Objects Practice Sheet

Question 1: Create a simple object

Question: Ek simple object create karo jo ek person ka name aur age store karta ho.

```
let person = {
   name: "John",
   age: 30
};
```

Explanation: Humne ek object person create kiya jisme name aur age properties hain.

Question 2: Access object properties

Question: Ek object ki properties ko access karo aur print karo.

```
console.log(person.name); // "John"
console.log(person.age);  // 30
```

Explanation: Dot notation use karke hum object ki properties ko access karte hain.

Question 3: Update object properties

Question: Ek object ki properties ko update karo.

```
person.name = "Jane";
console.log(person.name); // "Jane"
```

Explanation: Humne name property ko update kiya aur new value “Jane” set ki.

Question 4: Add new property to an object

Question: Ek naye property ko ek existing object mein add karo.

```
person.city = "New York";
console.log(person.city); // "New York"
```

Explanation: Humne city property ko add kiya aur uski value “New York” set ki.

Question 5: Delete a property from an object

Question: Ek property ko delete karo ek object se.

```
delete person.age;

console.log(person.age); // undefined
```

Explanation: delete operator use karke humne age property ko delete kar diya.

Question 6: Check if a property exists in an object

Question: Check karo ki ek specific property object mein exist karti hai ya nahi.

```
console.log('name' in person); // true

console.log('age' in person);  // false
```

Explanation: in operator use karke check kiya ki name property object mein exist karti hai ya nahi.

Question 7: Loop through object properties

Question: Object ki sabhi properties ko loop karke print karo.

```
for (let key in person) {

   console.log(key + ": " + person[key]);

}

// Output:

// name: Jane

// city: New York
```

Explanation: for...in loop use karke humne sabhi properties aur unki values ko print kiya.

Question 8: Nested objects

Question: Ek nested object create karo aur uski nested properties ko access karo.

```
let employee = {

   name: "Alice",
  address: {
    city: "Wonderland",
    zip: 12345
  }
};
console.log(employee.address.city); // "Wonderland"
```

Explanation: Humne ek nested object address create kiya aur uski city property ko access kiya.

Question 9: Merge two objects

Question: Do objects ko merge karo.

```
let obj1 = { a: 1, b: 2 };
let obj2 = { b: 3, c: 4 };
let mergedObj = Object.assign({}, obj1, obj2);
console.log(mergedObj); // { a: 1, b: 3, c: 4 }
```

Explanation: Object.assign method use karke do objects ko merge kiya.

Question 10: Copy an object

Question: Ek object ko copy karo.

```
let original = { x: 10, y: 20 };
let copy = Object.assign({}, original);
console.log(copy); // { x: 10, y: 20 }
```

Explanation: Object.assign method use karke humne original object ki copy banayi.

Question 11: Object destructuring

Question: Object destructuring use karke properties ko variables mein assign karo.

```
let { name, city } = person;
```

```
console.log(name); // "Jane"

console.log(city); // "New York"
```

Explanation: Object destructuring se person object ki properties ko alag-alag variables mein assign kiya.

Question 12: Computed property names

Question: Computed property names ka use karke ek object create karo.

```
let prop = "age";

let user = {

  [prop]: 25

};

console.log(user.age); // 25
```

Explanation: Computed property names se prop variable ki value ko property name banaya.

Question 13: Function inside object

Question: Ek function ko ek object ke andar define karo aur usse call karo.

```
let car = {

  brand: "Toyota",

  getBrand: function() {

    return this.brand;

  }

};

console.log(car.getBrand()); // "Toyota"
```

Explanation: car object mein ek method getBrand define kiya aur use call kiya.

Question 14: this keyword in object method

Question: this keyword ka use karke ek object method mein property access karo.

```
let user = {
    name: "John",
    greet: function() {
        console.log("Hello, " + this.name);
    }
};
user.greet(); // "Hello, John"
```

Explanation: this keyword se current object ki name property access ki.

Question 15: Arrow function as object method

Question: Arrow function ko ek object method ke tarah define karo aur call karo.

```
let person = {
    name: "Emma",
    greet: () => {
        console.log("Hi, " + this.name);
    }
};
person.greet(); // "Hi, undefined"
```

Explanation: Arrow functions this context ko lexically bind karte hain,
isliye name undefined mila.

