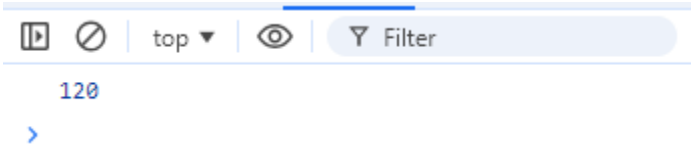
NAME: SANTHOSH AS
ROLL NO: 717823F148

TASK 1

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>TASK</title>
</head>
<body>
   <script>
      function fact(n){
         if(n==0)
         return 1
      else{
         return n*fact(n-1);
      }
      }
     console.log(fact(5))
   </script>
</body>
</html>
```

OUTPUT



TASK 2

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>TASK</title>
</head>
<body>
   <script>
      function fibo(n){
         if(n<=1)
         return 0
         if(n==2)
         return 1
         return fibo(n-1)+fibo(n-2)
      }
     console.log(fibo(6))
   </script>
</body>
</html>
```

OUTPUT

```
8
```

TASK 3

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>TASK</title>
</head>
<body>
    <script>
        function countWays(n){
            if(n<0)
            return 0
            if(n==0)
            return 1
            return countWays(n - 1) + countWays(n - 2) + countWays(n - 3);
        }
       console.log(countWays(4))
    </script>
</body>
</html>
```

OUTPUT

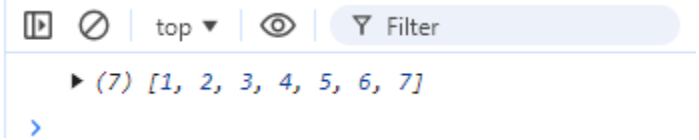
```
7
>
```

TASK 4

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>TASK</title>
</head>
<body>
    <script>
       function flattenArray(arr) {
        let result = [];
        for (let i = 0; i < arr.length; i++) {
            if (Array.isArray(arr[i])) {
                result = result.concat(flattenArray(arr[i]));
            } else {
                result.push(arr[i]);
            }
        }
        return result;
    }
    console.log(flattenArray([1, [2, 3], [4, [5, 6]], 7]));
    </script>
```

```
</body>
</html>
```

OUTPUT



TASK 5

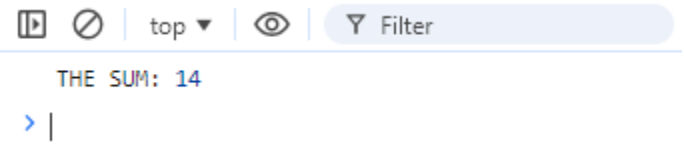
OUTPUT

TASK 6

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>TASK</title>
</head>
<body>
   <script>
      function sum(...a){
       return a.reduce((s,e)=>s+e,0)
      }

   console.log("THE SUM:",sum(2,3,9));
   </script>
</body>
</html>
```

OUTPUT



TASK 7

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>TASK</title>
</head>
<body>
   <script>
      function sum(...a){
       return a.reduce((s,e)=>s+e,0)
      }
   arr=[1,5,2]
   console.log("THE SUM:",sum(...arr));
   </script>
</body>
</html>
```

OUTPUT

```
THE SUM: 8
> |
```

TASK 8

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        let object={
            name:"san",
            rollno:321
        };
        console.log(JSON.stringify(object));

    </script>
</body>
</html>
```

OUTPUT

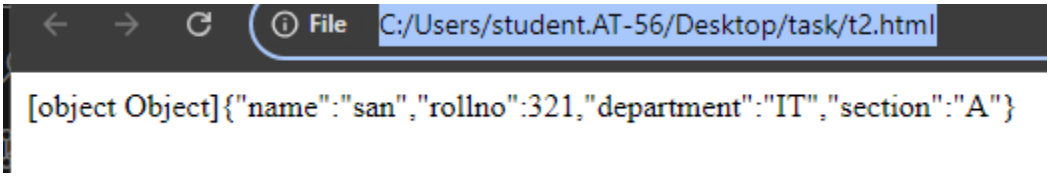
```
{"name":"san","rollno":321}
> |
```

TASK 9

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        let student={
            name:"san",
            rollno:321
        };
        let dept={
            department:"IT",
            section:"A"
        };
        let details={
            ...student,...dept
        }
        document.write(details)
        var sv=JSON.stringify(details);
```

```
        document.write(sv)
    </script>
</body>
</html>
```

