PROMISES ASSIGNMENT

1. What will be output of following code?

```
function job() {
return new Promise(function(resolve, reject) {
reject();
});
}
let promise = job();
promise
.then(function() {
console.log('Success 1');
})
.then(function() {
console.log('Success 2');
})
.then(function() {
console.log('Success 3');
})
.catch(function() {
console.log('Error 1');
})
.then(function() {
console.log('Success 4');
});
```

Output: -

```
Error 1
Success 4
```

2. Write a sleep function using a promise in javascript?

```
function sleep(millisecs){
    let promise = new Promise(function(resolve, reject){
        console.log(`sleeping for ${millisecs} milliseconds`);
        setTimeout(() => {
            console.log("finished sleep");
            resolve();
```

```
                }, millisecs);
            });

        }
```

Same code is also present in “Task_2.js” in the package.

3. What is the output of the following code?

```
const promise = new Promise(res => res(2));
promise.then(v => {
                console.log(v);
                return v * 2;
        })
        .then(v => {
                console.log(v);
                return v * 2;
        })
        .finally(v => {
                console.log(v);
                return v * 2;
        })
        .then(v => {
                console.log(v);
        });
```

Output: -

```
2
4
undefined
8
```

4. What is the output?

```
console.log('start')
const fn = () => (new Promise((resolve, reject) => {
        console.log(1);
        resolve('success')
}))
console.log('middle')
fn().then(res => {
        console.log(res)
```

})

console.log('end')

Output:

```
start
middle
1
end
success
```

5. Write a function delay that returns a promise. And that promise should return a setTimeout that calls resolve after 1000ms.

```
function delay(){
    let promise = new Promise((resolve, reject) => {
        let timeOut = setTimeout(()=>{
            resolve();
        },1000);
        return timeOut;
    });

    return promise;
}
```

Same code is also present in “Task_5.js” in the package.

6. What will be the output?

```
Promise.resolve('Success!')
.then(data => {
        return data.toUpperCase()
})
.then(data => {
        console.log(data)
})
```

Output

```
SUCCESS!
```

7. What will be the output?

```
var p = new Promise((resolve, reject) => {
        reject(Error('The Fails!'))
})
.catch(error => console.log(error))
.then(error => console.log(error))
```

Output

```
Error: The Fails!
 …Stacktrace of the error
…
…..
Undefined
```

Stacktrace will be printed because we are printing the error object in ‘catch’. If we print ‘error.message’ then only “The Fails!’ will be printed.

“Undefined” is printed as the catch block didnt return anything.

8. What will be the output?

```
console.log('start')
setTimeout(() => {
        console.log('setTimeout')
})
Promise.resolve().then(() => {
        console.log('resolve')
})
console.log('end')
```

Output

```
start
end
resolve
setTimeout
```

9. What will be the output

```
console.log('start')
Promise.resolve(1).then((res) => {
        console.log(res)
})
Promise.resolve(2).then((res) => {
        console.log(res)
})
console.log('end')
```

Output:

```
start
end
1
2
```