Question 16: Object.keys() method

Question: Object.keys method ka use karke object ke keys ko array mein convert karo.

```
let keys = Object.keys(person);
console.log(keys); // ["name", "city"]
```

Explanation: Object.keys method se object ke keys ki array banayi.

Question 17: Object.values() method

Question: Object.values method ka use karke object ke values ko array mein convert karo.

```
let values = Object.values(person);

console.log(values); // ["Jane", "New York"]
```

Explanation: Object.values method se object ke values ki array banayi.

Question 18: Object.entries() method

Question: Object.entries method ka use karke object ke key-value pairs ko array of arrays mein convert karo.

```
let entries = Object.entries(person);

console.log(entries); // [["name", "Jane"], ["city", "New York"]]
```

Explanation: Object.entries method se object ke key-value pairs ki array banayi.

Question 19: Using a constructor function to create objects

Question: Ek constructor function define karo aur uska use karke objects create karo.

```
function Person(name, age) {

   this.name = name;

   this.age = age;

}


let person1 = new Person("Alice", 25);

let person2 = new Person("Bob", 30);


console.log(person1); // Person { name: 'Alice', age: 25 }

console.log(person2); // Person { name: 'Bob', age: 30 }
```

Explanation: Constructor function Person define kiya aur new keyword se objects create kiye.

Question 20: Prototype property

Question: Ek object ki prototype property set karo aur uska use karo.

```
Person.prototype.greet = function() {
  console.log("Hello, " + this.name);
};

person1.greet(); // "Hello, Alice"
```

Explanation: Prototype property se greet method ko sabhi Person instances mein add kiya.

Question 21: Object.create() method

Question: Object.create method ka use karke ek new object create karo.

```
let proto = { greet() { console.log("Hello!"); } };
let obj = Object.create(proto);
obj.greet(); // "Hello!"
```

Explanation: Object.create se proto ko prototype banake new object create kiya.

Question 22: Object.freeze() method

Question: Object.freeze ka use karke ek object ko freeze karo aur uske properties ko change karne ki koshish karo.

```
let frozenObj = Object.freeze({ a: 1 });
frozenObj.a = 2;
console.log(frozenObj.a); // 1
```

Explanation: Object.freeze se object ko immutable banaya, changes apply nahi hue.

Question 23: Object.seal() method

Question: Object.seal ka use karke ek object ko seal karo aur usme naye properties add karne ki koshish karo.

```
let sealedObj = Object.seal({ b: 2 });

sealedObj.b = 3;

sealedObj.c = 4;

console.log(sealedObj.b); // 3

console.log(sealedObj.c); // undefined
```

Explanation: Object.seal se object ko seal kiya, properties update hui lekin naye properties add nahi hui.

Question 24: Object.assign() method

Question: Object.assign ka use karke ek object ki properties ko copy karo.

```
let


 source = { x: 10, y: 20 };

let target = {};

Object.assign(target, source);

console.log(target); // { x: 10, y: 20 }
```

Explanation: Object.assign se source object ki properties ko target object mein copy kiya.

Question 25: Creating methods in an object

Question: Ek object mein methods define karo aur call karo.

```
let calculator = {

  add: function(a, b) {

    return a + b;

  },
    subtract: function(a, b) {
        return a - b;
    }
};
console.log(calculator.add(5, 3)); // 8
console.log(calculator.subtract(5, 3)); // 2
```

Explanation: calculator object mein add aur subtract methods define kiye aur call kiye.

Question 26: Object property shorthand

Question: Object property shorthand ka use karke ek object create karo.

```
let name = "Charlie";
let age = 28;
let user = { name, age };
console.log(user); // { name: 'Charlie', age: 28 }
```

Explanation: Object property shorthand se variables name aur age ko direct properties banaya.

Question 27: Dynamic property names

Question: Dynamic property names ka use karke ek object create karo.

```
let key = "username";
let value = "john_doe";
let user = {
    [key]: value
};
console.log(user); // { username: 'john_doe' }
```

Explanation: Dynamic property names se key variable ki value ko property name banaya.

Question 28: Object property value shorthand

Question: Object property value shorthand ka use karke ek object create karo.

```
let user = {
    name,
    age
};
console.log(user); // { name: 'Charlie', age: 28 }
```

Explanation: Property value shorthand se existing variables ko direct properties banaya.

