DAY-1 TASK

1. **To read:**
    a. https://stackoverflow.com/questions/1517582/what-is-the-difference-between-statically-typed-and-dynamically-typed-languages
    b. https://stackoverflow.com/questions/17253545/scripting-language-vs-programming-language
    c. https://cs.lmu.edu/~ray/notes/paradigms/

2. **Write a blog on Difference between HTTP1.1 vs HTTP2**

Major performance comparison between Http 1.1 and Http 2 are as follows:-

Http 1.1	Http2
a. **Web traffic** It provides faster delivery of web pages and reduces web traffic if you compare it to Http 1.0. However, there is an increased risk of network congestion.	The Http 2 version utilizes multiplexing and server pushes to effectively reduce the page load time by a greater margin along with being sensitive to network delays.
b. **Performance optimization** Some of the optimizations used in the Http 1.1 version are sprinting, inlining, domain sharding, and concatenating.	This protocol version removes the need for unnecessary optimization hacks.
c. **Authentication mechanism** Protocol Http 1.1 is much more secure than Http 1.0 because it uses digest authentication and NTLM authentication.	The security concern in Http 2 version is also good and almost the same as Http 1.1. Rather Http 2 is better equipped to deal with security threats because of the new features it brings. For example, new TLS features like connection error of type inadequate security.
d. **Security** In this version, SSL or secure sockets layer is not required but recommended. Digest authentication is an improvement over Http 1.0 which is now being used in Http 1.1. Moreover, Https	In Http 2 protocol, security is not at all recommended. It is because the security is encrypted since almost all clients demand traffic to be encrypted. It also has minimum standards and minimum key size for encryption.

uses SSL/TLS for secure encrypted communication.	

3. Write a blog about objects and its internal representation in Javascript.

Objects are the representation of real-world entities in any language representing things by defining its properties along with their values. In Javascript, objects may be defined as an unordered collection of related data, of primitive or reference types, in the form of “key: value” pairs.

Object literal

object literal is a comma-separated list of name-value pairs wrapped in curly braces. Object literals encapsulate data, enclosing it in a tidy package.

```
var car={id:1 , name:'abc' , display:function() }
```

from the above example property values can be of any data type, including array literals, functions, nested object literals, or primitive data type.

Object.create()

The method creates a new object, using an existing object as the prototype of the newly created object.

using the object literal example as prototype-

```
var car2 = Object.create(car);
car.id=2;
car.name='xyz';
```

Object constructor

Useful when we require to create multiple objects of similar type. In this case, a constructor (kind of blueprint) is created and multiple objects can be initialized using the new keyword using the constructor as a wrapper for the newly created objects.

construction function-

```
function Person(name, age, eye) {
this.Name = name;
this.age = age;
this.eyeColor = eye;
}
```

creating objects using constructor-

```
var p1= new Person(“John”, 50, “blue”);
var p2= new Person(“Sally”, 48, “green”);
```

Object.assign()

It is used to copy the values and properties from one or more source objects to a target object. It invokes getters and setters since it uses both [[Get]] on the source and [[Set]] on the target.

Here is an example where properties from three source objects are getting assigned to a new object using Object.assign()

```
Input : var obj1 = { a: 10 };
var obj2 = { b: 20 };
var obj3 = { c: 30 };
var new_obj = Object.assign(o1, o2, o3);
console.log(new_obj);
Output : Object { a: 10, b: 20, c: 30 }
```

Object.fromEntries

This method transforms a list of key-value pairs into an object.

```
const entries = new car([
['id', 4],
['color', 'blue']
]);
const car1= Object.fromEntries(entries);
console.log(car1);
output: Object { id: 4, color: 'blue'}
```

Unlike other object-oriented programming languages, javascript doesn't have classes instead of that javascript is a prototype-based language allowing all the functionalities as in other class-based programming languages like JavaScript allows you to create hierarchies of objects and to have the inheritance of properties and their values and all this is done mainly using the constructor functions.

4. codekata practice