Slot 5.2  -Hàm mũi tên, callback

==========

onPress={increment()}
=> gọi mỗi lần render, không cần press
onPress={()=>increment()}
=> gọi mỗi khi press (callback)

========

Hàm mũi tên:
-sử dụng để truyền callback vào các sự kiện như onPress, onClick
-sử dụng để truyền tham số cho hàm sự kiện
ví dụ: Truyền vào hàm không có tham số:

```
import { Button } from "react-native";

const handlePress = () =>{
    console.log("Button vua duoc press");
};
<Button title="Press me" onPress={handlePress}/>
```

Ví dụ về cách truyền vào hàm có tham số:

```
const handlePress1 = (id) =>{
    console.log(`Button vua duoc press: ${id}`);
};
<Button title="Press me" onPress={()=>handlePress1(1)}/>
```

---

Đặc điểm của hàm mũi tên:
1.không có ngữ cảnh this
=> không cần xử ký **bind** trong constructor

2. dùng cho việc xử lý các sự kiện hoặc xử lý trong cac hook: useState, useEffect

```
const MyCom = () =>{
    const [count,setCount] = useState(0);
    const increment = () =>{
        setCount(count+1);
    };
    return(
        <Button title="Click me" onPress={()=>increment}/>
    );
};
```

3. Khi nào không dùng hàm mũi tên:

*Mỗi khi hàm mũi tên được gọi → render lại → mất nhiều bộ nhớ

=> cần tối ưu hóa trong việc sử dụng callback (useCallback) hoặc cần tách hàm mũi tên ra bên ngoài

*Hàm mũi tên không được gọi trong hàm khởi tạo

* không dùng từ khóa **new** với hàm mũi tên

4. Tham số của hàm mũi tên:

4.1 Tham số là 1 biến:

```
const greet = (name) =>{
    console.log(`Xin chao ${name}`);
};
greet("Nguyen Van A");
```

4.2 Tham số là 1 đối tượng:

```
const inThongTin = (person)=>{
    console.log(`Name: ${person.name}, Age: ${person.age}`);
};
const person = {name: "An", Age: 20};
inThongTin(person);
```

## 4.3 Tham số là mảng:

```
const hamTongMang = (numbers) =>{
    return numbers.reduce((total,num)=>total+num,0);
};
const numbers = [1,2,3,4,5];
console.log(hamTongMang(numbers));//15
```

## 4.4 Tham số là 1 callback:

Hàm gọi đến hàm ở bên trong

```
const executeCallback = (callback) =>{
    console.log("Truoc khi callback");
    callback();
    console.log("Sau khi callback");
};
executeCallback(()=> console.log("Day la ham callback"));
```

## 4.5. Tham số là đối tượng với cấu trúc phân rã (destructuring)

ý nghĩa: trích xuất 1 số thuộc tính từ 1 đối tượng

```
//----
const printPersonInfo = ({name,age})=>{
    console.log(`Ten: ${name}, Tuoi: ${age}`);
};

const person1 = {name: 'NVA', age: 20};
printPersonInfo(person1);
```

## 4.6 Tham số là mảng với cấu trúc phân rã Destructuring

Ý nghĩa: khi mảng quá lớn, mà ta chỉ cần sử dụng 1 hoặc vài phần tử của mảng

```
//-------
const inMotVaiPhanTuMang = ([first,second]) =>{
    console.log(`First: ${first}, Second: ${second}`);
};
const mang = [10,20,30,40,50,60,70,80];
inMotVaiPhanTuMang(numbers);//10,20
```

4.7. truyền danh sách các tham số ...args

ý nghĩa: khi các tham số truyền không cố định về mặt số lượng thì dùng ...args

```
const sum = (...numbers)=>{
    return numbers.reduce((total,num)=>total+num,0);
};
console.log(sum(1,2));
console.log(sum(1,2,3,4));
```

====

Hàm callback

-Hàm callback thường được truyền vào hàm khác như 1 tham số (là tham số của 1 hàm khác)

-sử dụng khi muốn thực hiện 1 hành động **sau khi 1 hành động khác kết thúc**

- Hay dùng trong việc xử lý sự kiện, xử lý đồng bộ và bất đồng bộ, sử dụng trong các API

========