Question 29: Spread operator with objects

Question: Spread operator ka use karke do objects ko merge karo.

```
let obj1 = { a: 1, b: 2 };
let obj2 = { c: 3, d: 4 };
let mergedObj = { ...obj1, ...obj2 };
console.log(mergedObj); // { a: 1, b: 2, c: 3, d: 4 }
```

Explanation: Spread operator ... se obj1 aur obj2 ko merge kiya.

Question 30: Object with a method using this

Question: Ek object mein method define karo jo this ka use karke object properties access kare.

```
let user = {
    name: "Alice",
    greet: function() {
        console.log("Hello, " + this.name);
    }
};
user.greet(); // "Hello, Alice"
```

Explanation: greet method mein this keyword use karke name property access ki.

Question 31: Factory function to create objects

Question: Ek factory function define karo jo objects create kare.

```
function createUser(name, age) {
    return {
        name,
        age,
        greet() {
            console.log("Hi, " + name);
        }
    };
}

let user1 = createUser("Bob", 25);
let user2 = createUser("Carol", 30);

console.log(user1); // { name: 'Bob', age: 25, greet: [Function: greet] }
console.log(user2); // { name: 'Carol', age: 30, greet: [Function: greet] }
```

Explanation: Factory function createUser define kiya jo new user objects create karta hai.

Question 32: Chaining object methods

Question: Ek object define karo jisme methods chaining support karte ho.

```
let calculator = {
    result: 0,
    add(value) {
        this.result += value;
        return this;
    },
    subtract(value) {
        this.result -= value;
        return this;
    }
};

calculator.add(5).subtract(3);
console.log(calculator.result); // 2
```

Explanation: Methods add aur subtract ko chaining support karne ke liye return this kiya.

Question 33: Using Object.defineProperty

Question: Object.defineProperty ka use karke ek object property define karo with getter aur setter.

```
let user = {};
Object.defineProperty(user, "fullName", {
    get() {
        return this.firstName + " " + this.lastName;
    },
    set(value) {
        [this.firstName, this.lastName] = value.split(" ");
    }
});


user.fullName = "Alice Wonderland";
console.log(user.firstName); // "Alice"
console.log(user.lastName);  // "Wonderland"
console.log(user.fullName);  // "Alice Wonderland"
```

Explanation: Object.defineProperty se fullName property define ki jo getter aur setter use karti hai.

Question 34: Converting an object to JSON

Question: Ek object ko JSON string mein convert karo.

```
let person = { name: "Alice", age: 25 };
let jsonString = JSON.stringify(person);
console.log(jsonString); // '{"name":"Alice","age":25}'
```

Explanation: JSON.stringify method se object ko JSON string mein convert kiya.

Question 35: Parsing a JSON string to an object

Question: Ek JSON string ko object mein parse karo.

```
let jsonString = '{"name":"Alice","age":25}';
let person = JSON.parse(jsonString);
console.log(person); // { name: 'Alice', age: 25 }
```

Explanation: JSON.parse method se JSON string ko object mein convert kiya.

Question 36: Object.entries() with for…of loop

Question: Object.entries aur for...of loop ka use karke object ke key-value pairs ko loop karo.

```
let person = { name: "Alice", age: 25 };

for (let [key, value] of Object.entries(person)) {

   console.log(`${key}: ${value}`);

}

// Output:

// name: Alice

// age: 25
```

Explanation: Object.entries se key-value pairs ko array mein convert kiya aur for...of loop use karke iterate kiya.

Question 37: Object.fromEntries() method

Question: Object.fromEntries method ka use karke key-value pairs ki array ko object mein convert karo.

```
let entries = [['name', 'Alice'], ['age', 25]];

let person = Object.fromEntries(entries);

console.log(person); // { name: 'Alice', age: 25 }
```

Explanation: Object.fromEntries method se key-value pairs ki array ko object mein convert kiya.

Question 38: Using a symbol as a key

Question: Ek symbol ko ek object key ke tarah use karo.

```
let sym = Symbol("id");

let obj = {

   [sym]: 123

};

console.log(obj[sym]); // 123
```

Explanation: Symbol sym ko object key ke tarah use kiya aur uski value access ki.

Question 39: Object literal enhancement

Question: Object literal enhancement ka use karke ek object create karo.

```
let x = 10, y = 20;

let obj = { x, y, sum() { return x + y; } };

console.log(obj); // { x: 10, y: 20, sum: [Function: sum] }

console.log(obj.sum()); // 30
```

Explanation: Object literal enhancement se existing variables aur method ko direct object mein add kiya.

