Assignment B07

- Title: Encode & decode JSON objects.

- Problem Statement
  Encode & decode JSON objects using JAVA / Perl / PHP / Python / Ruby.

- Objectives:
  - To encode & decode JSON objects.
  - Performing read & write operation using JAVA & mongodb connectivity.

- Outcomes:
  We will be able to understand java and mongodb connectivity by performing read & write functions using encoding & decoding of JSON objects.

- S/w & H/w Requirements:
  MongoDB, 64-bits Windows OS, eclipse, Jar files, keyboard, mouse.

- Theory:
  JSON stands for Javascript object Notation, it is quick & easy to parse & generate.

- JSON like XML is a text based format that's easy to write/understand for both humans & computers, but unlike XML, JSON data structures occupy less bandwidth than their XML versions.

- This is defined as a collection of key/value pairs, each object begins with a left curly bracket '{'. Multiple key/value pairs are separated by comma ','

- This is defined as an ordered list of values. An array begins with a left bracket '[' and ends with a ']'. Values separated by ','

- In JSON keys are always strings while values can be a string number, true or false, null or object of an array. Strings must be enclosed in '"' & can contain escape characters such as '\n', '\t', etc.

- Encoding JSON data in JAVA

- Ask for all values
- Create a JSON object
- Insert values in JSON object by 'put' function.
- Create a connection with mongodb by including jar files

Syntax:

```
MongoClient mongo = MongoClients.create;
MongoDatabase db = mongo.getDatabase(
                   "mydatabase")
```

- Decoding JSON data in Java:

- Create a MongoCursor & iterate over Mongo collection.
- Till the cursor has next values, parse the 'Jsonvalue to JSON object'
- Read all the values in JSON object & print it.
- close the cursor.

- Conclusion:

Therefore, we successfully implemented encoding & decoding of JSON objects.

```
import com.mongodb.client.*;
import com.mysql.cj.xdevapi.JsonValue;
import java.util.Iterator;
import java.util.Scanner;
import com.mongodb.*;
import org.bson.Document;
import org.json.simple.JSONValue;
import org.json.simple.JSONObject;
import org.json.simple.JSONArray;
public class B7
{
public static void main(String[] args) {
try {
MongoClient mongo = MongoClients.create("mongodb://localhost:27017");
MongoDatabase db = mongo.getDatabase("b7");
MongoCollection<Document> collection = db.getCollection("game");
Scanner sc = new Scanner(System.in);
JSONObject obj = new JSONObject();
JSONArray jarr = new JSONArray();
String uname, badge;
int u_number;
System.out.print("Enter Name :");
uname = sc.nextLine();
System.out.print("Enter User Number : ");
u_number = sc.nextInt();
sc.nextLine();
while (true) {
System.out.print("Enter badge:");
badge = sc.nextLine();
jarr.add(badge);
System.out.print("Enter 'Y' to add more");
badge = sc.nextLine();
if (!badge.equals("Y"))
break;
}
//encoding json
obj.put("name", uname);
obj.put("user_number", u_number);
obj.put("badges", jarr);
Document document;
document = Document.parse(obj.toString());
collection.insertOne(document);
//System.out.print(obj);
for (Document doc : collection.find()) {
// System.out.print(doc.toJson().toString());
Object o1 = JSONValue.parse(doc.toJson());
JSONObject jsonObject = (JSONObject) o1;
//decoding json object
System.out.printf("%15s %15s %30s", jsonObject.get("name").toString(),
```

```
jsonObject.get("user_number"), jsonObject.get("badges"));
System.out.print('\n');
}
} catch (Exception e) {
System.out.println(e);
}
}
}
```

OUTPUT - Enter Name :Prachi
Enter User Number : 6
Enter badge:Rookie
Enter 'Y' to add more
n
Prathamesh 1 ["Rookie"]
Sakshi 2 ["Rookie"]
sparsh 3 ["Rookie","Veteran"]
Ronak 4 ["Rookie", “Veteran”]
Prajwal 5 ["Rookie"]
Atharv 6 ["Rookie"]
Process finished with exit code 0