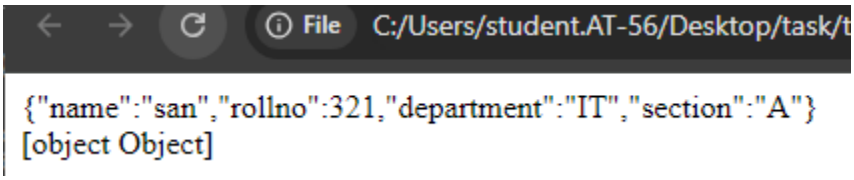
OUTPUT



TASK 10

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        let student={
            name:"san",
            rollno:321
        };
        let dept={
            department:"IT",
            section:"A"
        };
        let details={
            ...student,...dept
        }
        var sv=JSON.stringify(details);
        document.write(sv)
        var sv=JSON.parse(sv);
        document.write("<br>",sv)
    </script>
</body>
</html>
```

OUTPUT

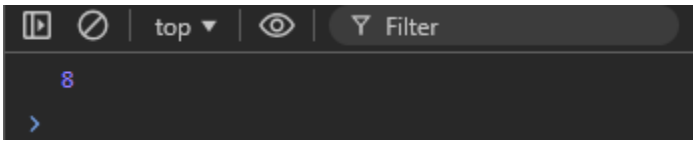


TASK 11

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
```

```
    <script>
        var one=(x)=>{
            return(y)=>x+y
        }
        var two=one(5)
        console.log(two(3));
    </script>
</body>
</html>
```

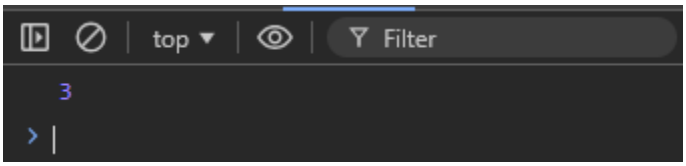
OUTPUT



TASK 12

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
      function counter(){
        var count=0;
        return{
            increment:function(){
            count++
        },
            gcount:function(){
                return count
            }
        };
      }
      let cc=counter();
      cc.increment()
      cc.increment()
      cc.increment()
      console.log(cc.gcount());
    </script>
</body>
</html>
```

OUTPUT

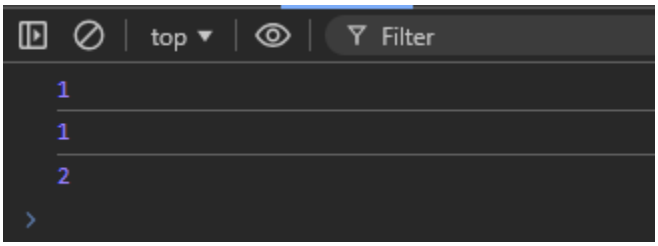


TASK 13

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
      function counter(){
        var count=0;
        return function increment(){
            count++
            console.log(count,);
        }
      }
      let cc=counter();
      cc()
      let mc=counter()
      mc()
      mc()
    </script>
</body>
</html>
```

OUTPUT

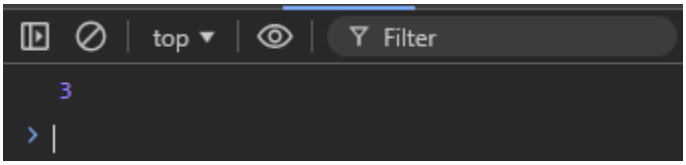


TASK 14

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
      function counter(){
        var count=0;
        return{
            increment:function(){
            count++
        },
            gcount:function(){
                return count
            }
        };
      }
      let cc=counter();
      cc.increment()
```

