## Easy Level

**Question:**

```
 const person = { name: "Alice", age: 25 };
console.log(person.name);
```

1. **Output:**

**Question:**

```
 const obj = {};
obj.name = "John";
console.log(obj);
```

2. **Output:**

**Question:**

```
 const person = { name: "Bob", age: 30 };
delete person.age;
console.log(person);
```

3. **Output:**

**Question:**

```
 const obj = { x: 10, y: 20 };
console.log("z" in obj);
```

4. **Output:**

**Question:**

```
 const obj = { a: 1, b: 2 };
console.log(Object.keys(obj));
```

5. **Output:**

**Question:**

```
 const obj = { a: 10 };
obj.b = 20;
console.log(obj.a + obj.b);
```

6. **Output:**

**Question:**

```
 const person = { name: "Sam" };
person.name = "Max";
console.log(person.name);
```

7. **Output:**

**Question:**

```
 const obj = { x: 5, y: 10 };
console.log(obj.hasOwnProperty("x"));
```

8. **Output:**

**Question:**

```
 const person = { name: "Alice" };
console.log(person.age || "Not defined");
```

9. **Output:**

**Question:**

```
 const obj = { a: 5, b: 10, c: 15 };
console.log(Object.values(obj));
```

10. **Output:**

**Question:**

```
 const obj = { a: 1 };
console.log(obj.b === undefined);
```

11. **Output:**

**Question:**

```
 const obj = { a: 1, b: 2 };
console.log(Object.entries(obj));
```

12. **Output:**

**Question:**

```
 const obj = { x: 10, y: 20 };
obj.z = 30;
console.log(Object.keys(obj).length);
```

13. **Output:**

**Question:**

```
 const person = { name: "Jane", age: 22 };
console.log(person.gender ?? "Unknown");
```

14. **Output:**

**Question:**

```
 const obj = { x: 10 };
Object.freeze(obj);
obj.y = 20;
console.log(obj);
```

15. **Output:**

**Question:**

```
 const obj = { a: 1, b: 2 };
console.log("a" in obj);
```

16. **Output:**

**Question:**

```
 const obj = {};
console.log(obj.toString());
```

17. **Output:**

**Question:**

```
 const obj = { a: 5, b: 10 };
console.log(obj.a + obj["b"]);
```

18. **Output:**

**Question:**

```
 const obj = { a: 1, b: 2 };
obj.a = obj.a + 5;
console.log(obj.a);
```

19. **Output:**

**Question:**

```
 const obj = { a: 1, b: undefined };
console.log(obj.b ?? "Not available");
```

20. **Output:**

---

## Medium Level

**Question:**

```
 const obj = { a: { b: 10 } };
console.log(obj.a.b);
```

21. **Output:**

**Question:**

```
 const person = { name: "John", age: 30 };
delete person.name;
console.log(person);
```

22. **Output:**

**Question:**

```
 const obj = { a: 5 };
Object.seal(obj);
```

```
obj.b = 10;
console.log(obj);
```

23. **Output:**

**Question:**

```
 const obj = { a: 1, b: { c: 2 } };
console.log(obj.b.c);
```

24. **Output:**

**Question:**

```
 const obj = { x: 10, y: 20 };
console.log(Object.entries(obj));
```

25. **Output:**

**Question:**

```
 const obj = { a: 5, b: 10 };
for (const key in obj) {
  console.log(key);
}
```

26. **Output:**

**Question:**

```
 const obj = { x: 1, y: 2 };
for (const key in obj) {
  console.log(obj[key]);
}
```

27. **Output:**

**Question:**

```
 const obj = { a: 5, b: 10 };
console.log(Object.keys(obj).length);
```

28. **Output:**

**Question:**

```
 const obj = { a: 1, b: 2, c: 3 };
console.log(Object.values(obj).reduce((sum, value) => sum + value));
```

29. **Output:**

**Question:**

```
 const obj1 = { a: 1 };
const obj2 = { b: 2 };
const merged = { ...obj1, ...obj2 };
console.log(merged);
```

