Day 3.1

JAVASCRIPT

- Java is a strongly typed(datatype) language.
- Javascript is a weakly typed(datatype) language.
- For example:
    - In java you define int a= 2; or boolean x=true; or double x=12.55532;.
    - In Javascript no need to define all different kinds of datatype, only the following keywords are used to define a variable
        - var
        - const
        - let
    - var a=23; for this, javascript will automatically know that ‘a’ is a number
    - var a=”newton”; for this, javascript will automatically know that ‘a’ is a string.

var keyword

- var keyboard is used to define variable in javascript. This keyword tells that the variable has a global scope.
- var a=23;, 1000lines later now you say a=”newton”. This is acceptable is javascript. But first the datatype was a number after thousand lines the datatype become string.
- If other person changes the code then they won’t understand what is present in ‘a’. Is it a string or int or boolean. They might get confused.
- Javascript can tell which type of variable it is using typeof operator

typeof operator

```
if( typeof a == "string" )
```

- type of tells you what is the datatype of the variable. The above statement will become true if variable ‘a’ is of string datatype. That means ‘a’ contains a string value then if condition is satisfied.
- Var causes so many code maintenance issue

const and var keywords

- const a=23; This statement will store value 23 in variable ‘a’.
- We cannot reassign a. Once you assign value you can’t assign anything else to variable ‘a’.
- const is like a final keyword in java.
- Difference between const and var is that, you cannot reassign value to a variable using const and you can reassign value to a variable using var.

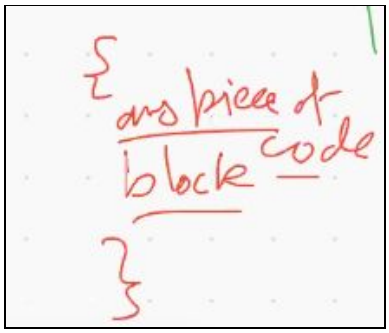
let keyword

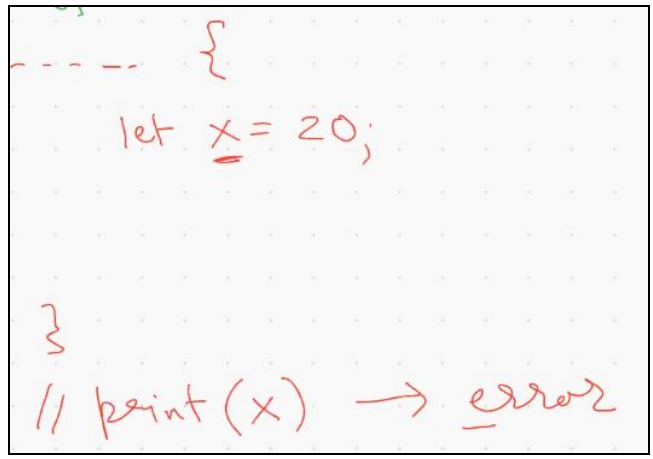
- let keyword is similar to var keyword, let a=24; the only difference between ‘let’ keyword and ‘var’ keyword is that ‘let’ keyword has a block scope while ‘var’ keyword has a global scope.

- ‘let’ is preferable to use than var.
- let is very useful when you use loop.
- Always use ‘const’, if you have to change the value of variable in future only than use ‘let’.

Diff between let and var explained in detail.

- Scoping: var has a global scope and let has block scope.
- A block means any piece of code between two curly braces.