```
    cc.increment()
    cc.increment()
    console.log(cc.gcount());
  </script>
</body>
</html>
```

OUTPUT

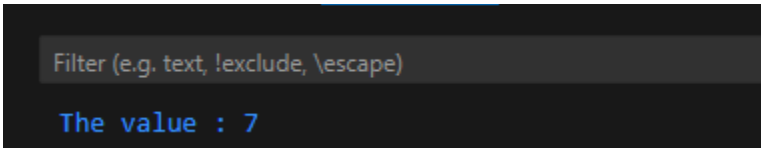


TASK 15

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>TASK</title>
</head>
<body>
  <script>
     function cal(value){
     if(value=="add")
     return (x,y)=>x+y;
     if(value=="sub")
     return (x,y)=>x-y;
    }

    let c=cal("add");
    console.log("The value :",c(5,2));
  </script>
</body>
</html>
```

OUTPUT

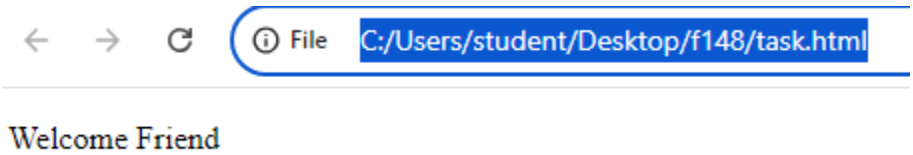


TASK 16

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>TASK</title>
</head>
<body>
  <p id="d"></p>
  <script>
  let prom = new Promise(function (resolve, reject) {
     setTimeout(function() {
        resolve("Welcome Friend");
```

```
        }, 2000);
    });
    prom.then(
        function(val) { show(val); },
        function(error) { show(error); }
    );
    function show(val) {
        document.getElementById("d").innerHTML = val;
    }
    </script>
</body>
</html>
```

OUTPUT

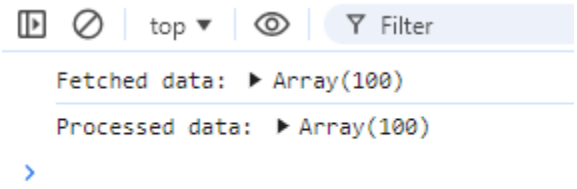


TASK 17

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>TASK</title>
</head>
<body>
    <p id="d"></p>
    <script>
fetch('https://jsonplaceholder.typicode.com/posts')
  .then(response => {
    if (!response.ok) {
      throw new Error('Network response was not ok');
    }
    return response.json();
  })
  .then(data => {
    return new Promise((resolve, reject) => {
      const processedData = data.map(post => ({
        title: post.title,
        summary: post.body.substring(0, 50)
      }));
      resolve(processedData);
    });
  })
  .then(processedData => {
    console.log('Processed data:', processedData);
  })
  .catch(error => {
    console.error('There was an error:', error);
  });
```

```
    </script>
</body>
</html>
```

