Team Software Engineering

Introduction

First, 30 minutes to design together. Then, you are coding alone for 2 hours. Finally, 30 minutes to put your work together. You're going to need a very good design to make everything work together!

Competition

You will design a search engine that receives a query and returns every sentence that matches this query from its sentences database.

Expected input/output

Input

A Command Line Interface that takes the sentences file as the first argument and the query as the second.
So you should have something like this : ./program sentences.txt "this is a query"

Output

Results printed to the console

You will find a base for python and java that does that. Your code goes in search_engine.py / SearchEngine.java. You can edit anything in this base to fit your needs, but you must meet the I/O requirements. You can also use any language if you meet those requirements.

You will also find an example sentences.txt file that contains sentences for your tests. You may add whatever you want to this file. This is not the sentences.txt file that will be used for correction, we will use different sentences.

Run in Java

Go to the bin folder generated by your IDE. Run this command :
java org.csgames.tse.Main ../sentences.txt “this is an \”example query\””
If you don’t use an IDE for java, you should already know how to compile and run it.

Run in Python
Go where main.py is. Run this command :
python main.py sentences.txt “this is an \”example query\””

Queries

Notes

- [ ] in explanations are used to distinguish words in explanation from words in query/sentence
- These are not the same queries that will be used during correction : words are not the same, number of words in exact match not the same, different CALC operators, etc.
- Results are not case sensitive (see query #1), but you can consider that keywords AND, OR, NOT, CALC are always going to be in CAPS in the query.
- You need to match the exact word (see query #1 again). There is no punctuation at the end of words. If there is, for example, a question mark at the end of a sentence, then it’s going to be separated by a space ([Hello ?], not [Hello?])
- Wild card (*) replaces 0 or more characters (see query #7)
- Wild card (*) does not mean the same thing dependant of the context (mixed with a word, in a quote, etc.)

Format

1st line : Self-explanatory...
2nd line : Query description : example query
3rd line : What the query should return
4th line : Precisions

Simple queries

#1 (2 pts)
Simple word : example
Returns sentences that contain the word [example] (**exact word**, should not find [exam]).
Precisions : remember.. **NOT CASE SENSITIVE**. [Example], [example], [exAmPLe].. it shouldn’t matter.

#2 (2 queries, 2 pts each)
Exact match (quotes) : “an example”

Returns sentences that contains the exact sequence [an example].

#3 (3 pts)

AND : an example
Returns sentences that contains both words, anywhere in the sentence.
Precisions : AND is implicit.

#4 (3 pts)

OR : an OR example
Returns sentences that contains either words.

#5 (3 pts)

NOT : NOT example
Returns sentences that does not contain [example].

#6 (2 pts)

Wild card : *
Returns everything.

#7 (4 pts)

Wild card in exact match : “this * example”
Returns sentences with the following sequence : [this], followed by anything, then finishes by [example].
Precisions : * replaces 0 or more characters (the sequence can be [this example]).

#8 (3 pts)

Wild card at the end of a word : exam*
Returns sentences with words that begins with [exam].

#9 (3 pts)

Wild card at the beginning of a word : *ample
Returns sentences with words that ends with [ample].

#10 (2 pts)

Wild card at the beginning and the end of a word : *test*
Returns sentences with words that contains [test].

#11 (2 queries, 3 pts each)

CALC : {CALC 5+5*5}
Returns sentences that contains 30
Precisions : The brackets are always there, and there is no space between numbers. You have

to consider priority of operators and parentheses.

#12 (3 pts)

Range : 67..90
Returns sentences that contains numbers between 67 and 90
Precisions : Note it's **TWO** dots

#13 (3 pts)

Multiple occurrence : example{2}
Returns sentences that contains the word example, AT LEAST twice (number in brackets)

Mixed queries

#14 (4 pts)

Simple word + AND + Exact match : hello "an example"
Returns sentences that contains [hello] AND [an example]

#15 (4 pts)

Simple word + AND + NOT : hello NOT example
Returns sentences that contains hello and does not contain example

#16 (4 pts)

Simple word + AND + OR : hello test OR example
Returns sentences that contains [hello] AND [test], OR, simply [example]

#17 (4 pts)

Exact match + CALC : "There are {CALC 34+8} ninjas in this query"
Returns the same thing as if the query was : "There are 42 ninjas in this query"

#18 (4 pts)

Range + AND + simple word : 10..100 beers
Returns sentences that contains numbers between 10 and 100 AND the word [beers] in it.

#19 (4 pts)

Range + CALC : 42..{CALC 9*(30+30)}
Same as CALC 42..69

#20 (4 pts)

Exact match + multiple occurrences : "an example"{3}
Returns sentences that contains [an example] at least 3 times.

#21 (4 pts)

Simple word + AND + multiple occurrences : an{2} example

Returns sentences that contains [an] at least twice AND [example] at least once.

#22 (4 pts)

Simple word + OR + multiple occurrences : an{3} OR example
Returns sentences that contains [an] at least 3 times OR example at least once.

#23 (4 pts)

Range + OR : 15..25 OR 58..93
Returns sentences that contains a number between 15 and 25 OR a number between 58 and 93

#24 (4 pts)

Wild card in words and exact match : “this * exam*”
Returns sentences with the following sequence : [this], followed by anything, followed by a word that begins with [exam].

#25 (15 pts)

Fun query :
"the * example"{4} {CALC 8*(8+9/3)}..{CALC 90000/10+1} OR *park NOT bench