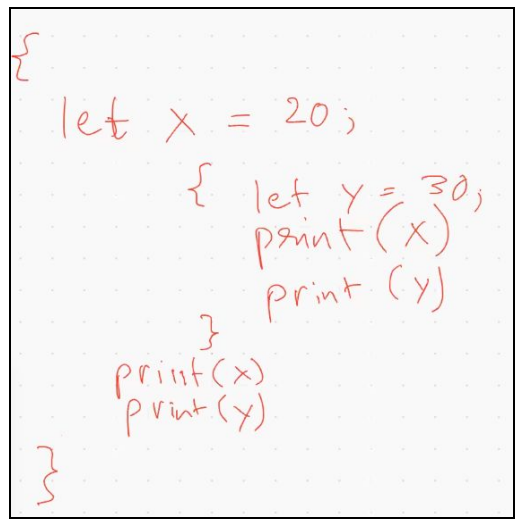


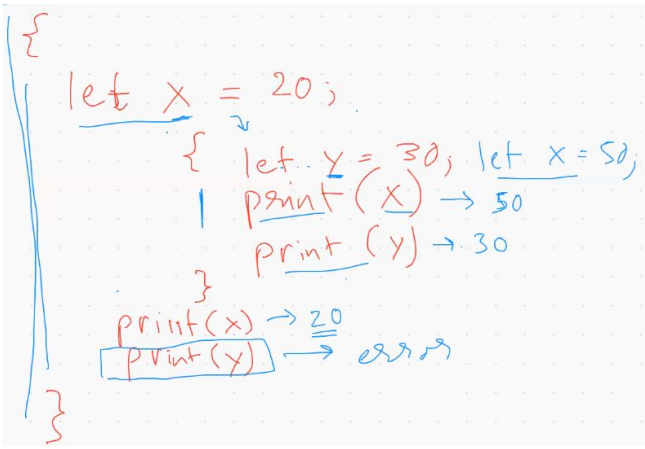


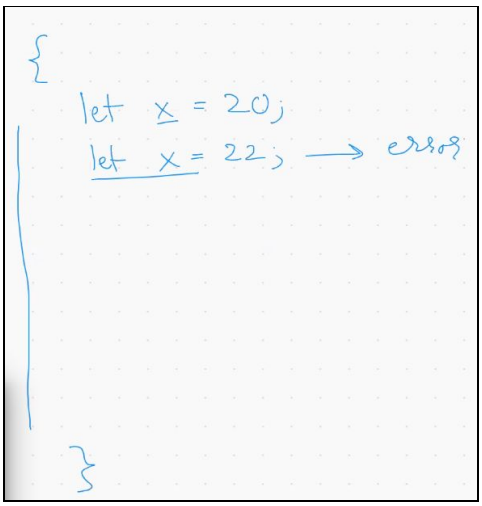
- Right image shows that printing x out of the curly braces will cause error.
- Warning: Their is nothing like print(“something”) in javascript. This is just an example for printing

Nested scope example

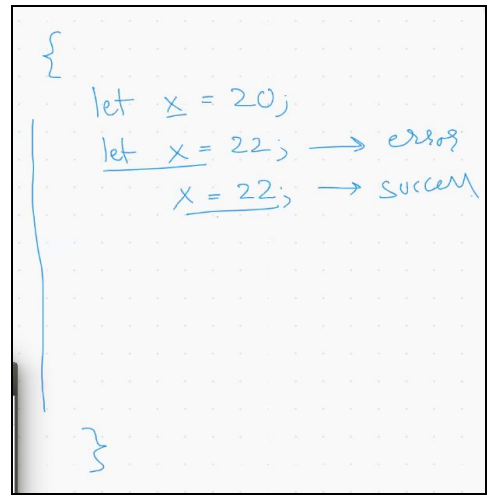
- From the image in the right which lines will cause an error. Total no. of lines are 10.
- Line number 9 (second last line will cause error)
- Reason: Because variable y is declared inside the nested scope. And we are trying to print it outside its scope.
- Below example explains where you will cause error. The variable ‘z’ has the entire scope. ‘let x=50;’ <- This whole this is a Declaration of variable x using keyword let and assigning the value ‘50’ to x.



