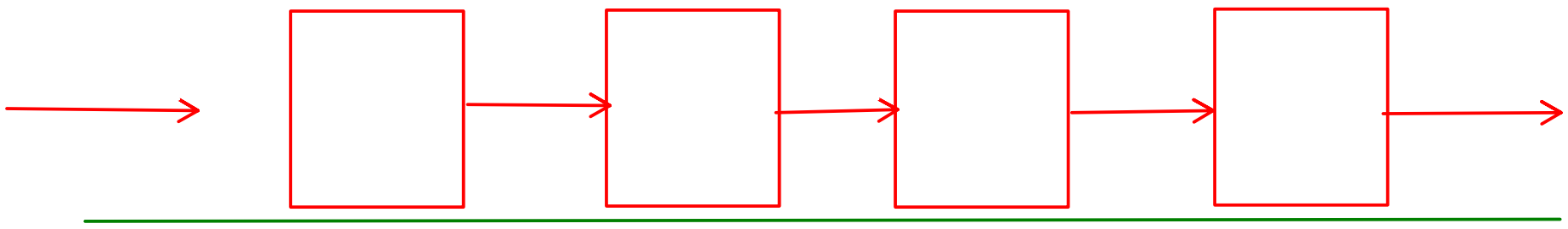
Java 8 interface

Lambda Expression

```
class subclass implements Comparator{
        int compare(o1,o2){

        }
}
sort((o1,o2)-> o1-o2 );
```

```
BinaryOperator op = Integer::sum
System.out::println;
```