Question 40: Object destructuring with default values

Question: Object destructuring ka use karke default values assign karo.

```
let person = { name: "Alice" };

let { name, age = 30 } = person;

console.log(name); // "Alice"

console.log(age);  // 30
```

Explanation: Object destructuring se age property ki default value 30 assign ki jab property exist nahi karti.

Question 41: Object destructuring with rest operator

Question: Object destructuring ka use karke rest operator ko implement karo.

```
let person = { name: "Alice", age: 25, city: "Wonderland" };

let { name, ...rest } = person;

console.log(name); // "Alice"

console.log(rest); // { age: 25, city: "Wonderland" }
```

Explanation: Rest operator ...rest se baki properties ko ek naya object rest mein assign kiya.

Question 42: Checking if a key exists using hasOwnProperty

Question: hasOwnProperty method ka use karke check karo ki ek key object mein exist karti hai ya nahi.

```
let person = { name: "Alice", age: 25 };

console.log(person.hasOwnProperty("name")); // true

console.log(person.hasOwnProperty("city")); // false
```

Explanation: hasOwnProperty method se check kiya ki name aur city properties object mein exist karti hain ya nahi.

Question 43: Enumerate properties with Object.getOwnPropertyNames

Question: Object.getOwnPropertyNames method

ka use karke object ki sabhi properties ko enumerate karo.

```
let person = { name: "Alice", age: 25 };

let properties = Object.getOwnPropertyNames(person);

console.log(properties); // ["name", "age"]
```

Explanation: Object.getOwnPropertyNames method se object ki sabhi properties ki array banayi.

Question 44: Object.preventExtensions()

Question: Object.preventExtensions ka use karke ek object ko non-extensible banao aur naye properties add karne ki koshish karo.

```
let obj = { a: 1 };

Object.preventExtensions(obj);

obj.b = 2;

console.log(obj.b); // undefined
```

Explanation: Object.preventExtensions se object ko non-extensible banaya, naye properties add nahi hui.

Question 45: Object.isExtensible()

Question: Object.isExtensible ka use karke check karo ki ek object extensible hai ya nahi.

```
let obj = { a: 1 };

console.log(Object.isExtensible(obj)); // true

Object.preventExtensions(obj);

console.log(Object.isExtensible(obj)); // false
```

Explanation: Object.isExtensible method se check kiya ki object extensible hai ya nahi.

Question 46: Setting a prototype with Object.setPrototypeOf

Question: Object.setPrototypeOf ka use karke ek object ka prototype set karo.

```
let animal = { eats: true };

let rabbit = {};

Object.setPrototypeOf(rabbit, animal);

console.log(rabbit.eats); // true
```

Explanation: Object.setPrototypeOf se rabbit object ka prototype animal set kiya.

Question 47: Using a getter to compute a property

Question: Ek getter define karo jo ek property ko compute kare.

```
let rectangle = {

  length: 10,

  width: 5,

  get area() {

    return this.length * this.width;

  }

};

console.log(rectangle.area); // 50
```

Explanation: Getter area define kiya jo length aur width properties se area compute karta hai.

Question 48: Using a setter to set a property

Question: Ek setter define karo jo ek property ko set kare.

```
let person = {
   firstName: "John",
   lastName: "Doe",
   set fullName(value) {
      [this.firstName, this.lastName] = value.split(" ");
   }
};
person.fullName = "Alice Wonderland";
console.log(person.firstName); // "Alice"
console.log(person.lastName);  // "Wonderland"
```

Explanation: Setter fullName define kiya jo firstName aur lastName properties ko set karta hai.

Question 49: Object property flags

Question: Object.defineProperty ka use karke ek property ko non-enumerable banao.

```
let user = {};
Object.defineProperty(user, "name", {
   value: "Alice",
   enumerable: false
});
console.log(Object.keys(user)); // []
console.log(user.name); // "Alice"
```

Explanation: Object.defineProperty se name property ko non-enumerable banaya.

Question 50: Creating an object with null prototype

Question: Ek object create karo jiska prototype null ho.

```
let obj = Object.create(null);

console.log(Object.getPrototypeOf(obj)); // null
```

Explanation: Object.create(null) se ek object create kiya jiska prototype null hai.

Yeh 50 questions aapko JavaScript objects ke concepts ko samajhne aur practice karne mein madad karenge.