

Question 1

Question1 : Write API:

Using Python Flask or ExpressJS, Write a REST API that reads the body and returns JSON.

```
const express = require('express');
const app = express();
const bodyParser = require('body-parser');
const cors = require("cors");

let port = process.env.PORT || 5000;
app.use(express.json());
app.use(bodyParser.urlencoded({ extended: true }));
app.use(cors());

app.get('/find_symbols_in_names',(req, res) => {

// let obj = (req.body);// this code use to get the request from frontend

let object = {
    chemicals: [ 'Amazon', 'Microsoft', 'Google' ],
    symbols: [ 'I', 'Am', 'cro', 'Na', 'le', 'abc' ]
};
let newObject = [];

let forchemicalsarrays = object.chemicals;
let forsymbolsarrays = object.symbols;

forchemicalsarrays.forEach((element1) =>
{
    for(let element2 of forsymbolsarrays)
    {
        if(element1.includes(element2) === true)
        {
            let index = element1.indexOf(element2)
let str = element1.substring(0, index) + `[${element2}]`
            + element1.substring(index+element2.length, element1.length+1);

            newObject.push(str);
        }
    }
});

a=newObject;

const str=new String(a);

str.slice(0,30)

const aa=str.replace(/"/g, '')
const ob={
```

```

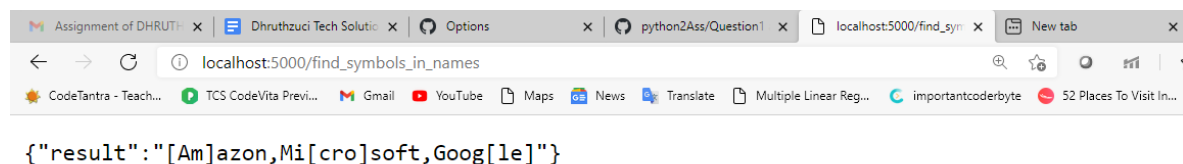
    result:aa,
}

    res.send(ob);
})

app.listen(port,()=>
{
    console.log(`Server started at ${port}`);
});

```

Output in chrome browser



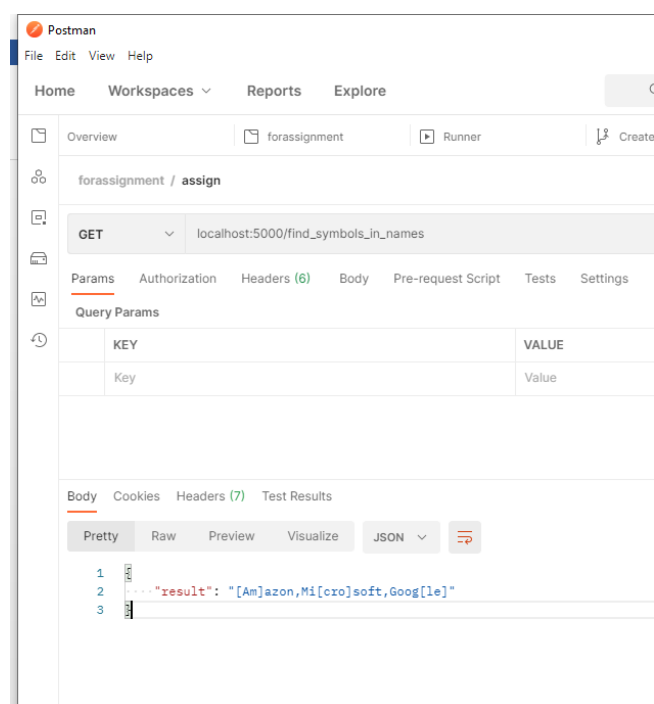
localhost:5000/find_symbols_in_names

```

{"result": "[Am]azon,Mi[cro]soft,Goog[le]"}

```

Output in postman



Postman

File Edit View Help

Home Workspaces Reports Explore

Overview forassignment Runner Create

forassignment / assign

GET localhost:5000/find_symbols_in_names

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```

1 {
2   "result": "[Am]azon,Mi[cro]soft,Goog[le]"
3 }

```

Question 2:

Given two arrays, write a function to compute their intersection.

