exclude to 20.

Javascript -6

Functions →

Syntax→

```
function    functionname ( ) {
            // do something
}
```

Calling →

```
functionname ( ) ;
```

Functions with arguements →

```
function    functionname (arg1, arg2 ----) {
            // do something
}
```

[args1, args2 are variables for function input]

classmate

PAGE

return keyword →

- return keyword is return some value
- and stop the next code to execute
- It Can return single value or variable
  that variable can int/string/array/object.etc

Scope →

A variable, object and function 's accessibility

Types → Function
Block
Lexical

- function scope → variables that defined inside function are not accessable outside. of function.
  [function scope > Global scope]
-
- Block scope → variables declared inside {} block cannot be accessed from outside the block
- Lexical scope → Variable defined outside a function can be accessible inside another function defined after the variable declaration. (nested function)

The opposite is not true.

hoisting →

function Expressions → a second way to write a function

Syntax →
```
let variable = function (a,b) {
                return a+b;
    }
```

calling →
```
variable ( a,b) ;
```

High order Functions →

A function that does one or both

→: takes one or multiple functions as arguments
→: returns a function

methods → functions that define in object

syntax → (a)
```
Const object = {
        key : function,
    }
```

[function = function () { do }]

syntax → (b)
```
const object = {
    variable (a,b) {
            return a+b ;
    }
}
```

[variable = key name/ function name]