CEGEP VANIER COLLEGE
CENTRE FOR CONTINUING EDUCATION
Web Services
420-941-VA

Teacher: Samir Chebbine **Lab 3** **Oct 07, 2024**

Lab 3: Web Services using REST Implementation

Complete all these following programs in class. All *missing coding statements* are presented with explanation and in Presentation 3.

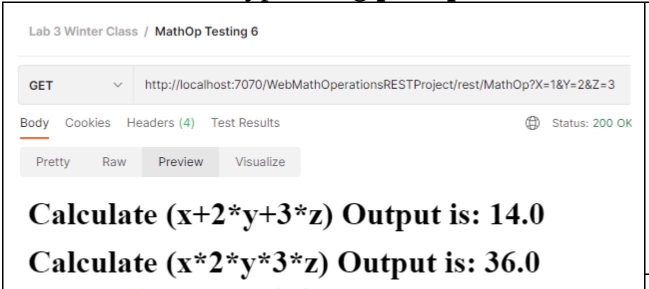
Create and Submit a Word file ***Lab3WebServicesYourName.doc*** which contains Answers of theory questions if any and output screenshots for every Java EE Project. Submit the Java projects too and submit the whole Lab 3 as compressed zip file

1. **Maven Dynamic Web Project: WebMathOperationsRESTProject**
   a) Create a new Dynamic Web project **WebMathOperationsRESTProject and convert it into Maven Project**. Check the output using Postman. Save your own screenshots.
   b) Add **Maven Project dependencies** in **pom.xml**. Create new package called **mathOperationsREST.**
   c) Deploy **WebMathOperationsRESTProject** within GalssFish Server.
   d) You need to develop a **Java class** called `MathOp`, which takes `x, y, z` as **private** `non static` members. The `MathOp` class contains the following method members:
      - Add a method called `calculateSum()` in `MathOp` class that returns `(x+2*y+3*z)`.
      - Add a method called `calculatePrd()` in `MathOp` class that returns `(x*2*y*3*z)`.
   e) Create a new REST Resource class **WebMathResource.java**

   1. Add a path URL mapping **("MathOp"**) to access REST resource using appropriate REST annotation and call the following methods **calculateHTMLOp()/displayXYZJSON()**.
   2. Add a method **calculateHTMLOp()** that returns a HTML media type using appropriate Java REST annotations.
   Add appropriate statements in **calculateHTMLOp()** using **query string parameters** `x, y, z` that calls implemented methods `calculateSum()/calculatePrd()` in `MathOp`.
   3. Add a method **displayXYZJSON()** that returns a JSON media type and instantiate an `object of MatOp class type`. Set its data attributes to (1, 2, 3).
   4. Add a new path URL mapping ("/listArray…") that calls a method **displayListZYZ()** that returns a HTML media type using appropriate Java REST annotations.
   -Add appropriate statements in **displayListZYZ ()** to instantiate a Java data structure **Array List** `of object of MatOp class type` to be referenced by (`listXYZ`). Add every component of Array List course object to the following values `(1,2,3)(4,5,6)(7,8,9)`.
   Skip through A`rray List of object` (`listXYZ`) and display its components as shown hereafter.
   5. Add a new path URL mapping ("/OpHashMap…") with path parameter x as search parameter to access REST resource searching into Hash Map.
   -Add a method **searchHashMapListZYZ()** using **path parameter** `x` and returns a JSON media type using appropriate Java REST annotations and will be fired upon using URL mapping ("/OpHashMap…").
   -Add appropriate statements in **searchHashMapListZYZ ()** to instantiate a data structure **HashMap** `of MathOp class type` to be referenced by (`opHashMap`) where hash map key represents `x (path parameter)` and value hash map of `MathOp class type`. Set every component of hash map to the following values:

`x = 1,y = 2,z = 3/ x = 4,y = 5,z = 6 / x = 7,y = 8,z = 9`

-Skip through `Hash Map collection` (`opHashMap`) and display the result of Hash Map search in JSON media type using **path parameter** `x`.

