

```
const fn = (isBad) => {
```

```
  return (n) => {
```

```
    let left = 1
```

```
    let right = n;
```

```
    while(left < right) {
```

```
      const mid = Math.floor((left + right) / 2);
```

```
      if (isBad(mid)) {
```

```
        right = mid;
```

```
      } else {
```

```
        left = mid + 1;
```

```
      }
```

```
    }
```

```
    return left;
```

```
  }
```

Diagram illustrating the binary search process for finding the minimum number of bits (b) required to represent a number (n) using the isBad function.

Initial state: $i = 1, j = 5, b = 3$

Step 1: $i = 1, j = 5, b = 3$

Step 2: $i = 1, j = 5, b = 4$

Step 3: $i = 1, j = 5, b = 4$

Step 4: $i = 1, j = 5, b = 4$