30. **Output:**

**Question:**

```
 const obj = { x: 10, y: 20 };
obj.z = obj.x + obj.y;
console.log(obj.z);
```

31. **Output:**

**Question:**

```
 const obj = { a: 1, b: 2 };
const copy = { ...obj };
copy.c = 3;
console.log(copy);
```

32. **Output:**

**Question:**

```
 const obj = { x: 1, y: 2 };
delete obj.x;
console.log("x" in obj);
```

33. **Output:**

**Question:**

```
 const obj = { a: 1, b: 2, c: 3 };
```

```
console.log(Object.keys(obj).join(", "));
```

34. **Output:**

**Question:**

```
 const obj = { a: 5, b: 10 };
console.log(Object.values(obj).filter((val) => val > 5));
```

35. **Output:**

**Question:**

```
 const obj = { x: 5 };
console.log(obj["x"] === obj.x);
```

36. **Output:**

**Question:**

```
 const obj = { x: 1, y: 2 };
const sum = Object.values(obj).reduce((total, val) => total + val, 0);
console.log(sum);
```

37. **Output:**

**Question:**

```
 const obj = { a: 1, b: 2 };
const hasC = obj.hasOwnProperty("c");
console.log(hasC);
```

38. **Output:**

**Question:**

```
 const obj = { x: 10 };
Object.seal(obj);
obj.x = 20;
console.log(obj.x);
```

39. **Output:**

**Question:**

```
 const obj = { a: 10 };
const desc = Object.getOwnPropertyDescriptor(obj, "a");
console.log(desc.writable);
```

40. **Output:**

**Question:**

```
 const obj = { a: 5, b: 10 };
console.log(Object.isFrozen(obj));
```

41. **Output:**

**Question:**

```
 const obj = { a: 5, b: 10 };
const keys = Object.keys(obj).map((key) => key.toUpperCase());
console.log(keys);
```

42. **Output:**

**Question:**

```
 const obj = { a: 1, b: 2 };
const json = JSON.stringify(obj);
console.log(json);
```

43. **Output:**

**Question:**

```
 const obj = { x: 10 };
Object.freeze(obj);
obj.x = 20;
console.log(obj.x);
```

44. **Output:**

**Question:**

```
 const obj = { a: 1 };
const clone = JSON.parse(JSON.stringify(obj));
```

```
console.log(clone);
```

45. **Output:**

**Question:**

```
 const obj = { x: 1, y: 2 };
const entries = Object.entries(obj);
console.log(entries.length);
```

46. **Output:**

**Question:**

```
 const obj = { x: 10, y: 20, z: 30 };
const result = Object.keys(obj).map((key) => obj[key] * 2);
console.log(result);
```

47. **Output:**

**Question:**

```
 const obj = { x: 10, y: 20 };
obj.z = obj.y;
console.log(obj.z === obj.y);
```

48. **Output:**

**Question:**

```
 const obj = { x: 1, y: 2 };
const copy = Object.assign({}, obj);
console.log(copy);
```

49. **Output:**

**Question:**

```
 const obj = { x: 10, y: 20 };
obj.x += 5;
console.log(obj);
```

50. **Output:**

**Question:**

```
 const obj = { a: 5 };
Object.defineProperty(obj, "b", { value: 10, writable: false });
obj.b = 20;
console.log(obj.b);
```

51. **Output:**

**Question:**

```
 const obj = { x: 1, y: { z: 2 } };
const clone = { ...obj };
clone.y.z = 10;
console.log(obj.y.z);
```

52. **Output:**

**Question:**

```
 const obj = { x: 5, y: 10 };
Object.seal(obj);
delete obj.x;
console.log(obj);
```

53. **Output:**

**Question:**

```
 const obj = { x: 5, y: 10 };
Object.freeze(obj);
obj.z = 15;
console.log(Object.keys(obj));
```

54. **Output:**

**Question:**

```
 const obj = { a: 1, b: 2 };
Object.defineProperty(obj, "sum", {
  get() {
    return this.a + this.b;
  },
});
```

console.log(obj.sum);

55. **Output:**

## Hard Level

Question:

```
 const obj = { a: 10, b: 20 };

const descriptors = Object.getOwnPropertyDescriptors(obj);

console.log(descriptors.a.enumerable);
```

56. Output:

Question:

```
 const obj = { x: 5, y: 10 };

Object.preventExtensions(obj);

obj.z = 15;

console.log(obj.z);
```

