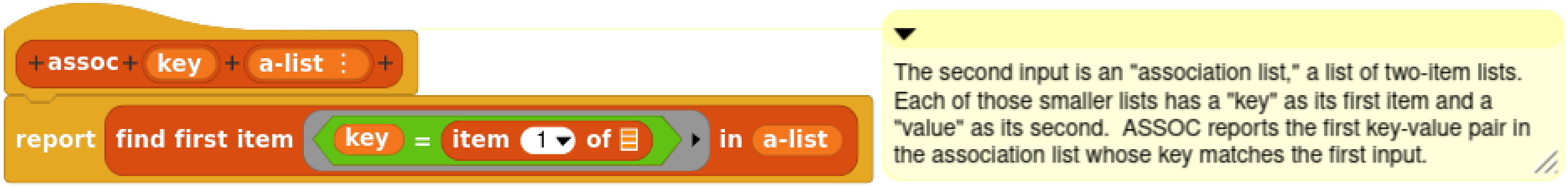
We can do the same for length of words



```
keep items ( item 1 of ⌸ = repubblica ) from lunghezza
```

Find in an association list

- The generalization is one of the central competence that a student should learn.
- Instead of teaching it as one of the Sacred Principle of the Art of Programming, it would be better to suggest at this point of the project that we could need to apply the same function to different data structure that have the same structure
- On the other way round, we build those data structure in that way to be able to write a general function to deal with them



```
+assoc+ key + a-list ⋮ +
report find first item ( key = item 1 of ☰ ) ▸ in a-list
```

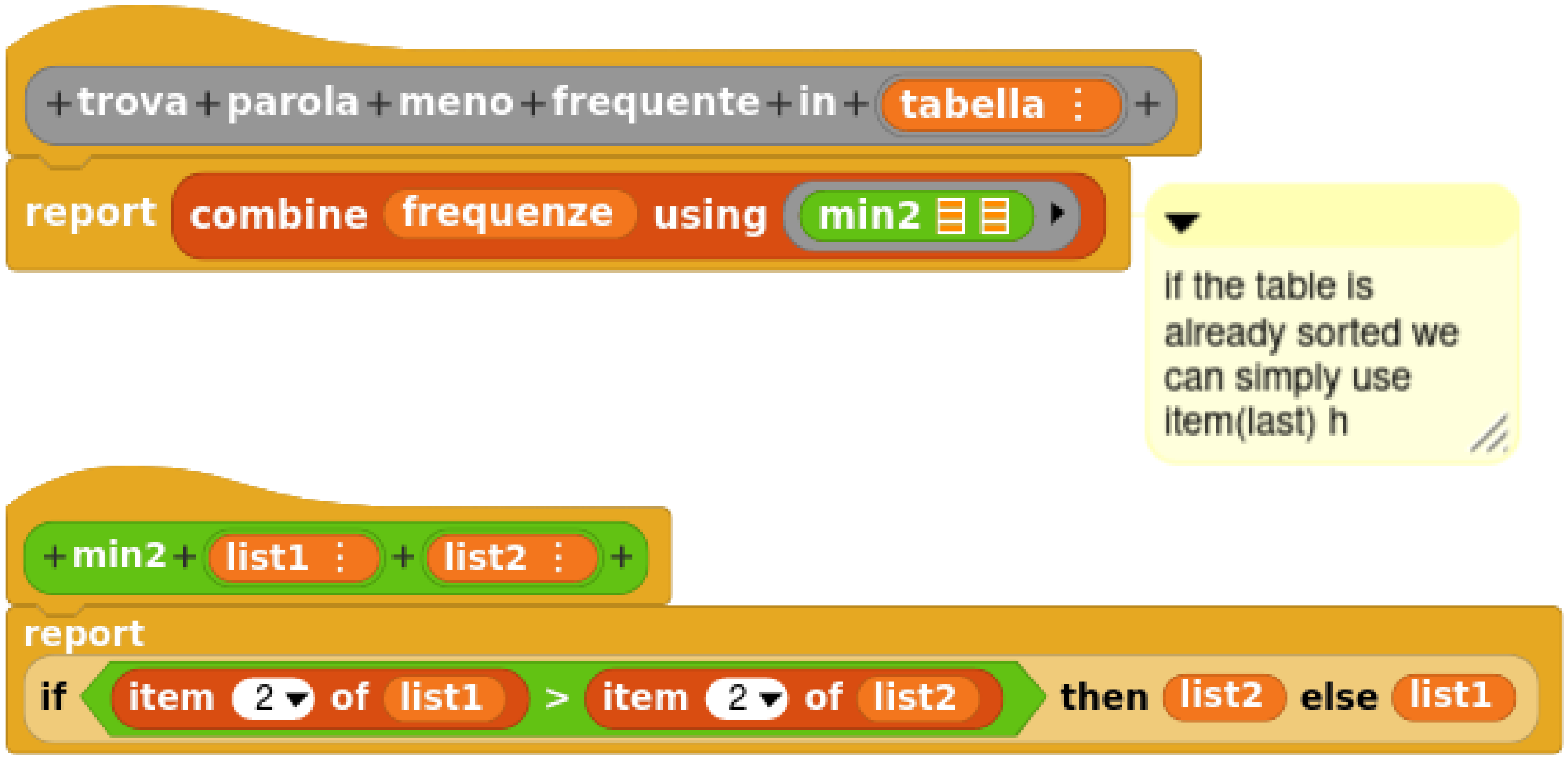
The second input is an "association list," a list of two-item lists. Each of those smaller lists has a "key" as its first item and a "value" as its second. ASSOC reports the first key-value pair in the association list whose key matches the first input.

A ring for all?