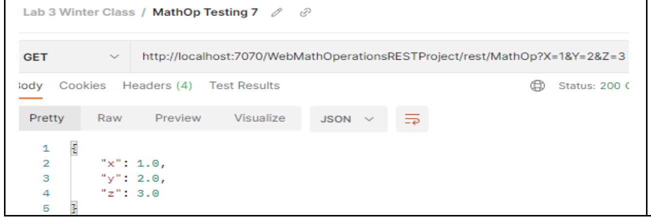

Lab 3 Winter Class / MathOp Testing 6

GET http://localhost:7070/WebMathOperationsRESTProject/rest/MathOp?X=1&Y=2&Z=3

Status: 200 OK

Calculate (x+2*y+3*z) Output is: 14.0

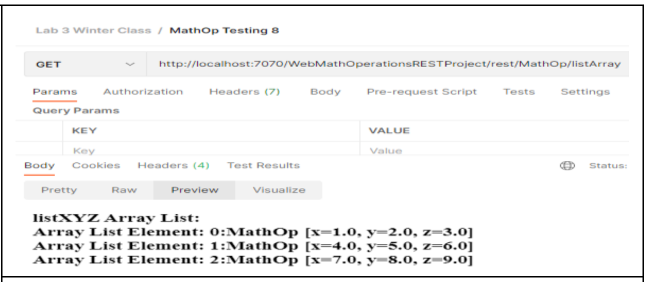
Calculate (x*2*y*3*z) Output is: 36.0


Lab 3 Winter Class / MathOp Testing 7

GET http://localhost:7070/WebMathOperationsRESTProject/rest/MathOp?X=1&Y=2&Z=3

Status: 200 (

```
{
    "x": 1.0,
    "y": 2.0,
    "z": 3.0
}
```

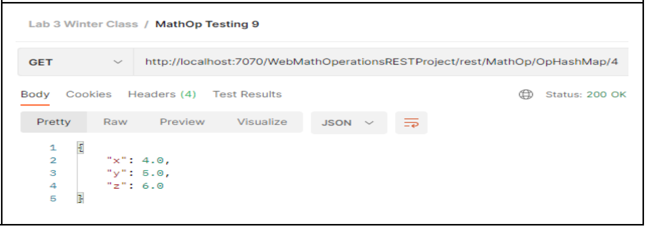

Lab 3 Winter Class / MathOp Testing 8

GET http://localhost:7070/WebMathOperationsRESTProject/rest/MathOp/listArray

Query Params

KEY	VALUE
Key	Value

```
listXYZ Array List:
Array List Element: 0:MathOp [x=1.0, y=2.0, z=3.0]
Array List Element: 1:MathOp [x=4.0, y=5.0, z=6.0]
Array List Element: 2:MathOp [x=7.0, y=8.0, z=9.0]
```


Lab 3 Winter Class / MathOp Testing 9

GET http://localhost:7070/WebMathOperationsRESTProject/rest/MathOp/OpHashMap/4

Status: 200 OK

```
{
    "x": 4.0,
    "y": 5.0,
    "z": 6.0
}
```

2. **Maven Dynamic Web Project: WebCarRESTProject**
   a) Create a new Dynamic Web project called **WebCarRESTProject and convert it into Maven Project**.
   b) Add **Maven Project dependencies** in **pom.xml**. Create new package called **webCarREST**.
   c) Deploy **WebCarRESTProject** within GalssFish Server to be executed by **Servlet class ServletContainer** specified in **web.xml**
   d) You need to develop a **Java class** called **`Car (see Block3)`**, which takes `vin, desc, price` as **private** `non static` members. The `Car` class contains the following method members:
   - Add a method called `discountPrice()` in Car class to calculate price discount of a given car after applying 10% discount on car price.

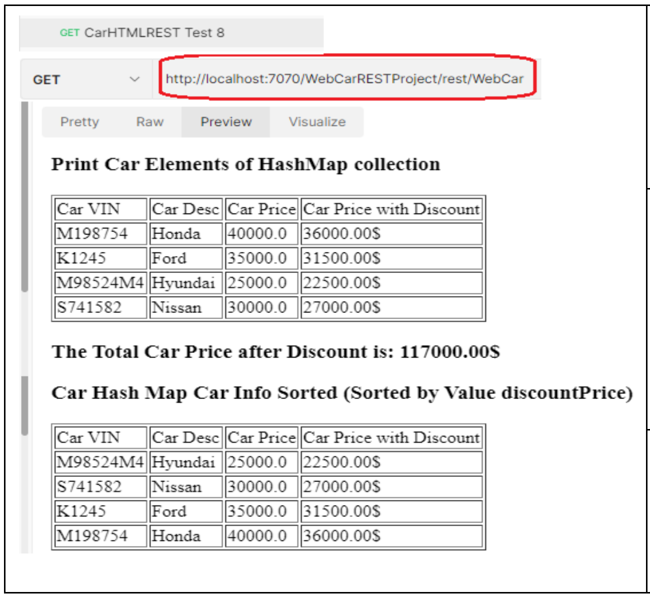
   e) Create a new REST Resource class **WebCarResource.java**

1. Add a path URL mapping ("WebCar") to access REST resource using appropriate Java REST annotation.
2. Add a method **displayHTMLCarInfo()** that returns a HTML media type using appropriate Java REST annotations.
3. Add appropriate statements in **displayHTMLCarInfo()** to instantiate a data structure **HashMap** `of Car class type` to be referenced by (`carHashMap`) where hash map key represents `vin` and value hash map of `Car class type`. Set every component of hash map to the following values read from text file **Car.in (use tab as separator)**:

```
vin = K1245,      desc = Ford,  price =35000/vin =M198754,  desc = Honda, price =40000
vin =M98524M4,desc = Hyundai,price =25000/vin =S741582,    desc = Nissan price =30000
```