57. Output:

Question:

```
const obj = { a: 1, b: 2 };

Object.defineProperty(obj, "c", { value: 3, enumerable: false });

console.log(Object.keys(obj));
```

58. Output:

Question:

```
 const obj = { x: 1, y: 2 };

const clone = JSON.parse(JSON.stringify(obj));

clone.x = 5;

console.log(obj.x);
```

59. Output:

Question:

```
 const obj = { x: 5, y: { z: 10 } };
const shallowCopy = { ...obj };
shallowCopy.y.z = 20;
console.log(obj.y.z);
```

60. Output:

Question:

```
 const obj = { x: 1 };
Object.defineProperty(obj, "y", { value: 2 });
console.log(obj.y);
```

61. Output:

Question:

```
 const obj = { x: 10 };
Object.defineProperty(obj, "doubleX", {
  get() {
    return this.x * 2;
  },
});
console.log(obj.doubleX);
```

62. Output:

Question:

```
 const obj = { x: 5 };
Object.defineProperty(obj, "y", { value: 10, configurable: false });
```

```
delete obj.y;
console.log(obj.y);
```

63. Output:

Question:

```
 const obj = { x: 5, y: 10 };
const frozen = Object.freeze(obj);
console.log(frozen === obj);
```

64. Output:

Question:

```
 const obj = { x: 1 };
Object.defineProperties(obj, {
  y: { value: 2 },
  z: { value: 3, enumerable: true },
});
console.log(Object.keys(obj));
```

65. Output:

Question:

```
 const obj = { x: 1, y: 2 };
const entries = Object.entries(obj);
entries.push(["z", 3]);
const newObj = Object.fromEntries(entries);
console.log(newObj);
```

66. Output:

Question:

```
 const obj = { a: 1, b: 2 };
Object.defineProperty(obj, "sum", {
  get() {
    return this.a + this.b;
  },
  set(value) {
    this.a = value - this.b;
  },
});
obj.sum = 10;
console.log(obj.a);
```

67. Output:

Question:

```
 const obj = { x: 5 };
Object.defineProperty(obj, "y", { value: 10, writable: false });
obj.y = 20;
console.log(obj.y);
```

68. Output:

Question:

```
 const obj = { a: 10, b: 20 };
const descriptors = Object.getOwnPropertyDescriptors(obj);
console.log(descriptors.a.configurable);
```

69. Output:

Question:

```
 const obj = { x: 1, y: { z: 2 } };
const deepClone = JSON.parse(JSON.stringify(obj));
deepClone.y.z = 10;
console.log(obj.y.z);
```

70. Output:

Question:

```
 const obj = { x: 5, y: 10 };
Object.seal(obj);
obj.z = 15;
console.log(obj.z);
```

71. Output:

Question:

```
 const obj = { a: 1, b: 2, c: 3 };
delete obj.b;
console.log("b" in obj);
```

72. Output:

Question:

```
 const obj = { x: 5 };
Object.defineProperty(obj, "y", { value: 10, configurable: false });
delete obj.y;
console.log("y" in obj);
```

73. Output:

Question:

```
 const obj = { a: 1, b: 2 };
const json = JSON.stringify(obj);
const parsed = JSON.parse(json);
console.log(parsed.b);
```

74. Output:

Question:

```
 const obj = { a: 10, b: 20 };
const descriptors = Object.getOwnPropertyDescriptors(obj);
console.log(descriptors.b.writable);
```

75. Output:

Question:

```
 const obj = { a: 1 };
Object.defineProperty(obj, "b", {
  get() {
    return this.a * 2;
  },
  set(value) {
    this.a = value / 2;
  },
});
obj.b = 8;
console.log(obj.a);
```

76. Output:

Question:

```
 const obj = { a: 1 };
const freeze = Object.freeze(obj);
console.log(Object.isFrozen(freeze));
```

77. Output:

Question:

```
 const obj = { x: 10, y: 20 };
const sum = Object.values(obj).reduce((acc, val) => acc + val, 0);
console.log(sum);
```

78. Output:

Question:

```
 const obj = { x: 1 };
Object.defineProperty(obj, "y", { value: 2, enumerable: true });
console.log(Object.keys(obj));
```

79. Output:

Question:

```
 const obj = { x: 5, y: 10 };
Object.seal(obj);
obj.x = 15;
console.log(obj.x);
```

80. Output: