```
console.log(10+10);
```
output: 20

```
console.log(10+"10");
```
output: 1010

```
console.log(10 + + "10");
```
output: 20

+"10": The + operator before the string "10" is a unary plus operator. When applied to a string, it tries to convert the string to a number. In this case, the string "10" is converted to the number 10.

```
console.log(10 + "10" + 10);
```
output: 101010

10 + "10": The number 10 is implicitly converted to a string, and then concatenated with the string "10", resulting in "1010".
"1010" + 10: The string "1010" is concatenated with the number 10, resulting in the string "101010".

```
console.log(10+ +"10" + 10);
```
output: 30

```
console.log(10 - "2");
```
output: 8

```
console.log(10 - "2" - "8");
```
output: 0

When JavaScript encounters the- operator, it tries to convert the operands to numbers. So, "2" and "8" are converted to the numbers 2 and 8 respectively.

```
console.log(10+"2" - "2");
```
output: 100

```
console.log(10>9>8);
```
output: False

10 > 9: This comparison evaluates to true because 10 is indeed greater than 9.
true > 8: Now, JavaScript tries to compare true with 8. In JavaScript, true is implicitly converted to the number 1, and false to 0 when involved in numerical operations. So effectively, this becomes 1 > 8.

```
console.log(10 * "10");
```
output: 100

```
console.log(100 / "100");
```
output: 1

```
console.log(100/"0");
```
output: Infinity

```
console.log(100 + +"100" - "100" * "100");
```
output: -9800

```
console.log(1 == "1");
```
output: True

```
console.log(1 === "1");
```
output: False

```
console.log(1 == "one");
```
output: False

```
console.log(1 === "one");
```
output: False

```
console.log(1+true);
```
output: 2

```
console.log(1 - true);
```
output: 0

```
console.log(1 + true - false);
```
output: 2

```
console.log("1" + true);
```
output: 1true

```
console.log(+"1" + true);
```
output: 2

```
console.log(undefined == undefined);
```
output: True

```
console.log(undefined === undefined);
```
output: True

```
console.log(null == null);
```
output: True

```
console.log(null === null);
```
output: True

```
console.log(undefined == null);
```
output: True

```
console.log(undefined === null);
```
output: False

```
console.log(2+NaN);
```
output: NaN

```
console.log("2"+NaN);
```
output: 2NaN

```
console.log("2"+undefined);
```
output: 2undefined

```
console.log(2+undefined);
```
output: NaN

```
console.log(typeof "123");
```
output: String

```
console.log(typeof 2);
```
output: number

```
console.log(typeof true);
```
output: Boolean

```
console.log(typeof undefined);
```
output: undefined

```
console.log(typeof null);
```
output: object

```
console.log(typeof []);
```
output: object

```
console.log(typeof 1n);
```
output: bigint

```
console.log(typeof 1n+2n);
```
output: bigint2

```
console.log(typeof 1+2n);
```
output: number2

```
console.log(typeof 1/1n);
```
output: error: cannot mix bigint and other types use explicit conversion.

---

1. What is the value of x after the operation: x = 5 + 3 * 2;?

```
let x;
 x = 5 + 3 * 2;
console.log(x);
```
output: 11

2. What is the value of y after the operation: y = 12 - 4 / 2;?

```
let y;
y = 12 - 4 / 2;
console.log(y);
```
output: 10

3. What is the value of z after the operation: z = 7 + 2 * 3 - 1;?

```
let z;
z = 7 + 2 * 3 - 1;
console.log(z);
```
output: 12

4. What is the value of a after the operation: a = 9 % 3 + 2;?

```
let a;
a = 9 % 3 + 2;
console.log(a);
```
output: 2

5. What is the value of b after the operation: b = 15 / 3 * 2;?

```
let b;
b = 15 / 3 * 2;
console.log(b);
```
output: 10

6. What is the value of c after the operation: c = 24 >> 2;?

```
let c;
c = 24 >> 2;
console.log(c);
```

output: 6

c = 24 >> 2;: Here, 24 is represented in binary as 11000. When you shift the bits of 11000 two positions to the right, you get 110, which represents the decimal number 6. So, c is assigned the value 6.

7. What is the value of d after the operation: d = 17 & 3;?

```
let d;
d = 17 & 3;
console.log(d);
```

output: 1

8. What is the value of e after the operation: e = 28 ^ 2;?

```
let e;
e = 28 ^ 2;
console.log(e);
```

output: 30

11100 XOR 00010 gives 11110 (i.e. 30)

9. What is the value of f after the operation: f = 11 + 3 << 2;?

```
let f;
f = 11 + 3 << 2;
console.log(f);
```

output: 56

10. What is the value of g after the operation: g = 25 - 5 | 3;?

```
let g;
g=25-5|3;
console.log(g);
```

output: 23

10100 | 00011 = 10111 = 23

---

1. What is the value of granted after the operation:

let username = "admin";

let password = "password";

let granted = (username === "admin" && password === "password") ? true : false;

output: True

2. What is the value of message after the operation:

let username = "user";

let password = "wrongpassword";

let message = (username === "admin" && password === "password") ? "Login successful!" : "Invalid credentials.";

output: Invalid credentials

3. What is the value of access after the operation:

let username = "admin";

let password = "password";

let access = (username === "admin" || password === "password") ? "Granted" : "Denied";

output: Granted

4.What is the value of status after the operation:

let username = "";

let password = "password";

let status = (username !== "" && password === "password") ? "Logged in" : "Please enter username and password";

output: Please enter username and password

5. What is the value of authenticated after the operation:

```
let username = "admin";
let password = "wrongpassword";
let authenticated = (username === "admin" && password === "password") ? true : false;
```

output: false

---

1. What is the value of name after the operation:

```
let user = { name: "John" };
let name = user?.name ?? "Unknown";
console.log(name);
```

output: John

2. What is the value of price after the operation:

```
let product = { price: null };
let price = product?.price ?? "N/A";
console.log(price);
```

output: N/A

(Nullish coaleshing operator(??) returns the second operand if the value is null or undefined)

3. What is the value of address after the operation:

```
let customer = { address: { street: "123 Main St" } };
let address = customer?.address?.street ?? "Not available";
console.log(address);
```

output: 123 Main St

4. What is the value of phone after the operation:

```
let contact = { phone: null };
let phone = contact?.phone ?? "Not provided";
console.log(phone);
```

output: Not provided

(Nullish coaleshing operator(??) returns the second operand if the value is null or undefined)

5. What is the value of description after the operation:

```
let item = { description: "" };
let description = item?.description ?? "No description available";
console.log(description);
```

output: empty string

(Nullish coaleshing operator(??) returns the first operand if the value is not null or undefined)