ARROW FUNCTION

Traditional Function

Traditional Juice machine

Arrow Function

Advanced Juice Machine

Arrow functions

Traditional Function Syntax:

```
Function   functionName(parameters) {
                                // code to be executed
                                Return => Output value
}
```

Arrow Function Syntax:

```
let   functionName = (parameters) => {
                                // code to be executed
                                Return => Output value
}
```

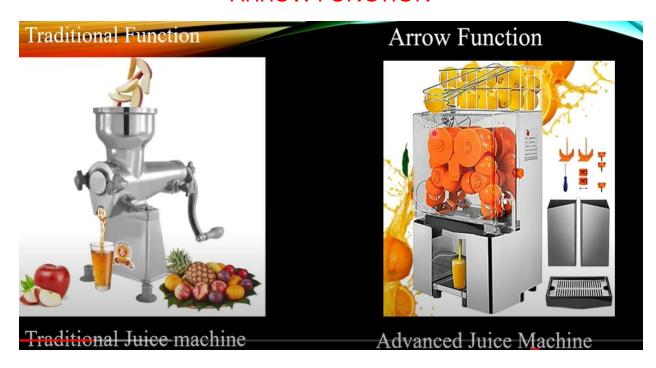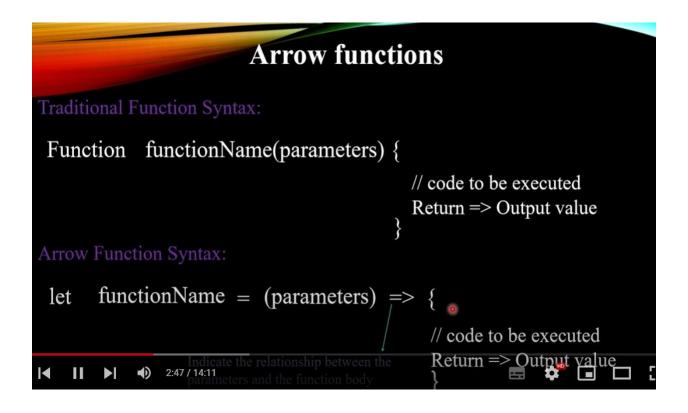Indicate the relationship between the
parameters and the function body

1)

```
ArrowFunction.js > ...
function addNumbers(x, y) {
    return x + y;
}
let result = addNumbers(5, 6);
console.log(result);
```

```
[Running] node "g:\W
11
```

Writing above one in Arrow function

```
let addNumbers = (x, y) => {
    return x + y;
};
let result = addNumbers(5, 6);
console.log(result);
```

```
[Running] node "g:\WEB JA
11
```

2) this(keyword) is accepted by traditional function

this is not accepted by Arrow function

implementation

```
let details = {
  names: "varshith",
  welcome: function () {
    console.log("welcome " + this.names); // here  **this** just refers to details
  },
  coming: () => {
    console.log("bye bye : " + details.names); //here if we use **this** we will get an error so we using details.names
  },
};

details.welcome();
details.coming();
```

```
[Running] node "g:\WEB JAVASCRIPT PRACTICE\22.ArrowFunction.js"
welcome varshith
bye bye : varshith
```

3) **return is compulsory in traditional function
   **return is not compulsory in Arrow function if we not written return in Arrow function then also we get output and only for single line code and we should not write it in brackets
by seeing down example we can understand

Implementation:--

```javascript
function multiplyA(x, y) {
  x * y; // output : undefined
}
let multiplyB = (x, y) => x * y; // output:30

console.log(multiplyA(5, 6));
console.log(multiplyB(5, 6));
```

```
[Running] node "g:\WEB JAVASCRIPT PRACTICE\22.ArrowFunction.js"
undefined
30
```