```
class Main2(object):
    def problem(self, nums1, nums2):
        """
        :type nums1: List[int]
        :type nums2: List[int]
        :rtype: List[int]
        """
        m = {}
        if len(nums1)<len(nums2):
            nums1,nums2 = nums2,nums1
        for i in nums1:
            if i not in m:
                m[i] = 1
            else:
                m[i]+=1
        result = []
        for i in nums2:
            if i in m and m[i]:
                m[i]-=1
                result.append(i)
        return result

ob1 = Main2()
res1=[]
res2=[]
number_list1 = []
n = int(input("Enter first list"))

# print("\n")
for i in range(0, n):
    print("Enter index", i,)
    item = int(input())
    number_list1.append(item)
for i in number_list1:
    if i not in res1:
        res1.append(i)

number_list2 = []
n = int(input("Enter second list "))

# print("\n")
for i in range(0, n):
    print("Enter index", i, )
    item = int(input())
    number_list2.append(item)
for i in number_list2:
    if i not in res2:
        res2.append(i)
print(ob1.problem(res1,res2))
```

case 1:

The screenshot displays the Visual Studio Code interface with a Python file named `union.py` open. The code defines a function `problem` that finds the intersection of two lists. The terminal shows the execution of the script, where the user enters the first list `[2, 2]` and the second list `[2, 1, 3]`, and the output is `[2]`.

```

union.py - pizzaproject - Visual Studio Code - Insiders
File Edit Selection View Go Run Terminal Help
main2.py main2.py union.py x
javafolderforprogram > union.py > ...
38 n = int(input("Enter second list: "))
39
40 #print("\n")
41 for i in range(0, n):
42     print("Enter index", i, )
43     item = int(input())
44     number_list2.append(item)
45 for i in number_list2:
46     if i not in res2:
47         res2.append(i)
48 print(ob1.problem(res1,res2))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
2: Code
Enter index 1
2
[2, 2]
PS E:\nodejs\pizzaproject> python -u "e:\nodejs\pizzaproject\javafolderforprogram\union.py"
Enter first list 4
Enter index 0
1
Enter index 1
2
Enter index 2
2
Enter index 3
1
Enter second list 2
Enter index 0
2
Enter index 1
2
[2]
PS E:\nodejs\pizzaproject>
Python 3.9.2 64-bit 0 0
Tab Moves Focus Ln 48, Col 30 (1079 selected) Spaces: 4 UTF-8 CRLF Python Go Live
Type here to search
16:29 04-03-2021

```

case 2:

The screenshot shows a Visual Studio Code editor window with the title bar "union.py - pizzaproject - Visual Studio Code - Insiders". The editor has three tabs: "main2.py", "union.py", and "union.py". The active tab is "union.py", which contains the following Python code:

```

38 n = int(input("Enter second list: "))
39
40 #print("\n")
41 for i in range(0, n):
42     print("Enter index", i, )
43     item = int(input())
44     number_list2.append(item)
45 for i in number_list2:

```

The terminal output shows the execution of the script:

```

PS E:\nodejs\pizzaproject> python -u "E:\nodejs\pizzaproject\javafolderforprogram\union.py"
Enter first list 3
Enter index 0
4
Enter index 1
9
Enter index 2
5
Enter second list 5
Enter index 0
9
Enter index 1
4
Enter index 2
9
Enter index 3
8
Enter index 4
4
[9, 4]
PS E:\nodejs\pizzaproject>

```

The status bar at the bottom indicates the Python version is 3.9.2 64-bit, the workspace is "0 0 0", and the current file is "union.py" at line 48, column 30. The system tray shows the date and time as 04-03-2021, 16:30.

Question 3:

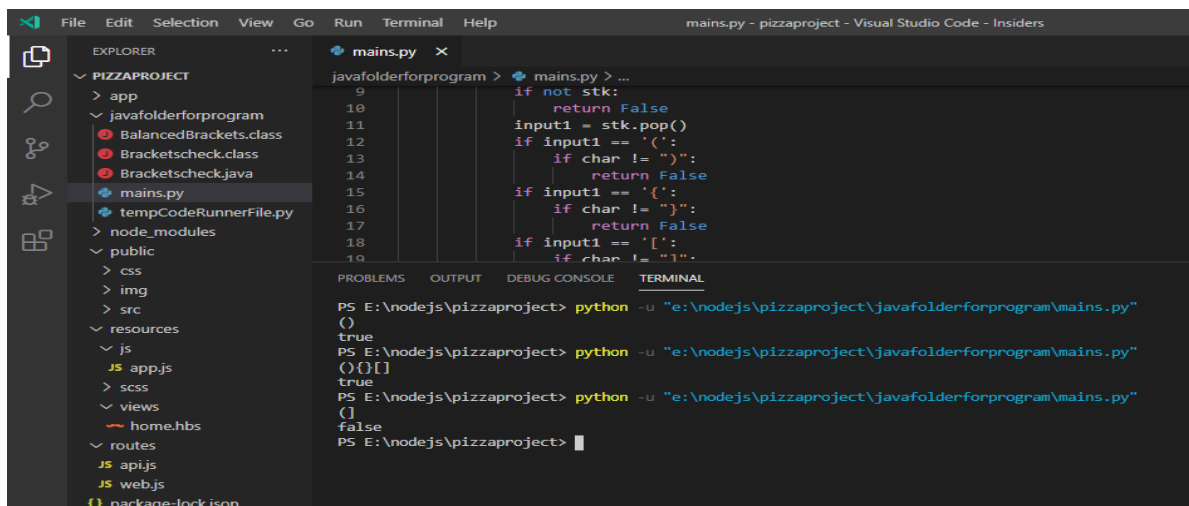
Given a string containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid.