- We can ask to students if, in their opinion, it is worthy to solve this problem once for all; generally speaking, how much work should be done in advance, just to be prepared in case, and how much should be left to the moment in which it is absolutely necessary to do so
- We could make a "debate": is it better to build default data structures and in this way keep the code smaller, more manageable and less prone to errors, or is it better to design specific data structure trying to represent the real relationships among data, with the need to write different functions?
- This kind of debates have the goal to push the students to be aware of the way they are programming. These meta-competences are similar to those required to speak fluently a language it it is not your mother language

Frequency/2

- How about searching the most (least) frequent word?
- If we had an ordered list <word>,<frequency>, we should have simply get the first (or last) element.
- We should order it on the second element (the frequency, not the word), and this kind or ordering is not as simple to do, even though Snap! has a lot of "addon" function that are borrowed from APL Language and are oriented to vector manipulation – but this can be an advanced topic
- If we don't want to build such an ordered list, we still could build a function that uses "combine" to compare every element of the list with each other and to report the one which has the max (or min) frequency index
- We cannot use the function "max" which accepts only numbers, but we have to build a new function comparing the first element of a li



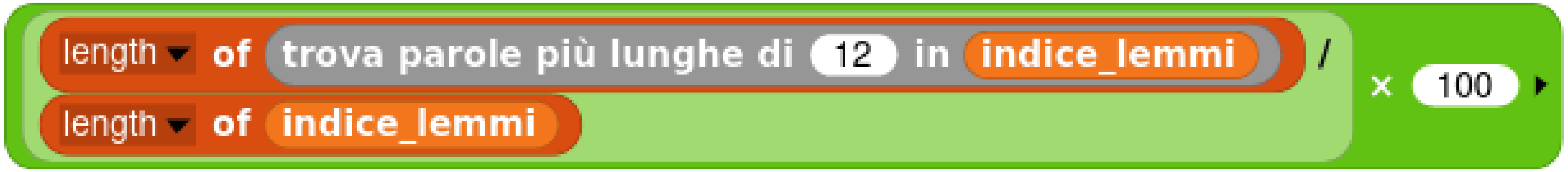
```
+trova+parola+meno+frequente+in+ tabella ⋮ +
report combine frequenze using min2 ▤ ▤ ▸
```

If the table is already sorted we can simply use item(last) h

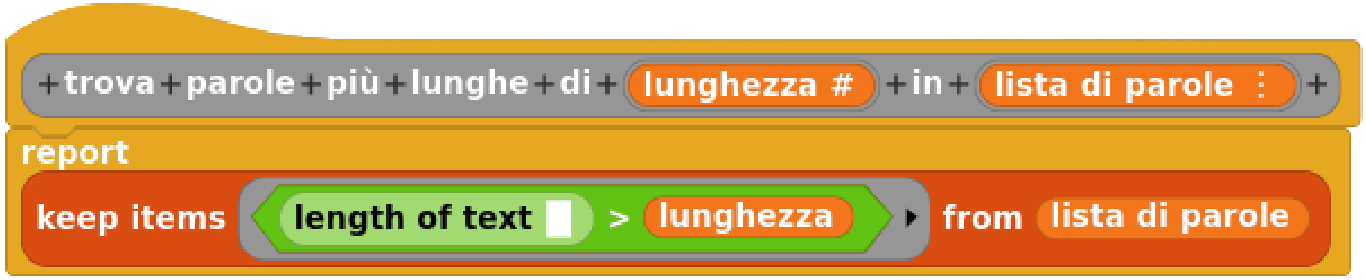
```
+min2+ list1 ⋮ + list2 ⋮ +
report
if item 2 ▾ of list1 > item 2 ▾ of list2 then list2 else list1
```

How many long (short) words

- How many long words are there on the total of words (in percentage)?



```
length of (trova parole più lunghe di 12 in indice_lemmi) /
length of indice_lemmi
× 100
```



```
+trova+parole+più+lunghe+di+ lunghezza # +in+ lista di parole ⋮ +
report
  keep items (length of text [ ] > lunghezza) from lista di parole
```

Results

- Total words: 2034
- Total lemmas: 897
- Most frequent word: "è" (135)
- Less frequent word: "abbiano" (1)
- Longest word: "originariamente" (15)
- Average length of words: 5.43
- Percentage words longer than 12: 4.43 %
- Percentage words shorter then 6: 22.18 %

Readability indexes

- There are some well known readability indexes, that is algorithms that calculate a number from a text.
- These algorithms use the length of the words, the length of the sentences and the number of sentences in the text
- They don't use the semantic difficulty (the rarity of a word), or the syntactic difficulty (the specific complexity of an series of relatives, like "John says that Mary think that Peter suppose that…")

Flesch

- The Flesch index (1948) has been adapted to Italian by Roberto Vacca:

$$206 - (0.6 \times S) - W$$

- where S is the number of syllables and W is the average number of words per sentence.

Gunning-Fog

- The Gunning-Fog index takes explicitly into account the number of complex words (in terms of syllables length). This is the formula:

$$0.4 \times [(WS) + 100 \times (CW)]$$

wher WS is words/sentences and CW is complex words/words

GULPEASE

- The GULPEASE index (1988) proposes this formula:

$$89 - (LW / 10) + (3 \times SW)$$

- where LW equals:

$$(100 \times \text{total letters}) / \text{total words}$$

- and SW equals:

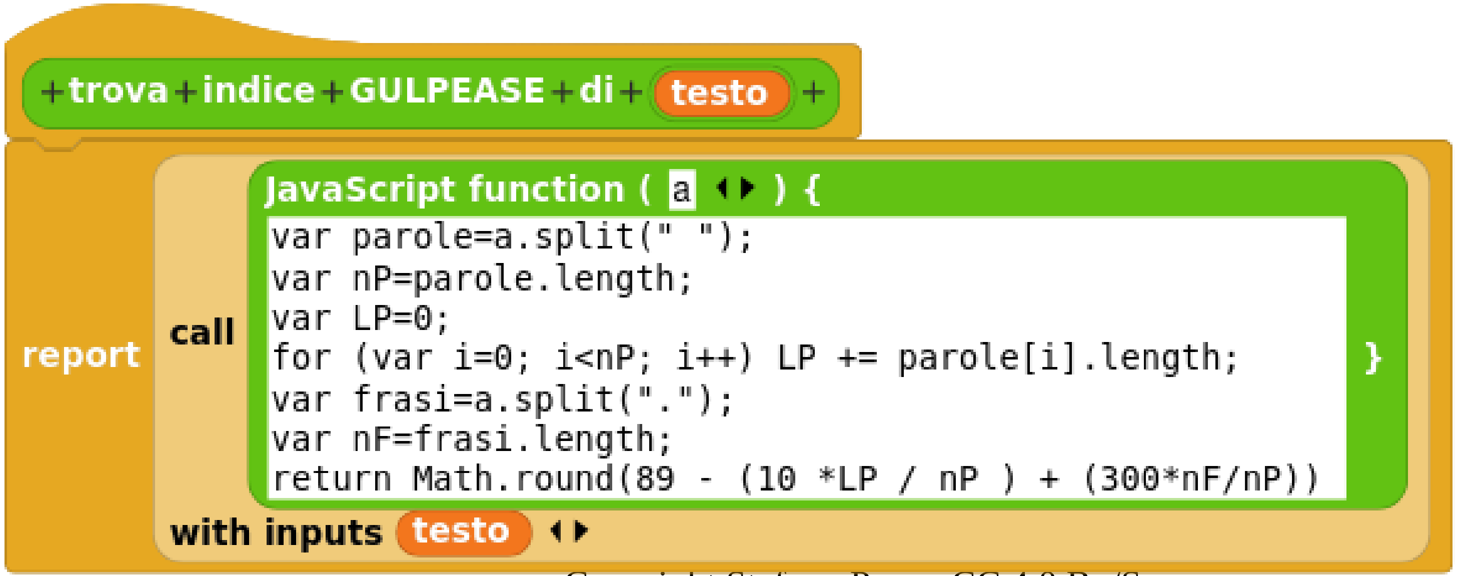
$$(100 \times \text{total sentences}) / \text{total words}$$

Creativity

- Even if it will be simple and attractive for the teacher to ask his/her students to implement all the three formulas and to confront the results, this is not very attractive for the students: it is just a translation task.
- More interesting could be to face the problem from the scratch: what do we call "readable"?
  - readable for all or only for adults with a standard culture?
  - readable now or readable for the citizens trying to read the text back in 1948?
- The students could be asked to read a number of different texts (newspaper articles, essays, short stories, emails) and to order them by readability; then to try to define which are the parameters that make the text n° 1 the most readable for all
- At this point, they can try to design an algorithm to automatically calculate an index

Gulpease

- If there is no time to imagine an alternative algorithm...
- This is a good way to introduce a powerful concept: languages are not closed world. It is often possible to use a language inside another.
- In this case we use Javascript within the "call" block to create Gulpease index


+trova+indice+GULPEASE+di+ testo +
report
call
JavaScript function (a) {
}
with inputs testo

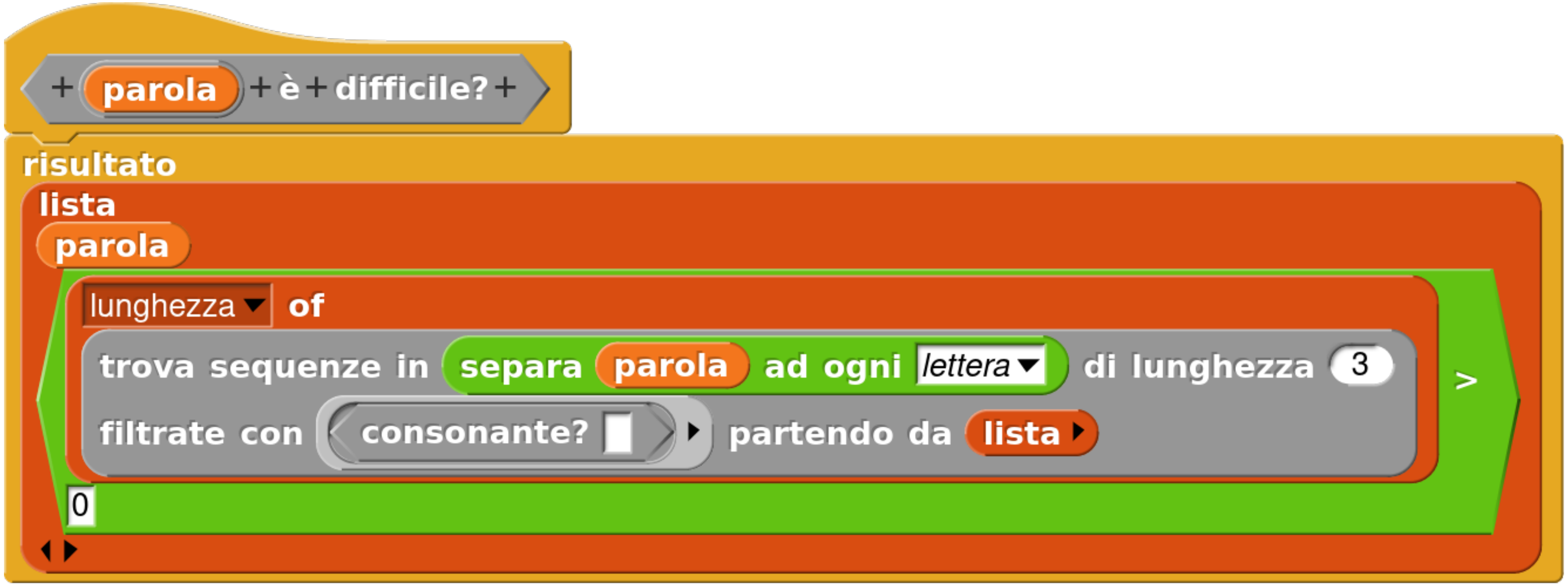
```
var parole=a.split(" ");
var nP=parole.length;
var LP=0;
for (var i=0; i<nP; i++) LP += parole[i].length;
var frasi=a.split(".");
var nF=frasi.length;
return Math.round(89 - (10 *LP / nP ) + (300*nF/nP))
```

