2. Understanding Conditionals

Q) Valid Voter

```
let age = parseInt(prompt("What's your age?"));

if (age >= 18) {
    console.log("You can vote");
} else {
    console.log("You can't vote");
}
```

ParseInt → 3.14 → 3 } Too age use
Number → 3.14 → 3.14 } Number

→ here if user gives invalid input like 'xyz', it will say "you can't vote", so to fix it we can run another condition.

* In JS,
Nan === Nan → false
to check Nan,
isNan(Number) → true
↳ convertable num

isNan("haha")
↓
true

```
if (isNan(age)) {
    console.log("Woong Input");
}
else if (age > 18) {
    console.log("You can vote");
}
else {
    console.log("can't vote");
}
```

Q) Shop Discount

Amount	Discount
0-5000	0%
5001-7000	5%
7001-9000	10%
more than 9000	20%

→ Find pay after discount

```
let num = Number(prompt("What's the given amount?:"));

function showAmtAfterDis(disPer, totalAmt){
    console.log("Amount After discount is",
    Math.round(totalAmt - ((disPer * totalAmt)/100)));
}
if(isNan(num)){
    console.log("Provide number please!");
}
else if(num>0 && num<=5000){
    showAmtAfterDis(0,num);
}
else if(num>5000 && num<=7000){
    showAmtAfterDis(5,num);
}
else if(num>7000 && num<=9000){
    showAmtAfterDis(10,num);
}
else{
    showAmtAfterDis(20,num);
}
```

Q) Electricity Bill

Unit	Price
upto 100	Rs 4.2/unit
101-700	Rs 6/unit
701-400	Rs 8/unit
more than 400	Rs 13/unit

→ lets say Unit is 250
upto 100 → 4.2 x 100 is applied
(remains → 250-100) = 150
next 100 → 6 x 100 → Rs 600
Remaining, 50 unit
50 x 8 = Rs 400
Total payable → 420 + 600 + 400
= Rs 1420

Approach → write code from bottom to top instead of top to bottom.
We will apply code with waterfall like structure

code:

```
let unit = Number(prompt("Enter the unit:"));   // lets say 700
let billAmount = 0;   → initially 0
if (unit > 400) {
    let applicableAmt = unit - 400;   // 700-400 → 300
    billAmount += applicableAmt * 13;   // 3900
    unit = 400;   // units above 400 is done, lets reduce the value of unit for another step
}
```

this is same as `billAmount = billAmount + applicableAmt * 13;`

```
if (unit <= 400 && unit > 200) {   // unit → 400
    let applicableAmt = unit - 200;   // 200
    billAmount += applicableAmt * 8;  // 3900 + 1600
    unit = 200;  → again changing unit      = 5500
}

if (unit <= 200 && unit > 100) {
    let applicableAmt = unit - 100;
    billAmount += applicableAmt * 6;  // 5500 + 600
    unit = 100;                       6100
}

billAmount += unit * 4.7;   // 6100 + 470
// only 1 condition is remaining, no need of if-statement

console.log("The bill is: ", billAmount);
                              → 6570
```

Q) INR Denomination

User will give a ~~no~~ number, your code have to calculate how many currency Amount it can create from biggest to smallest.

Currencies:

Rs 500 Rs 200 Rs 100 Rs 50 Rs 20
Rs 10 Rs 5 Rs 2 Rs 1

→ If user gives, Rs 770

He can have → 1 note of Rs 500
→ 1 note of Rs 200
→ 1 note of Rs 70

No. of Rs 500 notes in 720,
Math.trunc(500 720/500)) → 1
Remaining rupees → 720 % 500 → 200

⇒
let rupees = parseInt(prompt("Enter the rupees"));

//checking Rs 500 notes
if(rupees >= 500) {
console.log("Rs 500 notes:", Math.trunc(rupees/500)); // count Rs 500 notes
rupees = rupees % 500; //updating remaining Amou
}

//checking Rs 200 notes
if(rupees >= 200 && rupees < 500) {
console.log("Rs 200 notes:", Math.trunc(rupees/200));
rupees = rupees % 200;
}

... same logic for Rs 100, 50, 20, 10, 5, 2

if(rupees === 1) {
console.log("Re 1 coins:", rupees);
}

⇒ output for 555
Rs 500 notes: 1
Rs 50 notes: 1
Rs 5 notes: 1

Ternary operator

condition ? "if-true" : "else";

Eg: Find the number is positive, negative or zero

let num = 0; → user will give
console.log(num > 0 ? "Positive" : num < 0 ? "Negative" : "zero");

↳ nested ternary operator

Switch statement

↳ To stop using long if-else chain, we can use switch statement.
↳ It helps to perform multiple check condition check operation.

Eg: user = 1 (0 → monday, 1 → Tuesday etc)
↑ will match with value of user

```
switch(user) {
    case 0 : console.log("Monday");
    break;

    case 1 : console.log("Tuesday");
    break;
    ...
    default : console.log("Invalid week day");
}
```

* fallthrough in switch

If switch conditions does not have break, code will execute to next condition, even if it does not match expression

```
let day = 1;
switch (day) {
    case 0: console.log("Monday");
    break;
    case 1: console.log("Tuesday");
    break          → break is not used
    case 2: console.log("Wednesday");
    break
}
```

→ output:

Tuesday
Wednesday

* When can we use switch

↳ When we have constant value, and we have to match with many conditions.

Eg if we have classroom of many students. To find user, we can use switch condition with roll no to find user info

```
switch (rollNo) {
    switch 1:
        console.log("Ash");
        break
    ....
}
```

* You will roll a dice, make an effcient switch statement to tell the di number coming in dice is odd or even

```
let dice = _ → [1, 2, 3, 4, 5, 6]

switch (dice) {
    switch 1:
    switch 3:
    switch 5:
        console.log("odd");
        break

    switch 2:
    switch 4:
    switch 6:
        console.log("even");
        break
}
```

Truthy switch pattern

```
let score = 65;

switch (true) {
    case score > 50:
        console.log("Pass");
        break;
    case score <= 50:
        Console.log("Failed");
        break
}
```

→ make us eligible to run if-else like conditions inside switch

Precision

console.log(0.1+0.2) → 0.3000.....4

To get 0.3

console.log((0.1+0.2).toFixed(1)) → 0.3

0.78.toFixed(1) → 0.8

0.78.toFixed(3) → 0.780