Stream
-Pipeline of Operations

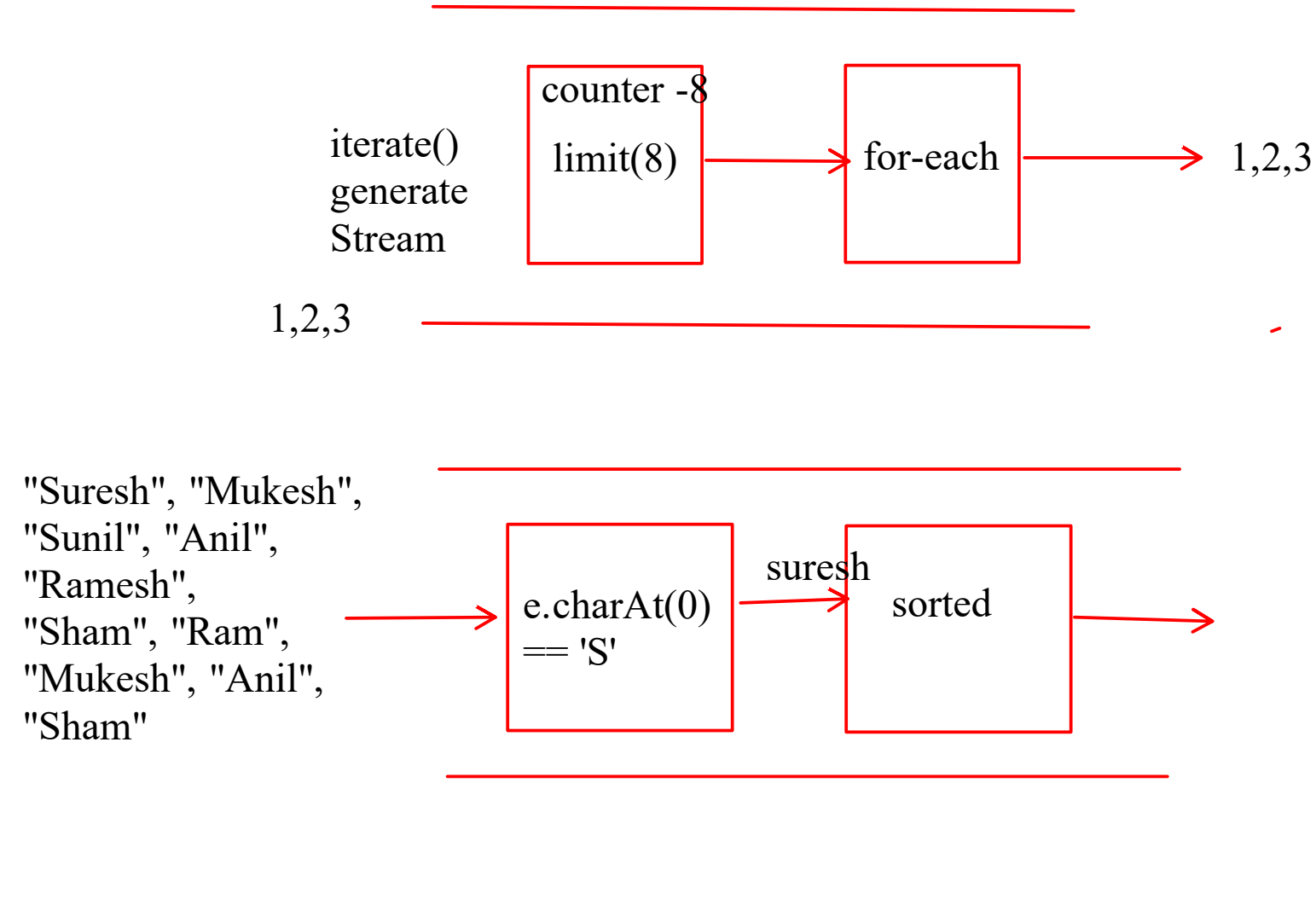


Two Type of Operations

1. Intermediate Operation
- That generates another Stream
- It is again categorized into 2 subtypes
    a. Stateless Operations
    b. Stateful Operation

2. Terminal Operation
- That produce result
- These do not generate another stream

```
        stream();

for(){
if()
    sysout
}
```



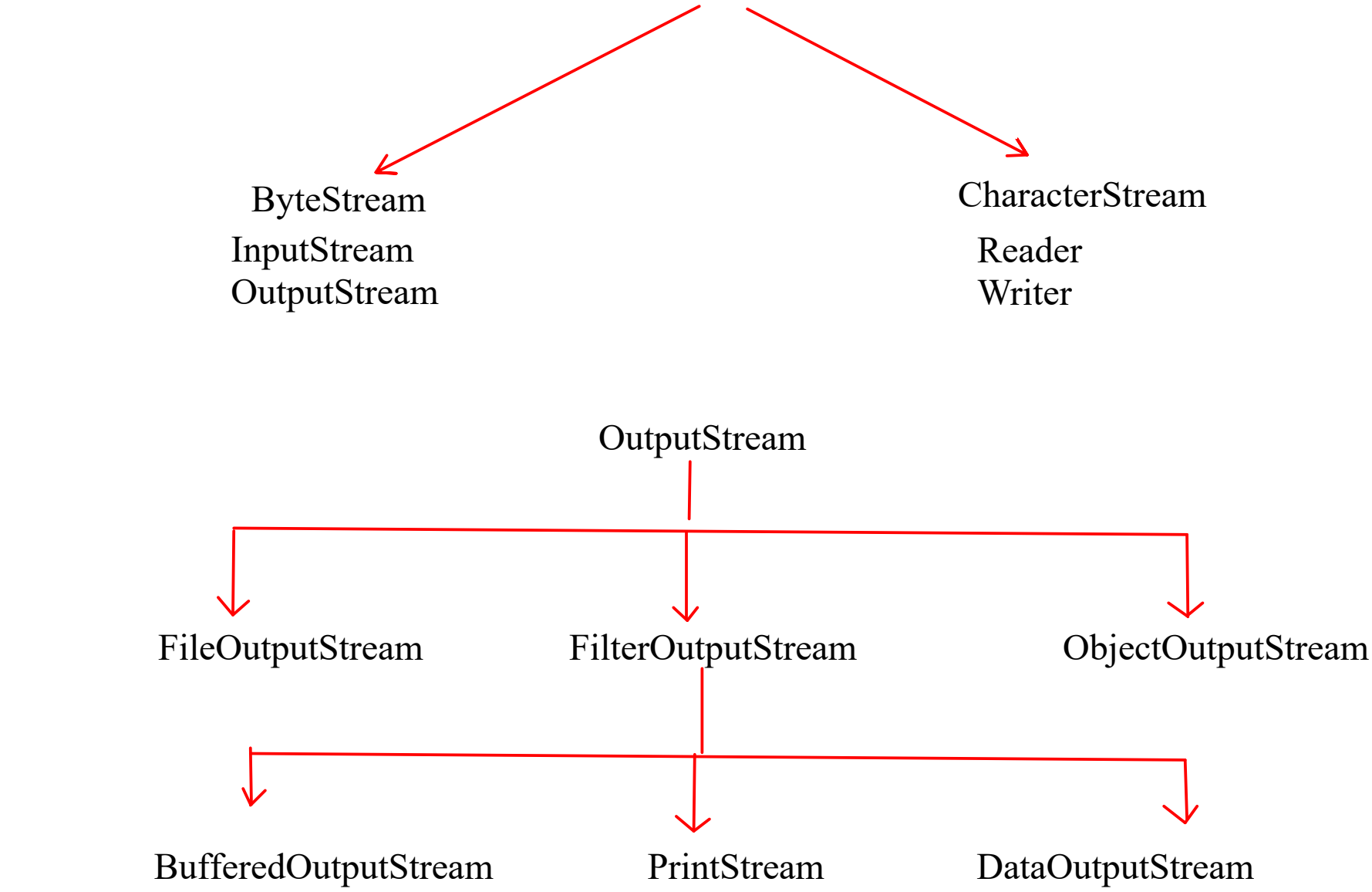
of
generate
iterate
foreach
count
limit
skip
sorted()
sorted(Comparator c)
distinct
filter
map