Difficult words

- Which are the words difficult to read?
  - Word with strange orthography (e.g. with a lot of consonants)
  - Long words
    - compound words
    - derived words
  - Rare words
    - words of a specialistic language (for example, juridical)
    - words derived from Latin or Greek
    - words from other contemporary languages

Consonants and difficulty

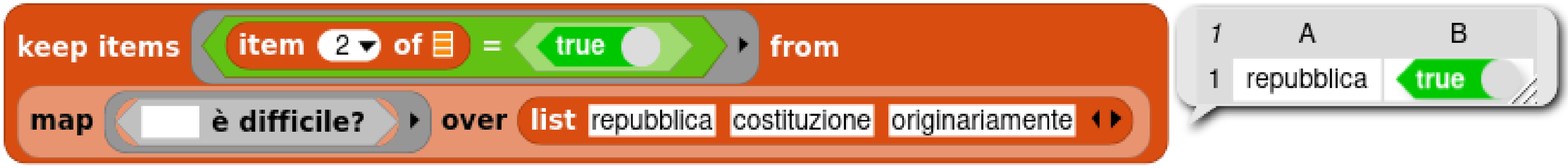
- A word is difficult to read (in Italian and for an Italian reader) if it has more then 3 consonants



```
+ parola + è + difficile? +
risultato
lista
parola
lunghezza of
  trova sequenze in separa parola ad ogni lettera di lunghezza 3
  filtrate con consonante? partendo da lista
> 
0
```

Comparison

- This function could be used to find if a word is difficult



1	A	B
1	repubblica	true

- This approach allow students to freely explore the text to validate certain hypotheses (is "originariamente" a difficult word? is "repubblica" a difficult word?)
- The students could try to compare difficulty of words using a mixed approach: too much consecutive consonants plus a length of more than 12 letters

Rare words

- How can we recognize a "rare" word? Can we simply measure its frequency in this text? or in all text of the same genre? or in all Italian texts?
- This could be an interesting research: measure the frequency of different words in the official documents (e.g. Ministry's communications to school, principal's communications to teachers, teacher's communication to family, and so on)
- But how should we treat "studente, studenti, studentessa, studentesse"? As four different words or as a single occurrence of a same lemma? The same for "esce, escono, uscirà, usciranno".

Different vocabulary and grammars

- We encounter here one of the biggest difficulties of our distant reading project. Italian language uses heavily conjugation. This means that for a verb there could be a lot of different forms (mode x time x person x number). Also, there is the concordance of adjectives on gender and number. This two facts together lead to a great number of different possible forms for a single lemma.
- In English there are much less different forms. There are a special forms for the third person (with -s); adjectives don't vary.
- To avoid this difficulty, we can decide to use an English translation of the Italian Constitution (there exists one here: https://www.senato.it/1024)
- If we decide to go on with the Italian version, we should find a solution. Before giving it to the students it could be a good idea to ask them to try to imagine it

Basic Vocabulary

- A well know work by professor De Mauro (1980, 2nd release 2016 with Isabella Chiari) defined a Basic Vocabulary for Italian Language
- Their research produced a list of 2.000 Italian words that are supposed to be known by all Italian native speakers
- The original list was published in PDF and was not designed to be applied by a software; luckily Alberto Pettarini released a much more useful version on Github: https://github.com/pettarin/nvdb
- With some work we can produce a table with a word per row and variant in different columns

Basic Italian Vocabulary

a s.f. e m.inv.

a prep.

abbagliante p.pres., agg., s.m.

abbaiare v.intr. e tr.

abbandonare v.tr.

abbandonato p.pass., agg., s.m.

Tabled BIV

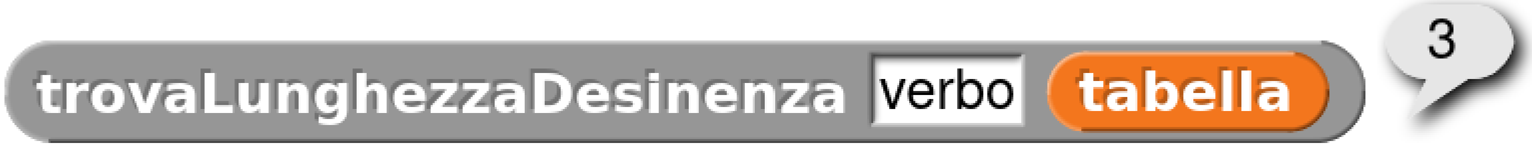
a	preposizione	sostantivo
abbagliante	aggettivo	sostantivo
abbaiare	verbo	
abbandonare	verbo	
abbandonato	aggettivo	sostantivo
abbandono	sostantivo	

Roots

- To recognize a verbal form, we have to define a root. For example, adottare → adott(are) → adott(ato)
- We can try to establish (with a variable degree of precision) how many letters should be trimmed on the end of a word to get the root on the basis of its grammatical role.
- Once we know which is the grammatical category of a word, we can extract the root: adott-

