

Infilect Assignment

Problem Assignment:

Part A: Largest Rectangle in Matrix via Fast API Problem Statement: You are given a matrix of integers. Your task is to find the largest rectangle formed by similar numbers in the matrix. A rectangle is defined by selecting a group of adjacent cells that contain the same number. The rectangle should have the maximum area among all rectangles formed by similar numbers.

Part B: Create a Fast API service that accepts the matrix on a POST API and returns the largest rectangle's area and the number itself.

Methodology:

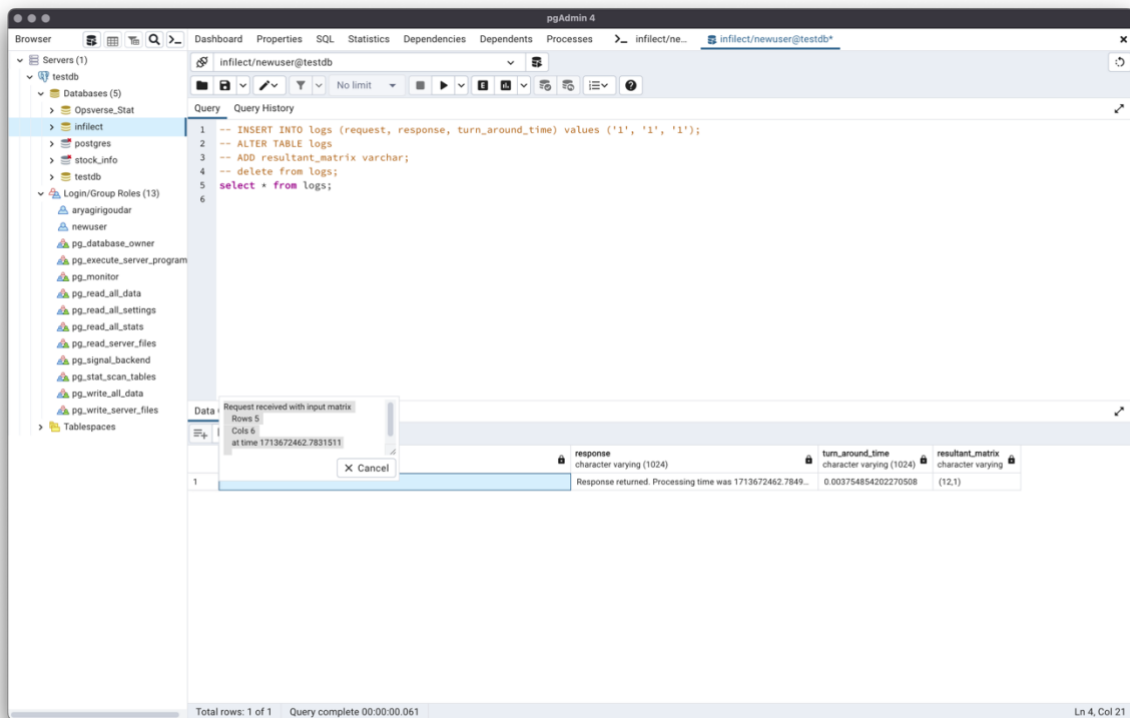
1. We first try to construct height list that is height of rectangle, i.e. when we add 1 to every element below element then result is height of that rectangle and we do this for whole matrix.
2. While 1st row is calculated we use monotonic stack to track increasing heights, because we need max height,
 - a. Height is already calculated, width can be calculated by adding current index with index of top of stack (largest yet rectangle). If stack is empty then there is not rectangle larger than current rectangle
For ex: if height is 2, and 3rd index rectangle is smaller than 2 then we pop element from stack and calculate width as 1 (current-index (3)-1 – stack index (2)) $3 - 2 = 1$
3. We do this for 1 and 0.
4. If there are numbers that are other than 1 and 0 then we need to get unique numbers and loop through one of each of them and temporarily make other numbers as 0, while we keep count of max area also
5. Due to requirement not clearly mention, Assuming squares as not rectangle, to do this we have condition to check if width = height that is if we have width same as height then its square