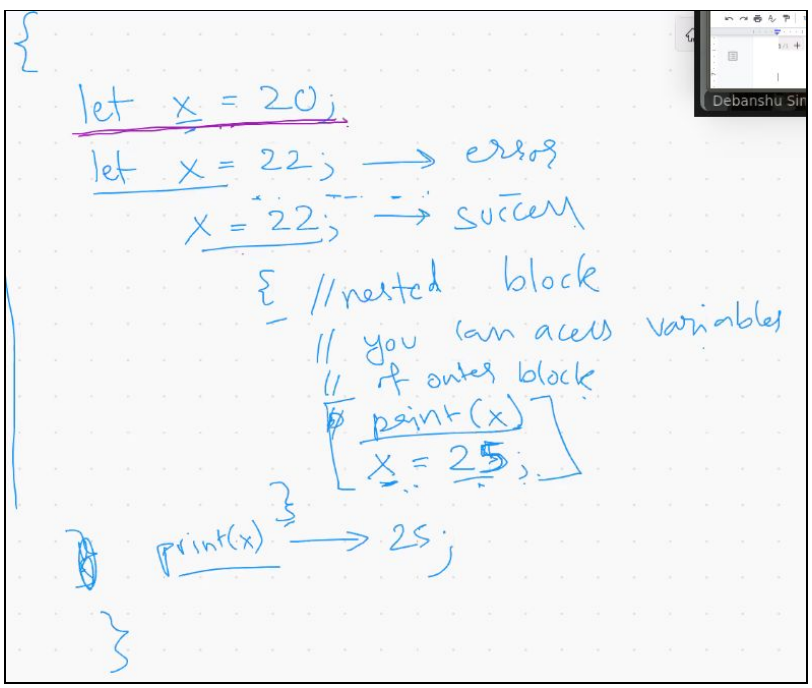




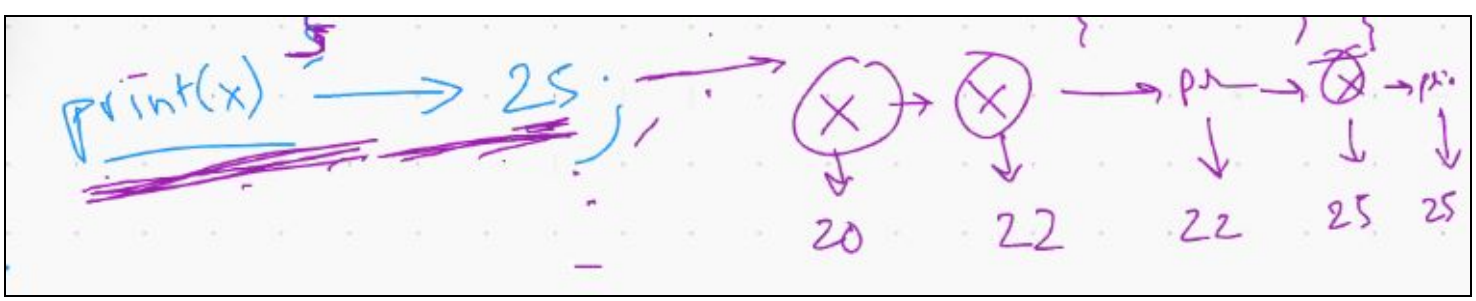
- The image in the right explains that, you cannot declare the variable with same name two times within one single scope. But it is possible to declare a variable with same name in different scope. It is possible to reassign the value to the variable within a scope as many times as you want. The blue line below opening curly brace and above closing curly brace shows the scope of that block.

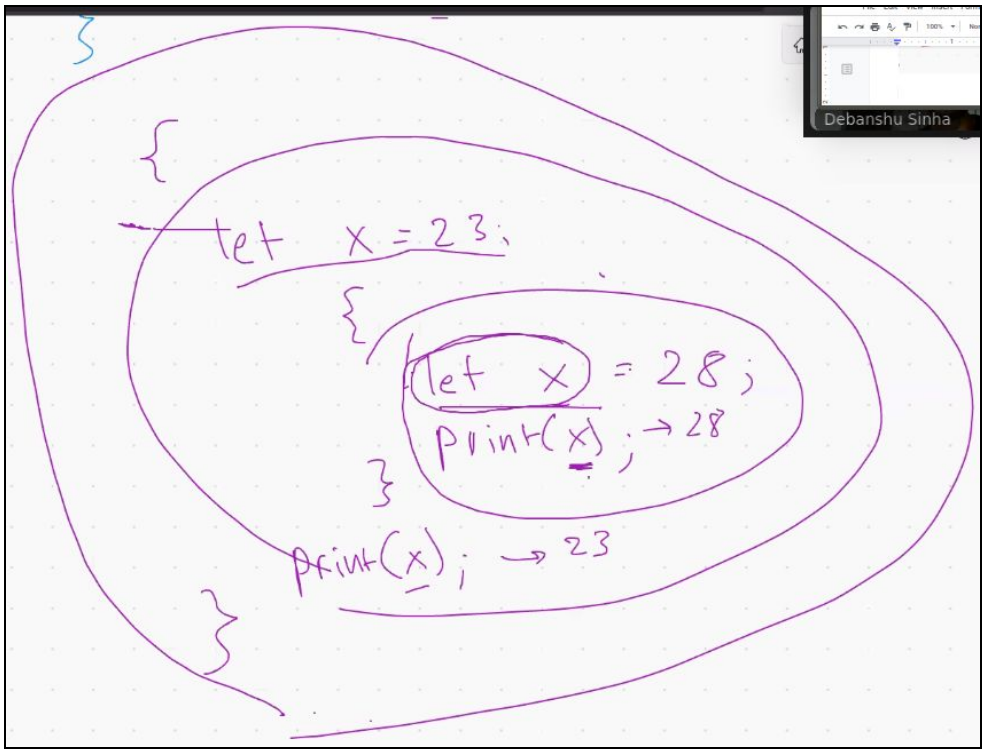


- Reassigning of value to variable x is shown in the right image.
- You can access the variables of outer block in the nested block. But you cannot access the variables of nested block in the outer block. This is shown in the program below.



- Next is the nested block, inside the nested block print(x) will print 22. x=25 is resetting the value of ‘x’ to 25. Next line print(x) will print 25. It will not print 22.





- The above image explains what is called as variable hiding. You are hiding x=23 of outer block. That means let x=23; of outer block has no effect on nested block because you redeclared it inside nested block.

- This is not a recommended practice we should not declare the same variable inside a nested block because it will cause confusion. Take another name of the variable instead of x.