Finding the root/1

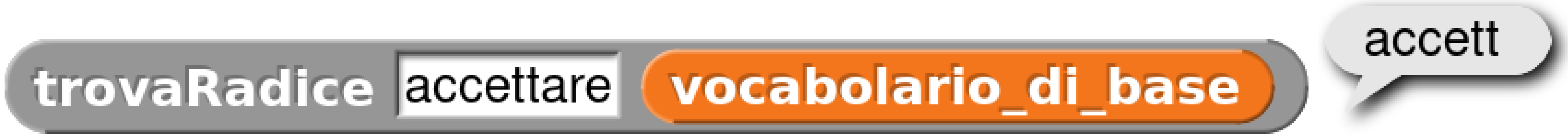
category	letters to be trimmed off	example
sostantivo	1	*adozion*-e
verbo	3	*adott*-are
aggettivo	1	*democratic*-o
avverbio modo	6	*democratic*-amente



Find the root/2

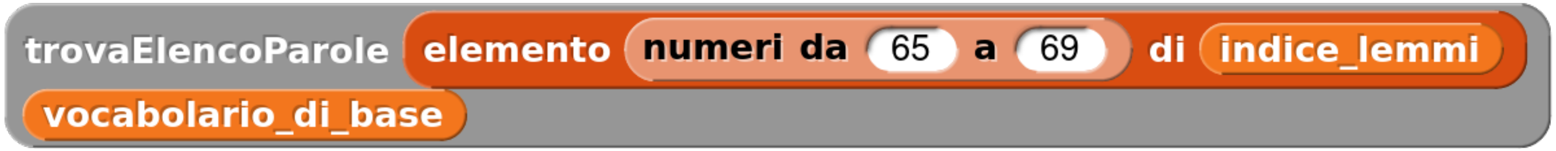
At this point, we should have a table like this one below:

Word	**Root**
adozione	adozion
adottare	adott
democratico	democratic
...	...

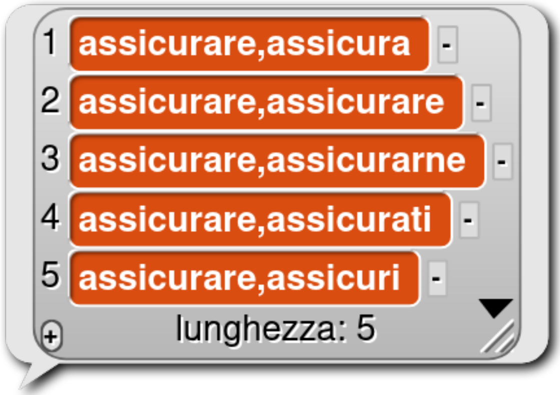


The lemma index

In the end, we can build an index with the main form (the lemma) and the occurrences (the forms)
We can query this index to find the root of different words
For example: if we search for "assicura", "assicurarne, "assicurati", "assicuri", we should get always the lemma "assicurare"



```
trovaElencoParole elemento numeri da 65 a 69 di indice_lemmi
vocabolario_di_base
```



```
1 assicurare,assicura
2 assicurare,assicurare
3 assicurare,assicurarne
4 assicurare,assicurati
5 assicurare,assicuri
lunghezza: 5
```

Now we can compare all the words of the Costituzione with the Basic Vocabulary

Results/1

- We found that the number of words of first 54 Articles of the Italian Constitution which are in the *Nuovo Vocabolario di Base* is of 811 su 897.
- Which means more than 90 %.
- We can enrich this result with other data calculated with our block: find how many words are too long, how many sentences are too long
- We can also give some examples

Results/2

- The Italian Constitution has 897 words, of which 811 are in the Vocabolario di Base
- 41 are too long; 56 are rare
- 12 sentences are too long; 5 are too complex.
- These are the more difficult words
  - 1. …
  - 2. …
  - 3. …
- These are the more difficult sentences:
  - 1. …
  - 2. …
  - 3. …

Part III
The meanings

Social profiling

- How (in a very, very simple manner) can we extract meanings from texts?
- We can start to talk about social profiling: how can Facebook know which are our interest?
- Partially because we declare them; partially on the basis of the interests of our "friends"; partially reading our posts