4. Skip through `Hash Map collection` (`carHashMap`) and display its unsorted components and sorted components with respect to car price discount into web table respectively as shown hereafter **`(see Block3 for sorting)`**. Check output using Postman. Save your own screenshot.
5. Add a method **displayTextCarInfo ()** that returns the same output as plain TEXT media type using appropriate Java REST annotations. Check the output using Postman.
6. Add a method **displayJSONCarInfo ()** that returns elements of `Hash Map collection` (`carHashMap`) as JSON media type. Check the output using Postman.
7. Add a new path URL mapping ("/searchCar…") with path parameter vin as search string parameter to access REST resource searching into Hash Map. Save your own Postman screenshot in word document.
8. Add a method **searchJSONCarInfo(String car_vin)** that returns the result of Hash Map search in JSON media type and will be fired upon using URL mapping ("/searchCar…") as

shown hereafter.



GET CarHTMLREST Test 8

GET http://localhost:7070/WebCarRESTProject/rest/WebCar

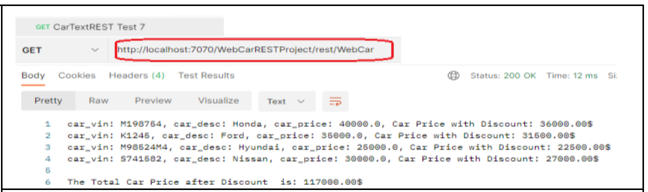
Print Car Elements of HashMap collection

Car VIN	Car Desc	Car Price	Car Price with Discount
M198754	Honda	40000.0	36000.00$
K1245	Ford	35000.0	31500.00$
M98524M4	Hyundai	25000.0	22500.00$
S741582	Nissan	30000.0	27000.00$

The Total Car Price after Discount is: 117000.00$

Car Hash Map Car Info Sorted (Sorted by Value discountPrice)

Car VIN	Car Desc	Car Price	Car Price with Discount
M98524M4	Hyundai	25000.0	22500.00$
S741582	Nissan	30000.0	27000.00$
K1245	Ford	35000.0	31500.00$
M198754	Honda	40000.0	36000.00$

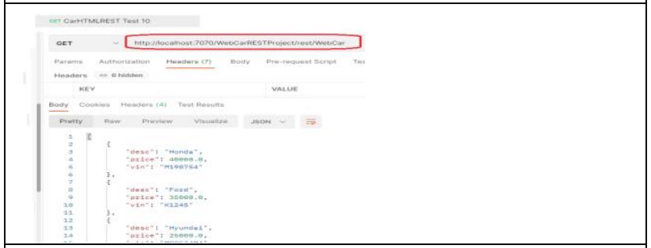


GET CarTextREST Test 7

GET http://localhost:7070/WebCarRESTProject/rest/WebCar

Status: 200 OK Time: 12 ms

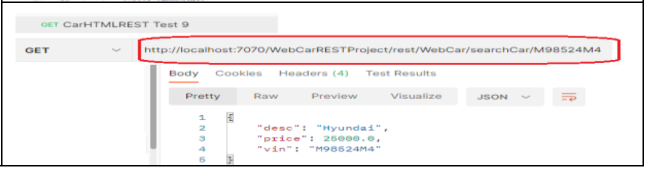
```
car_vin: M198764, car_desc: Honda, car_price: 40000.0, Car Price with Discount: 36000.00$
car_vin: K1245, car_desc: Ford, car_price: 35000.0, Car Price with Discount: 31500.00$
car_vin: M98524M4, car_desc: Hyundai, car_price: 25000.0, Car Price with Discount: 22500.00$
car_vin: S741582, car_desc: Nissan, car_price: 30000.0, Car Price with Discount: 27000.00$

The Total Car Price after Discount  is: 117000.00$
```



GET CarHTMLREST Test 10

GET http://localhost:7070/WebCarRESTProject/rest/WebCar

```
[
    {
        "desc": "Honda",
        "price": 40000.0,
        "vin": "M198764"
    },
    {
        "desc": "Ford",
        "price": 35000.0,
        "vin": "K1245"
    },
    {
        "desc": "Hyundai",
        "price": 25000.0,
```



GET CarHTMLREST Test 9

GET http://localhost:7070/WebCarRESTProject/rest/WebCar/searchCar/M98524M4

```
{
    "desc": "Hyundai",
    "price": 25000.0,
    "vin": "M98524M4"
}
```