Edge Case: When matrix rows are not all same, then find out the row and append 0

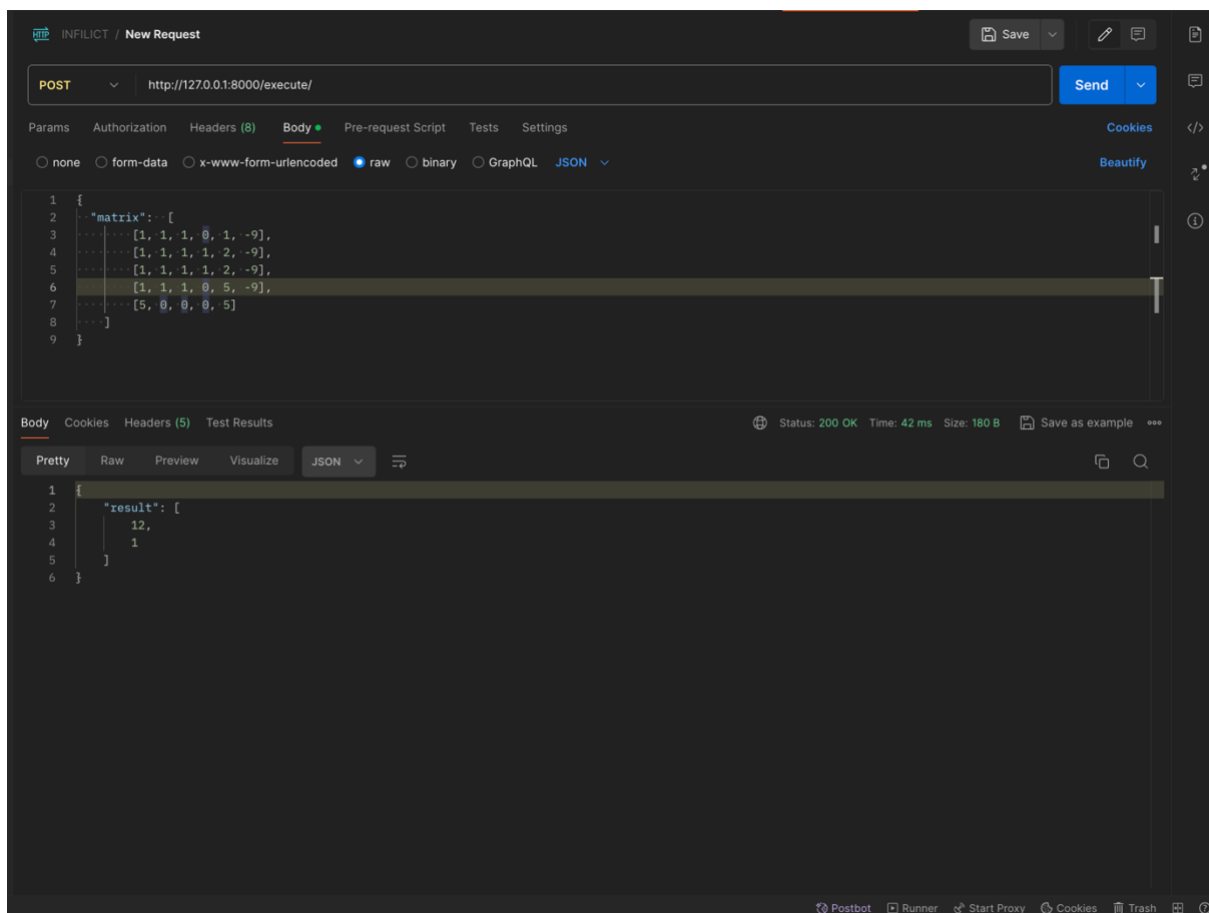
Request Log in app.log file

```
1 21-Apr-24 09:27:41 - Database connection was successful
2 21-Apr-24 09:27:41 - App will run on http://127.0.0.1:8000
3 21-Apr-24 09:28:03 - Request received with input matrix
4   Rows 5
5   Cols 6
6   at time 1713671883.2225978
7   Input Matrix : matrix=[[1, 1, 1, 0, 1, 1], [1, 1, 1, 1, 1, 1], [1, 1, 1, 0, 1, 1], [1, 1, 1, 0, 5, 1], [5, 0, 0, 0, 5]]
8 21-Apr-24 09:28:03 - Committed to database
9 21-Apr-24 09:28:03 - Inserted records
10 21-Apr-24 09:28:03 - inserted into DB
11
```