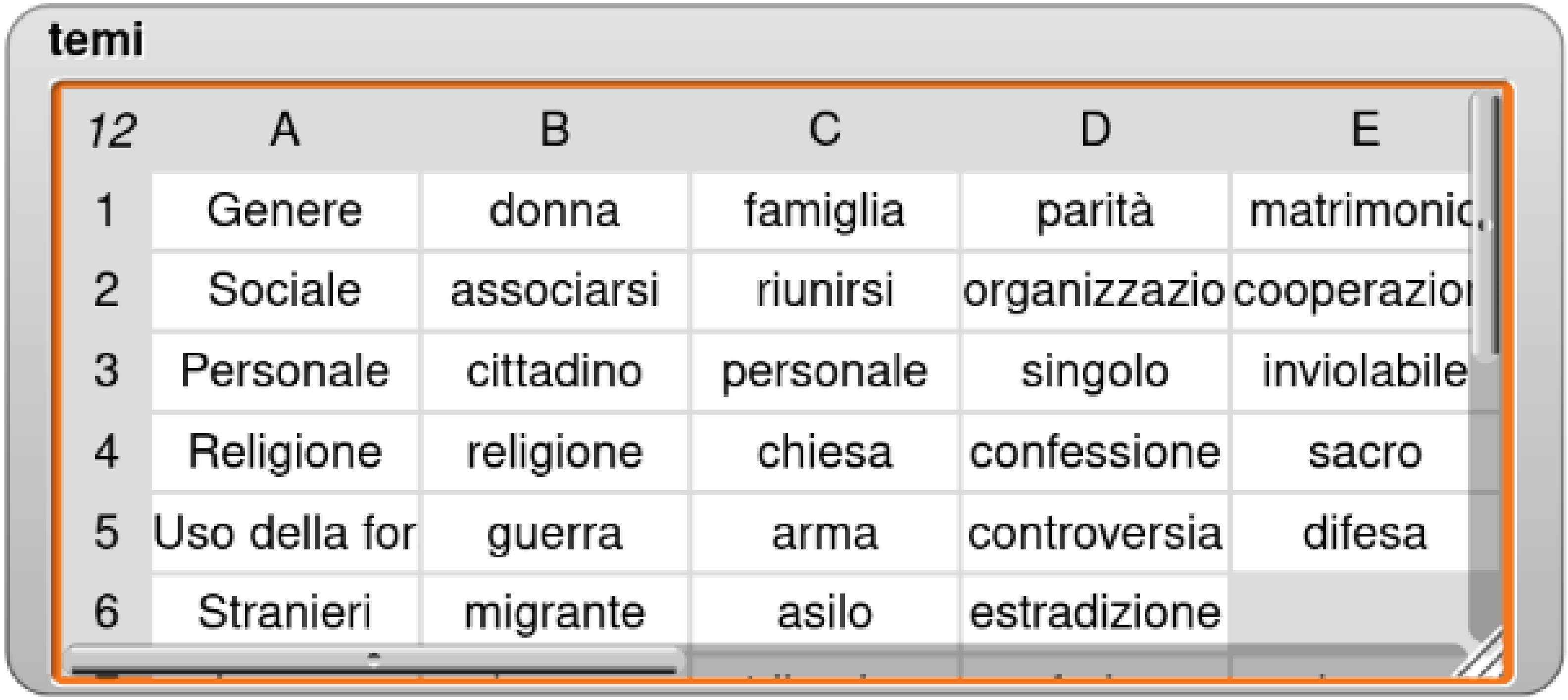
Thematization

- The simplest approach is probably the one based on themes and keywords
- A "theme" is recognized when one or more words of a closed list are found in the text
- For example, when in a post we found the words "medicine, cure, pills, drugs" probably the theme of the post is "health care"

Before starting

- The standard approach is to try to define the themes manually, just by reading the text and compiling a table with keywords that are signs of the presence of that theme
- This is an activity that should not be done by the teacher, but should be assigned to the students, in groups; the results should be compared and discussed