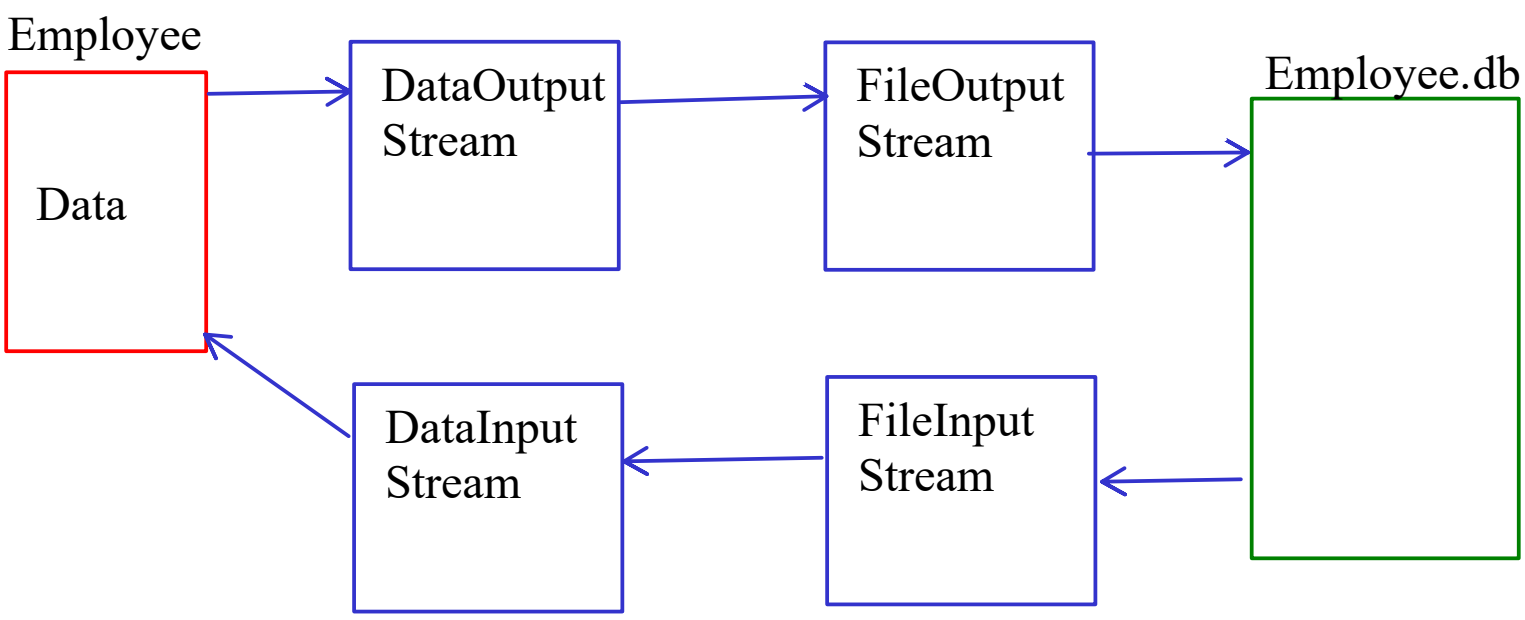
reduce,collect

```
interface Stream{
static Stream of(T ... val){
}

static Stream iterate(T seed,){
}


static Stream generate(){
}


}
```