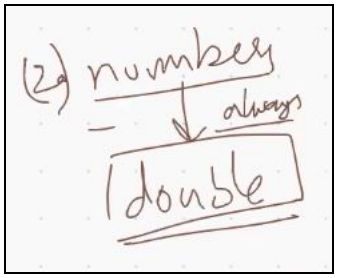
var keyword

- var a=100;
- var has the global scope. Global scope basically means once you declare a variable using var keyword, you can use it anywhere in the code.
- For Internet explorer 5, if you are writing code then you can't use 'const' and 'let' keyword.

Printing In javascript

```
console.log("Hello World!")
```

- console.log will always work link println. It always prints in next line. Their is no such code like print and println in javascript.



Important points:

- Semicolon is optional in javascript: Its on you that you wanna use semicolon at the end of the code or not.
- In javascript their is only one datatype present for numbers and that is 'double'.

- n/10 gives answer in decimal points to perform integer division use Math.floor(n/10) or parseInt(n/10) as shown in below image.

```
n1 = Math.floor(n1/10);
n2 = parseInt(n2/10);
```

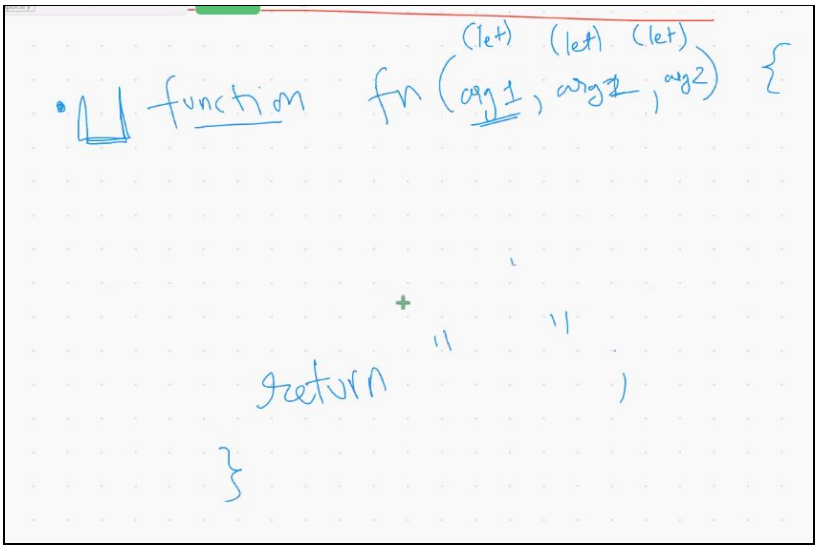
- If you don't know any of these function then

```
// n1 = Math.floor(n1/10);
// n2 = parseInt(n2/10);
n1 = (n1 - r1)/10;
n2 = (n2 - r2)/10;
```

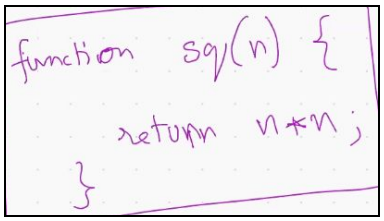
Where r1 and r2 are remainders.

- Use of const will cause less error in production time.

Declaring a function using keyword 'function'



- Image above is the syntax of a function.
- Functions in js are first-class members. It means function can act like a variable having a datatype named 'function'.
- Function can be used as a value in javascript. This will be clarified in the later lectures
- Write a function to that gives square of a number.



Write the function which will take a number and gives me the cube

```
function cube(n) {
    return n*n*n;
}
```

```
function nextLargerEven(n) {
    return n%2==0 ? n+2 : n+1;
}
```

Write a function which will take number and gives next larger even number

Function is like a data type

```
function tellType(a) {
    console.log(typeof a);
}

tellType(42);
```

```
STDOUT •  STDE
1  number
2
```

- number is a datatype of 42.

- 'typeof' keyword returns the string representation of the datatype.

```
function tellType(a) {
    const type = typeof a;
    console.log(typeof type);
    console.log(type);
}

tellType(42);
```

```
STDOUT •  STDI
1  string
2  number
3
```

- Instead of 42 if you use "42", then Output will be

String String

- Some Code with output.

```
function tellType(a) {
    const type = typeof a;
    console.log(typeof type);
    console.log(type);
}

function sq(n) {
    return n *n;
}

function cube(n) {
    return n*n*n;
}

function largerEven(n) {
    return n%2 ==0? n+2: n+1;
}

tellType(largerEven);
```

```
STDOUT •  STDER
1   string
2   function
3
```

- Function is act like a input variable in java that have a datatype named ‘function’.
- This is the meaning of function as 1st class member

What will be the output of following code?

```
function sq(n) {
    return n *n;
}

function cube(n) {
    return n*n*n;
}

function largerEven(n) {
    return n%2 ==0? n+2: n+1;
}

// tellType(cube(20));
function applyFn(n, fn) {
    return fn(n);
}
console.log(applyFn(3, sq));
```

Answer: 9