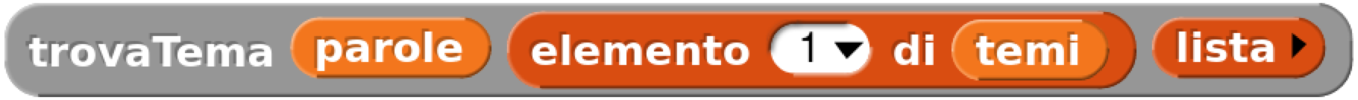
The Thematic Categories

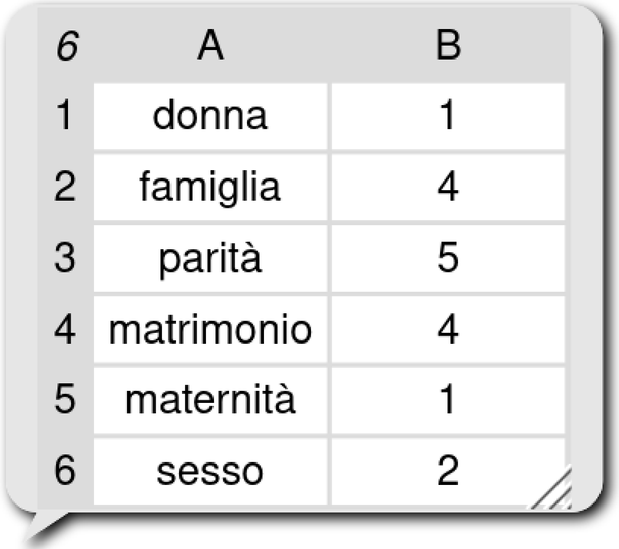


temi

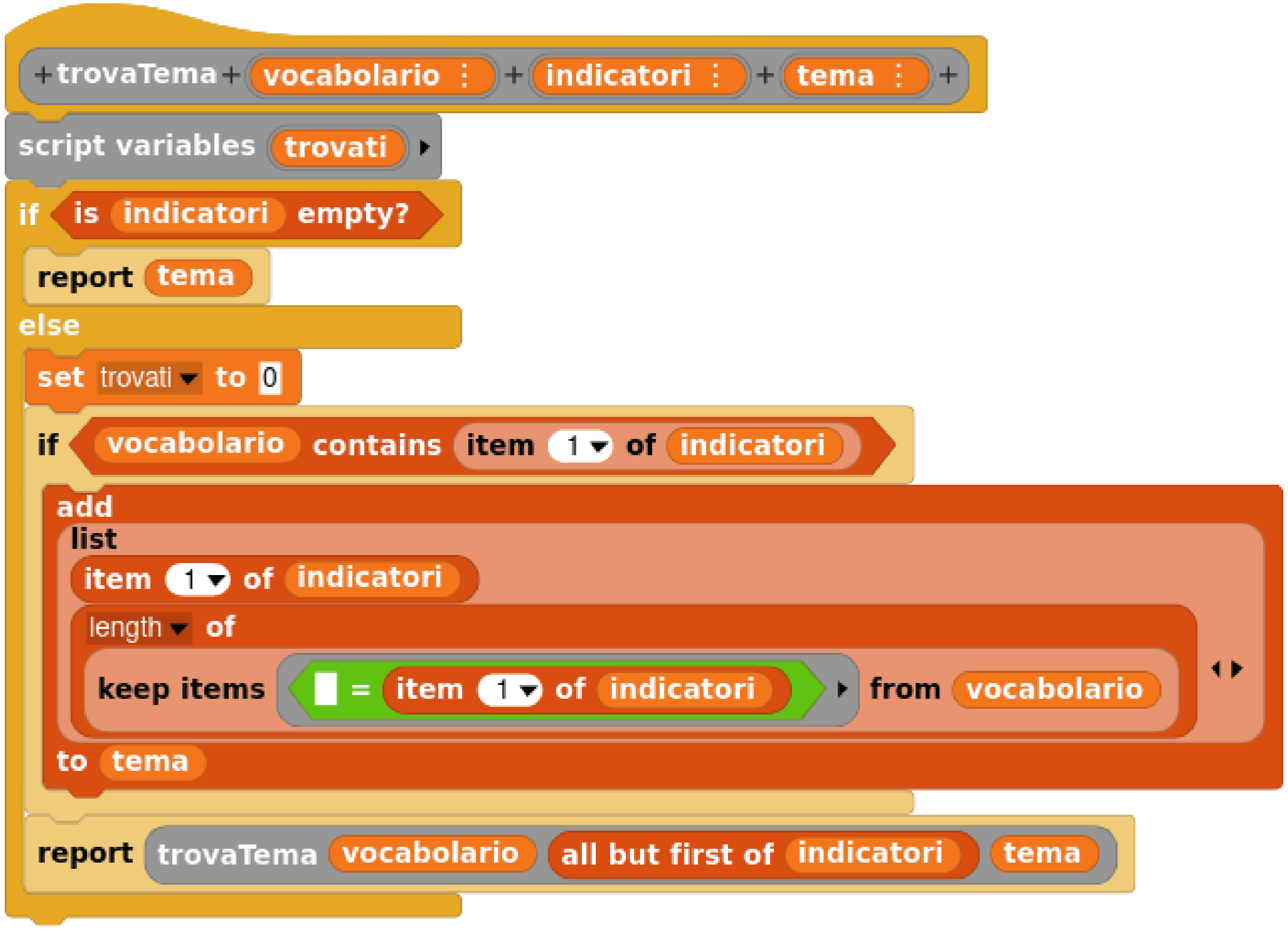
12	A	B	C	D	E
1	Genere	donna	famiglia	parità	matrimonio,
2	Sociale	associarsi	riunirsi	organizzazio	cooperazio
3	Personale	cittadino	personale	singolo	inviolabile
4	Religione	religione	chiesa	confessione	sacro
5	Uso della for	guerra	arma	controversia	difesa
6	Stranieri	migrante	asilo	estradizione	

- To find the detail about how many times in the text there are words related to the "family" theme we could build a block like the one shown in the next slide
- Here "parole" is the index, and "temi" is the list of topics we saw before (Gender, Social, Personal, Religion, Use of force, …)
- This is the result for "Gender" theme





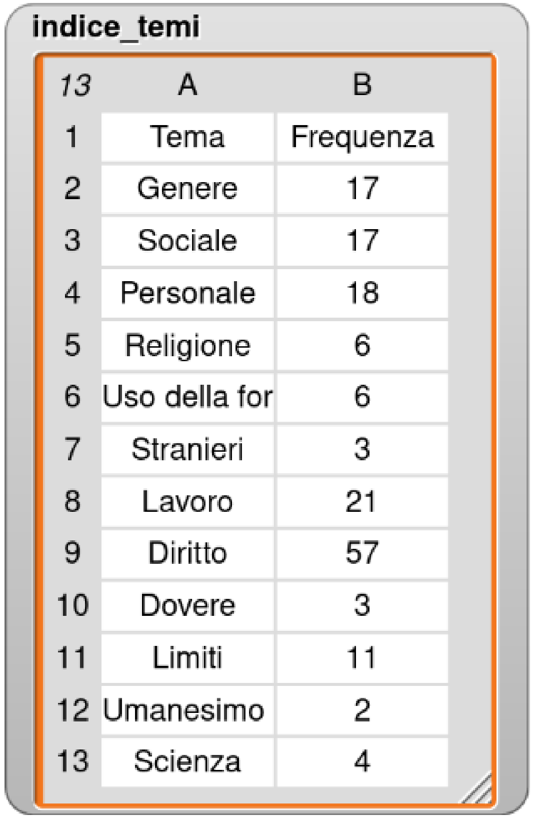
6	A	B
1	donna	1
2	famiglia	4
3	parità	5
4	matrimonio	4
5	maternità	1
6	sesso	2