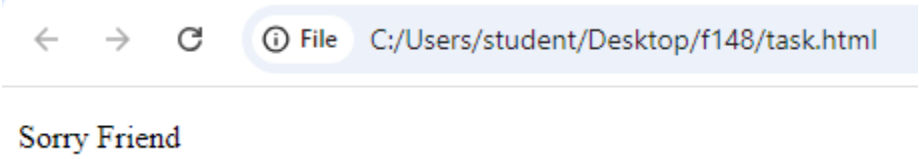
OUTPUT



TASK 18

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>TASK</title>
</head>
<body>
    <p id="d"></p>
    <script>//odd possible,even not possible
        let prom = new Promise(function (resolve, reject) {
            let reach=Number(prompt("Enter a Number:"))
        if(reach%2){
            setTimeout(function() {
            resolve("Welcome Friend");
        }, 2000);  }else{
        setTimeout(function() {
            reject("Sorry Friend");
        }, 2000);
    }
    });
    prom.then(
        function(val) { show(val); },
        function(error) { show(error); }
    );
    function show(val) {
        document.getElementById("d").innerHTML = val;
    }
    </script>
</body>
</html>
```

OUTPUT



Sorry Friend

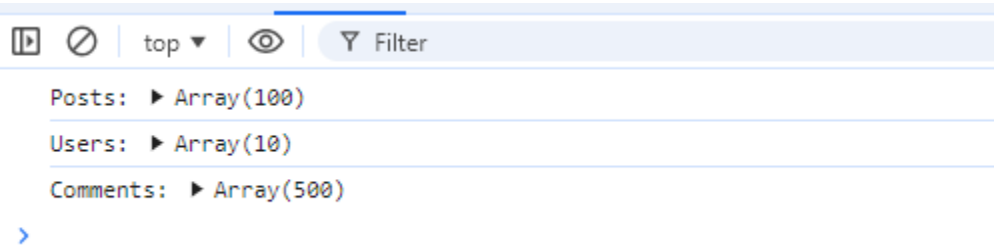
TASK 19

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>TASK</title>
</head>
<body>
    <p id="d"></p>
    <script>
const apiUrls = [
  'https://jsonplaceholder.typicode.com/posts',
  'https://jsonplaceholder.typicode.com/users',
  'https://jsonplaceholder.typicode.com/comments'
];
const promises = apiUrls.map(url => fetch(url).then(response => {
  if (!response.ok) {
    throw new Error(`Error fetching ${url}: ${response.statusText}`);
  }
  return response.json();
}));
Promise.all(promises)
  .then(results => {
    const posts = results[0];
    const users = results[1];
    const comments = results[2];

    console.log('Posts:', posts);
    console.log('Users:', users);
    console.log('Comments:', comments);
  })
  .catch(error => {
    console.error('Error:', error);
  });
    </script>
</body>
</html>
```

OUTPUT



TASK 20

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Promise Chain Example</title>
</head>
```

```
<body>
  <script>
    function fetchData() {
      return new Promise((resolve) => {
        setTimeout(() => {
          const data = { id: 1, name: "John" };
          console.log('Fetched data:', data);
          resolve(data);
        }, 1000);
      });
    }
    function processData(data) {
      return new Promise((resolve) => {
        setTimeout(() => {
          data.name = data.name.toUpperCase();
          console.log('Processed data:', data);
          resolve(data);
        }, 1000);
      });
    }
    function logResult(data) {
      return new Promise((resolve) => {
        setTimeout(() => {
          console.log('Final result:', data);
          resolve('Process complete');
        }, 1000);
      });
    }
    fetchData()
      .then(data => processData(data))
      .then(processedData => logResult(processedData))
      .then(result => console.log(result))
      .catch(error => console.error('Error:', error));
  </script>
</body>
</html>
```

OUTPUT

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL
> Fetched data: {id: 1, name: 'John'}
> Processed data: {id: 1, name: 'JOHN'}
> Final result: {id: 1, name: 'JOHN'}
  Process complete
```

TASK 21

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>JavaScript Quiz</title>
</head>
<body>
```

```
  <script>
    orderfood = (order) => {
      document.write(`Getting Order!........`)
      document.write("<br>")
      return new Promise((resolve)=>{
        setTimeout(() => {
          document.write(`Order is placed for ${order}`)
          document.write("<br>")
          resolve(order)
        }, 1000);
      })
    }
    preparefood = (order) => {
      document.write(`Your Food ${order} is Preparing!........`)
      document.write("<br>")
      return new Promise((resolve)=>{
        setTimeout(() => {
          document.write(`Your Food ${order} is Prepared!.....`)
          document.write("<br>")
          resolve(order)
        }, 1000);
      })
    }
    deliverfood = (order) => {
      document.write(`Getting Order to delivery!........`)
      document.write("<br>")
      return new Promise((resolve)=>{
        setTimeout(() => {
          document.write(`${order} is delivered`)
          document.write("<br>")
          resolve(order)
        }, 1500);
      })
    }
    food = async(value) => {
      const order = await orderfood(value)
      const Prepare = await preparefood(order)
      const deliver = await deliverfood(Prepare)
      if(deliver == order){
        document.write("Thank You!......");
      }else{
        document.write("Please Wait!...")
      }
    }
    value = prompt("Enter the Dish you desire: ")
  food(value)
  </script>
</body>
</html>
```

OUTPUT