pgAdmin Screen shot of request and response being logged with result and turnaround time



API Result in postman



Matrix 2 (input matrix with 2 forming max rectangle)

INFILCT / New Request

POSThttp://127.0.0.1:8000/execute/

Send

ParamsAuthorizationHeaders (8)BodyPre-request ScriptTestsSettings

noneform-datax-www-form-urlencodedrawbinaryGraphQLJSON

```
1 {
2   "matrix": [
3     [2, 2, 2, 2, 1, -9],
4     [2, 2, 2, 2, 2, -9],
5     [2, 2, 2, 2, 2, -9],
6     [2, 0, 2, 0, 5, -9],
7     [5, 0, 0, 0, 5]
8   ]
9 }
```

BodyCookiesHeaders (5)Test Results

Status: 200 OKTime: 10 msSize: 181 BSave as example

PrettyRawPreviewVisualizeJSON

```
1 {
2   "result": [
3     12,
4     2
5   ]
6 }
```

PostbotRunnerStart ProxyCookiesTrash

pgAdmin output

Data OutputMessagesNotifications				
	request character varying (1024)	response character varying (1024)	turn_around_time character varying (1024)	resultant_matrix character varying
1	Request received with input matrix	Response returned. Processing time was 1713686713.6807...	0.0035550594329833984	(8,1)
2	Request received with input matrix	Response returned. Processing time was 1713686713.6807...	0.004896879196166992	(8,1)
3	Request received with input matrix	Response returned. Processing time was 1713686713.6807...	0.005652904510498047	(8,1)
4	Request received with input matrix	Response returned. Processing time was 1713686713.6807...	0.0016047954559326172	(8,1)
5	Request received with input matrix	Response returned. Processing time was 1713672462.7849...	0.003754854202270508	(12,1)
6	Request received with input matrix	Response returned. Processing time was 1713672462.7849...	0.00463414192199707	(12,1)

Total rows: 6 of 6Query complete 00:00:00.179Ln 4, Col 21

Matrix 3 (Original Matrix)

The screenshot displays the INFILICT web interface for a REST client. At the top, the title bar reads "INFILICT / New Request". Below this, the request method is set to "POST" and the URL is "http://127.0.0.1:8000/execute/". A "Send" button is visible on the right. The "Body" tab is selected, showing the request body in raw format:

```
1 {
2   "matrix": [
3     [1, 1, 1, 0, 1, -9],
4     [1, 1, 1, 1, 2, -9],
5     [1, 1, 1, 1, 2, -9],
6     [1, 0, 0, 0, 5, -9],
7     [5, 0, 0, 0, 5]
8   ]
9 }
```

Below the request body, the response is shown in the "Body" tab, formatted as JSON. The status is "200 OK", the time is "50 ms", and the size is "180 B". The response body is:

```
1 {
2   "result": [
3     8,
4     1
5   ]
6 }
```

The interface includes various tabs for "Params", "Authorization", "Headers (8)", "Body", "Pre-request Script", "Tests", and "Settings". There are also buttons for "Cookies", "Beautify", and "Save as example". The bottom status bar shows "Postbot", "Runner", "Start Proxy", "Cookies", "Trash", and a help icon.