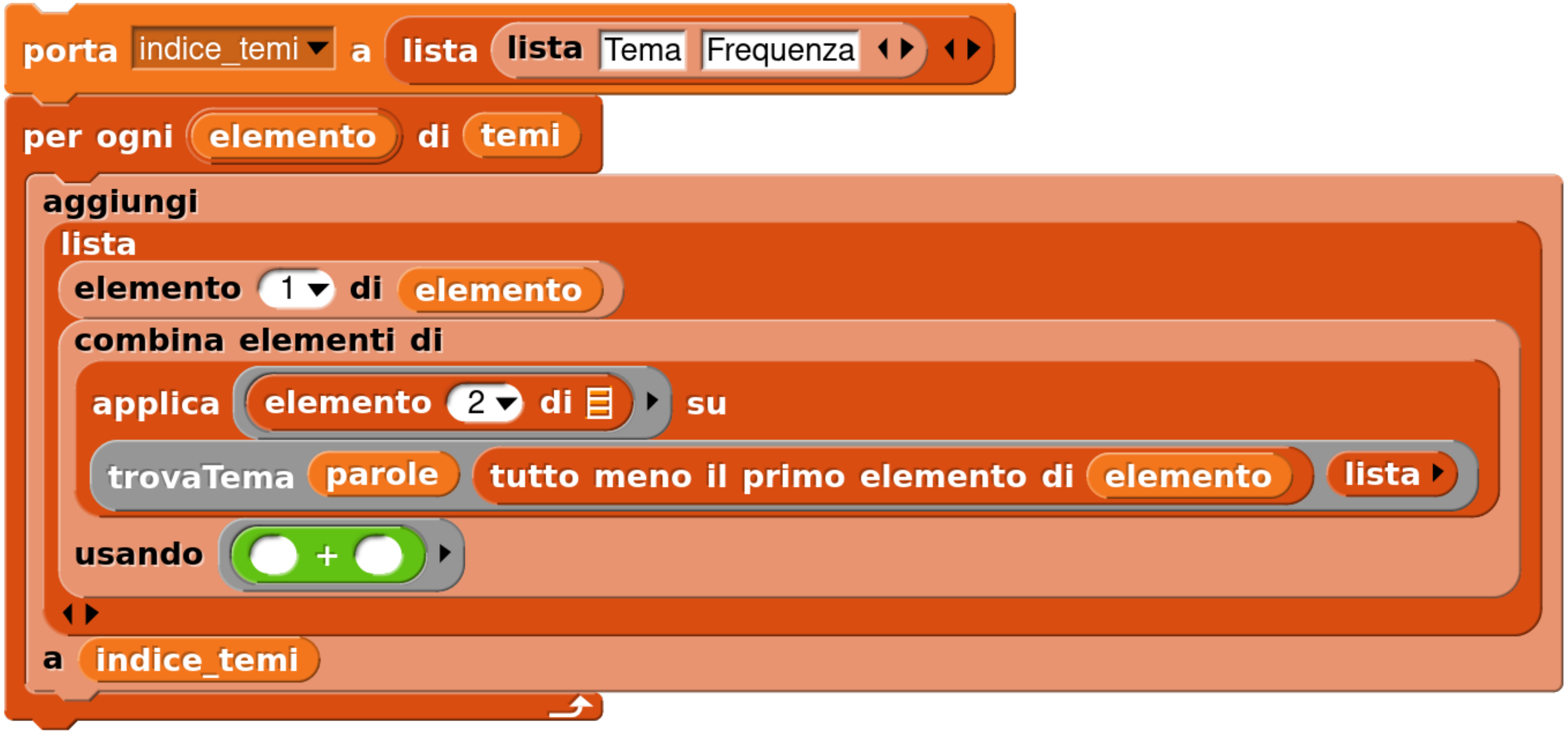
```
+trovaTema+ vocabolario : + indicatori : + tema : +
script variables trovati
if is indicatori empty?
    report tema
else
    set trovati to 0
    if vocabolario contains item 1 of indicatori
        add
        list
            item 1 of indicatori
            length of
                keep items ( [] = item 1 of indicatori ) from vocabolario
        to tema
    report trovaTema vocabolario all but first of indicatori tema
```

Frequency index

- In the end, we can build a table in which is shown the frequency of every theme in our list
- This table can be further analyzed to discover if in the Constitution there is more room for humanities or for science, for rights or for duties.


indice_temi

13	A	B
1	Tema	Frequenza
2	Genere	17
3	Sociale	17
4	Personale	18
5	Religione	6
6	Uso della for	6
7	Stranieri	3
8	Lavoro	21
9	Diritto	57
10	Dovere	3
11	Limiti	11
12	Umanesimo	2
13	Scienza	4



```
porta indice_temi a lista (lista Tema Frequenza)
per ogni elemento di temi
    aggiungi
        lista
            elemento 1 di elemento
            combina elementi di
                applica (elemento 2 di ☰) su
                    trovaTema parole (tutto meno il primo elemento di elemento) lista
                usando ( + )
    a indice_temi
```

Other approaches

- Among the Snap! addons (called library) there is a SciSnap!, which, as one can easily guess, is a collection of function to do some Machine Learning (but also more mathematics and some SQL query on an external db)
- Even without this kind of tools, we can imagine a naïve Machine Learning approach:
  - manually categorize certain articles as "speaking about" certain topics
  - then calculate the similarity of every article with those model
- This can be an interesting task: what is similarity and how can we calculate it? Can we use Levenshtein distance?

Representing data

- A third, simpler, approach, could be that of visual representation. It is not an essay to define a theme, but simply to leave to human reader to make her/his idea starting from a concise, spatial, colored (or with some sound) representation of words
  - Dataviz is a trendy topic and a tour with the students in some examples could be interesting
  - Data visualization is not a recent field of research and teaching. The Isotype system was developed by Otto Neurath and Gerd Arntz in Wien between 1925 and 1934
- A final activity of this project could be that of representing frequency of words with a word cloud
- Dimension could be used to indicate frequency, intensity of the color can be used to indicate rarity
- From the CS learning objectives, this could be an opportunity to teach the use of clones in Snap!, heritage, recursion.

modi
uguaglianza
adempime
stabilire
straniero
effettivo
sanità
giuridica
libero
rapporti
regolati
speciali
parità
interesse
sicure
degli
responsabilità
morale
funzione
professionale
risparmio
riunioni
previsto
suo
pubblica
statal
sulle
istituti
base
dovere
questa
avorisce
utilità
sul
liberamente
devono
successive
lavoratore
limitazioni
soltanto
questo
territorio
capacità
famiglia
inviolabile
anche
provvedimenti
nessuno
politica
quarantotto
penale
economica
ventiquattro
uffici
matrimonio
sua
garantisce
pubblici
siano
reati
privi
italiano
sindacati
mezzi
costitu
registrazione
unità
religione
stabilisce
ettive
internazionali
obbligh