```
Order is placed for CHICKEN BIRIYANI
Your Food CHICKEN BIRIYANI is Preparing!.....
Your Food CHICKEN BIRIYANI is Prepared!.....
Getting Order to delivery!.....
CHICKEN BIRIYANI is delivered
Thank You!.....
```

TASK 22

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>JavaScript</title>
</head>
<body>
  <script>
   async function fetchData(url) {
  try {
    const response = await fetch(url);
    if (!response.ok) {
      throw new Error('Network response was not ok');
    }
    const data = await response.json();
    return data;
  } catch (error) {
    console.error('There was a problem with the fetch operation:', error);
  }
}
function processData(data) {
  if (data && typeof data === 'object') {
    for (const [key, value] of Object.entries(data)) {
      console.log(`${key}: ${value}`);
    }
  } else {
    console.log('Received non-object data:', data);
  }
}
async function main() {
  const url = 'https://jsonplaceholder.typicode.com/todos/1';
  const data = await fetchData(url);
  if (data) {
    processData(data);
  }
}
main();
  </script>
</body>
</html>
```

OUTPUT

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL
  userId: 1
  id: 1
  title: delectus aut autem
  completed: false
```

TASK 23

```
<!DOCTYPE html>
<head>
   <title>TASK</title>
</head>
<body>
   <script>
      var s
      async function fun(a){
         var promise= new Promise((resolve, reject) => {
      if(a==0){
 setTimeout(() => {
   resolve('Success!');
 }, 1000);}
 else{
   reject("error")
 }
});try{
   var s= await promise
   document.write(s)
}catch(error){
   console.log(error);
}
 }
 fun(10)
   </script>
</body>
</html>
```

OUTPUT

```
OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
  error
```

TASK 24

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>JavaScript</title>
</head>
<body>
  <script>
  async function fetchData(url) {
  const response = await fetch(url);
```

```
  if (!response.ok) {
    throw new Error('Network response was not ok');
  }
  return response.json();
}
async function main() {
  const urls = [
    'https://jsonplaceholder.typicode.com/todos/1',
    'https://jsonplaceholder.typicode.com/todos/2',
    'https://jsonplaceholder.typicode.com/todos/3'
  ];

  try {
    const results = await Promise.all(urls.map(url => fetchData(url)));
    results.forEach((data, index) => {
      console.log(`Data from URL ${urls[index]}:`);
      console.log(data);
    });
  } catch (error) {
    console.error('There was an error:', error);
  }
}
main();
  </script>
</body>
</html>
```

OUTPUT

```
OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS                    Filter (e.g. text, !exc
  Data from URL https://jsonplaceholder.typicode.com/todos/1:
> {userId: 1, id: 1, title: 'delectus aut autem', completed: false}
  Data from URL https://jsonplaceholder.typicode.com/todos/2:
> {userId: 1, id: 2, title: 'quis ut nam facilis et officia qui', completed: false}
  Data from URL https://jsonplaceholder.typicode.com/todos/3:
> {userId: 1, id: 3, title: 'fugiat veniam minus', completed: false}
```

TASK 25

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>JavaScript Quiz</title>
</head>
<body>
  <script>
    orderfood = (order) => {
      document.write(`Getting Order!........`)
      document.write("<br>")
      return new Promise((resolve)=>{
        setTimeout(() => {
          document.write(`Order is placed for ${order}`)
          document.write("<br>")
          resolve(order)
```

```
      }, 1000);
    })
  }
  preparefood = (order) => {
    document.write(`Your Food ${order} is Preparing!........`)
    document.write("<br>")
    return new Promise((resolve)=>{
      setTimeout(() => {
        document.write(`Your Food ${order} is Prepared!.....`)
        document.write("<br>")
        resolve(order)
      }, 1000);
    })
  }
  deliverfood = (order) => {
    document.write(`Getting Order to delivery!........`)
    document.write("<br>")
    return new Promise((resolve)=>{
      setTimeout(() => {
        document.write(`${order} is delivered`)
        document.write("<br>")
        resolve(order)
      }, 1500);
    })
  }
  food = async(value) => {
    const order = await orderfood(value)
    const Prepare = await preparefood(order)
    const deliver = await deliverfood(Prepare)
    if(deliver == order){
      document.write("Thank You!.....");
    }else{
      document.write("Please Wait!...")
    }
  }
  value = prompt("Enter the Dish you desire: ")
 food(value)
 </script>
</body>
</html>
```

OUTPUT

```
Order is placed for CHICKEN BIRIYANI
Your Food CHICKEN BIRIYANI is Preparing!........
Your Food CHICKEN BIRIYANI is Prepared!.....
Getting Order to delivery!........
CHICKEN BIRIYANI is delivered
Thank You!.....
```