ASSIGNMENT 1

1.What will be the output of this code?

```
console.log(x);
var x=5; //undefined
```

A) Undefined

Exp: When console.log(x); runs the variable x has been declared but not yet assigned a value, so its value is undefined. The value 5 is assigned to x only after the console.log statement. So, it's undefined.

2.What will be the output of this code?

```
console.log(a);
Var a;
```

A)Undefined

Exp: Here, variable declaration using var are hoisted to top of their scope. Since no value is assigned to "a", its value is undefined. When console.log(a) runs, it prints undefined because a exists but it has no value. So, it's undefined.

3.What will be the output of this code?

```
console.log(b);

b=10;

var b;
```

A)Undefined Error

Exp: Hoisting moves declaration var b; to the top, but not the assignment b = 10. When console.log(b) runs, b is declared but not assigned a value, so it prints undefined. After the console.log, b is assigned the value 10, but it does not affect the output.

4.What will happen here.

```
console.log(c);
```

A) Reference Error

Exp: Here c is not declared or defined, it throws reference error.

5.What will be the output of this code?

```
console.log(e);

var e=10;

console.log(e);

e=20;

console.log(e);
```

A)Undefined

10

20

Exp: Here, the first console.log(e) prints the output undefined, because e is hoisted and it’s not assigned any value.

Next, logs print outputs 10 and 20, because the values are assigned to e.

6.What will be the output of this code?

```
console.log(f);
var f=100;
var f;
console.log(f);
```

A)Undefined

100

Exp: Here, the first console.log(f) prints undefined, the next console.log(f) prints 100.

7.What will be the output of this code?

```
console.log(g)

var g=g+1;

console.log(g);
```

A)Undefined

NaN

Exp: Here, in first console we get output as “undefined” because g is hoisted but not initialized when the log runs.

In the second console, we are trying to add two different data types which results the output NaN.

8.What will be the output of this code?

```
var h;
```

```
console.log(h);

h=50;

console.log(h);
```

A)undefined

50

Exp: Here, in the first console var h is just declared but not initialized. So the output will be undefined. At this point, it exists in memory but has no assigned value yet.

In the second console, the var h is initialized, so the output will be 50.

9.What will be the output of this code?

```
console.log(i);

i=10;

var i=5;

console.log(i);
```

A)undefined

5

Exp: Here, in the first console I is initialized, but any value is not assigned.

Next, `i` is assigned the value `10`, but it doesn't affect the hoisted declaration.

Next,`var i = 5; assigns `5` to `i`, but it's hoisted to the top of the function or global scope.

In the second `console; we get outputs 5, because it’s the most recent value assigned to i.