**0) Function under normal blocks are not considered/checked if hoistable**

**function foo(){**

console.log("foo");

}

**function bar(){**

console.log("bar");

}

for (var i = 0; i < 10; i++){

if (i%2 === 0){

**foo();**

} else {

**bar();**

}

}

console.log("done");

**Results for example.js:**

Function ---> foo is the outer most function

Function ---> bar is the outer most function

**1) Nested function declaration and call , checked if hoistable**

**function foo(){**

var a = 1;

if(a === 1){

var b = a + 2;

}

var c = function **goo(){**

return 5;

};

var d = b + c() ;

console.log(d);

}

**foo();**

**Results for example1.js:**

Function ---> foo is the outer most function

Function ---> goo under goo can be hoisted

**2) Nested functions, inner functions declared outside are not checked if hoistable**

var x = 23;

**function g(step)** {

return x + step;

}

**function f(a)** {

**g(a);**

}

**f(1);**

**Results for example2.js:**

Function ---> g is the outer most function

Function ---> f is the outer most function

3) **Nested functions, inner functions declared inside are checked if hoistable**

var x = 23;

**function f(a)** {

**function g(step)** {

return x + step;

}

**g(a);**

}

**f(1);**

**Results for example3.js:**

Function ---> f is the outer most function

Function ---> g under f can be hoisted

**4)More that one nested function declaration in same level and calls are also same level**

var x = 23;

function f(a) {

function g(step) {

return x + step;

}

function h(step) {

return x + step;

}

g(a);

h(a);

}

f(1);

**Results for example4.js:**

Function ---> f is the outer most function

Function ---> g under f can be hoisted

Function ---> h under f can be hoisted

**5)more that one nested function decleration in same level and calls are different level**

var x = 23;

function f(a) {

function g(step) {

h(step);

return x + step;

}

function h(step) {

return x + step;

}

g(a);

}

f(1);

**Results for example5.js:**

Function ---> f is the outer most function

Function ---> g under f can be hoisted

Function ---> h under f can be hoisted

**6)More that one nested function decleration in different level and calls in corresponds with the level**

var x = 23;

function f(a) {

var y = 25;

var b = 1;

z = 27; // z is a global variable as var is not used , but not able to intercept in analysis

function g(step) {

var c = 12;

function h(step) {

return y + z + step;

}

h(step);

return

x + step;

}

g(a);

}

f(1);

**Results for example6.js:**

Function ---> f is the outer most function

Function ---> g under f can not be hoisted because one of its child is non hoistable

Function ---> h under g can not be hoisted Due to the variable y decleared under ansestor f

7)

function add(a, b) {

function addB(x) {

return x + b;

}

function add5(x) {

return x + 5;

}

if (b === 5) {

return add5(a);

} else {

return addB(a);

}

}

add(3,5);

**Results for example7.js:**

Function ---> add is the outer most function

Function ---> addB under add can not decide if hoistable

Function ---> add5 under add can be hoisted

8)

/\*scenarion 1\*/

function add1(a, b) {

function add1B(x) {

return x + 5;

}

function add15(x) {

return x + b;

}

return add15(a) + add1B(a);

}

add1(3,4);

/\*scenarion 2\*/

function add2(a, b) {

function add2B(x) {

return x + b;

}

function add25(x) {

return x + 5;

}

return add2B(a) + add25(a);

}

add2(3,4);

**Results for example8.js:**

Function ---> add1 is the outer most function

Function ---> add2 is the outer most function

Function ---> add1B under add1 can be hoisted

Function ---> add15 under add1 can not be hoisted Due to the variable b decleared under ansestor add1

Function ---> add2B under add2 can not be hoisted Due to the variable b decleared under ansestor add2

Function ---> add25 under add2 can be hoisted

9)

var x = 23;

function f1(a) {

var y= 24;

function g(step) {

return x+ y + step;

}

g(a);

function h1(step) {

return x + step;

}

h1(a);

}

function f2(a) {

var y= 25;

function g(step) {

return x+ y + step;

}

function h(step) {

var y = 1;

return x+ y + step;

}

g(a);

h(a);

}

f1(1);

f2(1);

**Results for example9.js:**

Function ---> f1 is the outer most function

Function ---> f2 is the outer most function

Function ---> g under f1 can not be hoisted Due to the variable y decleared under ansestor f1

Function ---> h1 under f1 can be hoisted

Function ---> g under f2 can not be hoisted Due to the variable y decleared under ansestor f2

Function ---> h under f2 can be hoisted

10)

var x = 23;

function fLevel1(a) {

var y= 24;

function fLevel2(step2) {

var z=25;

function fLevel3(step3) {

var zz=26;

function fLevel4(step4) {

return x +z+ step4;

}

fLevel4(zz);

return x + step3;

}

fLevel3(z);

return x+ step2;

}

fLevel2(a);

}

fLevel1(1);

**Results for example10.js:**

Function ---> fLevel1 is the outer most function

Function ---> fLevel2 under fLevel1 can be hoisted

Function ---> fLevel3 under fLevel2 can not be hoisted because one of its child is non hoistable

Function ---> fLevel4 under fLevel3 can not be hoisted Due to the variable z decleared under ansestor fLevel2