```
def bracket(prtnum):
    stk = []
    for char in prtnum:
        if char in ["(", "{", "["]:
            stk.append(char)
        else:
            if not stk:
                return False
            input1 = stk.pop()
            if input1 == '(':
                if char != ")":
                    return False
            if input1 == '{':
                if char != "}":
                    return False
            if input1 == '[':
                if char != "]":
                    return False

    if stk:
        return False
    return True

if __name__ == "__main__":
    dev=input("")
    prtnum = dev
    if bracket(prtnum):
        print("true")
    else:
        print("false")
```

case 1 and case 2 both



```
mains.py - pizzaproject - Visual Studio Code - Insiders

EXPLORER
  PIZZAPROJECT
    > app
    > javafolderforprogram
      BalancedBrackets.class
      Bracketscheck.class
      Bracketscheck.java
      mains.py
      tempCodeRunnerFile.py
    > node_modules
    > public
      > css
      > img
      > src
    > resources
    > js
      JS app.js
    > scss
    > views
    > home.hbs
    > routes
      JS api.js
      JS web.js
    package-lock.json

mains.py
9
10
11
12
13
14
15
16
17
18
19

javafolderforprogram > mains.py > ...
9
10
11
12
13
14
15
16
17
18
19

if not stk:
    return False
input1 = stk.pop()
if input1 == '(':
    if char != ")":
        return False
if input1 == '{':
    if char != "}":
        return False
if input1 == '[':
    if char != "]":
        return False

if stk:
    return False
return True

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS E:\nodejs\pizzaproject> python -u "e:\nodejs\pizzaproject\javafolderforprogram\mains.py"
()
PS E:\nodejs\pizzaproject> python -u "e:\nodejs\pizzaproject\javafolderforprogram\mains.py"
(){}[]
true
PS E:\nodejs\pizzaproject> python -u "e:\nodejs\pizzaproject\javafolderforprogram\mains.py"
()
false
PS E:\nodejs\pizzaproject>
```

Question 4:

Given a non-empty array of integers, every element appears twice except for one. Find that single one.

```
def NonRepeatvalue(arr, n):

    mp={}

    for i in range(n):
        if arr[i] not in mp:
            mp[arr[i]]=0
        mp[arr[i]]+=1

    for x in mp:
        if (mp[x]== 1):
            print(x,end=" is single \n")

sub=[]
ar2=int(input("total input"))
for i in range(0,ar2):
    ar=int(input())
    sub.append(ar)

n = len(sub)
NonRepeatvalue(sub, n)
```

Case 1:

```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL 2: Code
PS E:\nodejs\pizzaproject>
PS E:\nodejs\pizzaproject> python -u "e:\nodejs\pizzaproject\javafolderforprogram\ques1.py"
total input 3
2
2
1
1 is single
PS E:\nodejs\pizzaproject> |
```

Case 2:

```
2
2
1
1 is single
PS E:\nodejs\pizzaproject> python -u "e:\nodejs\pizzaproject\javafolderforprogram\ques1.py"
total input 5
4
1
2
1
2
4 is single
PS E:\nodejs\pizzaproject> |
```