# TEST TASK

Create a simple REST API service using Node.js (Typescript is preferred). Also you can use any bundler if you want.

Service must accept a set of mathematical expressions and evaluate the results.

Service must contain following endpoints:

* POST /data
* GET /result

### POST /data endpoint

* Request: a set of mathematical expressions in JSON format

Example of HTTP request body:

{"expressions":[

"(10 + 16) / 2" ,

"5\*3\*2",

"2+2 \* 2",

"200 / (50\*2)",

"(14-7) \* (21/3)"

]}

* Response:
  + HTTP status 200 (if everything is ok)
  + HTTP status 400 (if input data is invalid)

### GET /result endpoint

* Request: (no data)
* Response: HTTP status 200 and a set of evaluated results of mathematical expressions in JSON format. Example of HTTP response body:

{"results":[13,30,6,2,49]}

If no results are available, then return HTTP status 200 and empty result set:

{"results":[]}

### Requirements

* Service must parse each mathematical expression from POST /data request and evaluate the result.
* Each expression can contain any integer numbers between 0 and 1000.
* Each expression can contain any of the following symbols: ' ', '+', '-', '\*', '/', '(', ')'.
* You can use any node packages for this service apart from evaluating mathematical expressions logic. **Parsing/evaluating of mathematical expressions must be performed without using eval() or any third-party modules/libraries.**
* Evaluated results of mathematical expressions must be saved to the file “results.txt” on a server side. Results must be separated with line break.
  + Example of results.txt file data:

13

30

6

2

49

* Service must store only one file with last evaluated results. All previous results in a file must be overwritten.
* The order of results must correspond to the order of input mathematical expressions.
* Writing and reading from the “results.txt” file must be asynchronous.
* When GET /result is requested, the service must read the last evaluated set of results from the “results.txt” file and send it back to the user according to the above protocol.
* Application must work with any number of mathematical expressions that are given within POST /data request.

The goal of the task is to create a program with clean, readable and extensible code, that does the specified job.