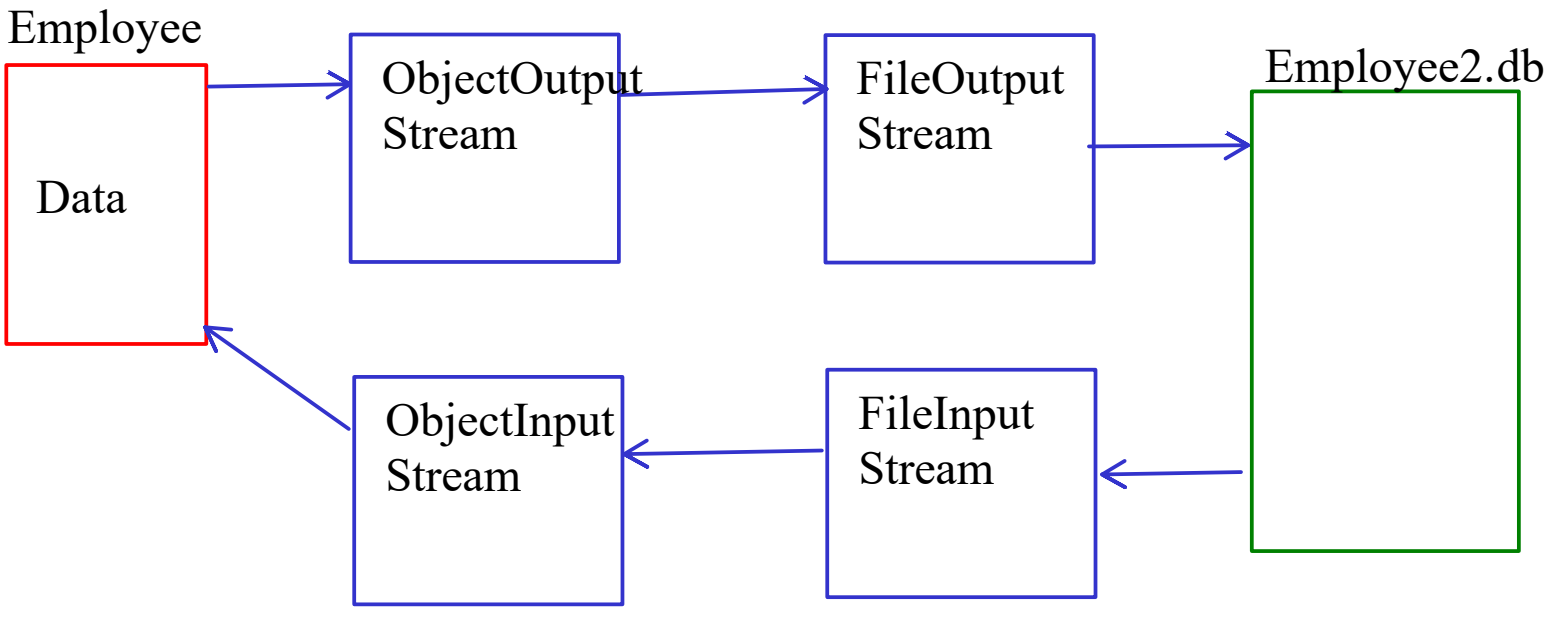
```
interface Shape{

static Shape shapeFactory(int choice){
        return new Circle
}
}
```

File IO






```
FileOutputStream fos = new FileOutputStream("employee.db");
DateOutputStream dos = new DataOutputStream(fos);
 Employee e = new Employee(1,"Anil",10000);
 dos.writeInt(e.empid);
```




```
FileOutputStream fos = new FileOutputStream("employee2.db");
ObjectOutputStream oos = new ObjectOutputStream(fos);
```