computeIfAbsent(key, func)

Purpose: Map me agar key **nahi hai**, to pehle key create karta hai aur uski value func se assign karta hai.

Phir **us value ko return** karta hai (chahe purani ho ya nayi).



Step by step:

- 1. Agar neighbor key **nahi hai** map me \rightarrow
 - \rightarrow k -> new HashSet<>() chalega \rightarrow ek empty HashSet banega
 - \rightarrow map me entry add ho jaayegi \rightarrow neighbor = new HashSet<>()
- 2. Phir .add(node) karega → us set me node add ho jaayega
- Agar neighbor pehle se hai → to uska existing set le lega aur usme node add karega
- So yes yeh key absent ho to create karta hai, aur hamesha data add karta hai.
- getOrDefault(key, defaultValue)

Purpose: Map me agar key **present hai**, to uski value return karo; agar **absent hai**, to defaultValue return karo (lekin map me add nahi karta).

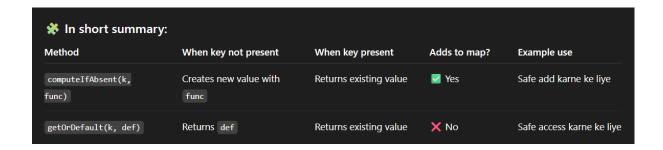
```
java

D Copy code

prereqMap.get(neighbor).addAll(prereqMap.getOrDefault(node, new HashSet<>()));
```

Meaning:

- neighbor ka set mil gaya (guaranteed present hai).
- node ka set lene ki koshish agar nahi hai, to ek temporary empty set mil gaya.
- Fir addAll() se node ke saare prerequisites neighbor me add ho gaye.



Tera one-line explanation 👇

✓ computeIfAbsent → "agar key nahi hai to banao aur use karo"
✓ getOrDefault → "agar key nahi hai to temporary default value do, map mat badlo"

Step 1: Function<T, R> kya hai?

Function<T, R> ek built-in functional interface hai jo Java 8 se aaya hai (java.util.function package me).

Iska matlab:

"Ek aisa function jo ek input le (type T) aur ek output de (type R)."

```
java

@FunctionalInterface
public interface Function<T, R> {
    R apply(T t);
}
```

Simple words me:

```
Agar likho
Function<Integer, String>
to iska matlab hai:
```

"Yeh function ek Integer input lega aur ek String return karega."

Without lambda (Anonymous Class version)

```
java

Function<Integer, Set<Integer>> func = new Function<Integer, Set<Integer>>() {
    @Override
    public Set<Integer> apply(Integer k) {
        return new HashSet<>();
    }
};

Yahan tum Function interface ka ek anonymous implementation bana rahe ho.

"Anonymous class" = a class without a name that implements an interface (or extends a class) and you create its object immediately.
```

Lambda version

Lambda version likhne ka shortcut hota hai:

```
java

☐ Copy code

Function<Integer, Set<Integer>> func = k -> new HashSet<>();

Yahan Java automatically samajh jaata hai:

• Input type → Integer

• Return type → Set<Integer>

• Method → apply()

Internally yeh likhne ke barabar hai ♣
```

Summary		
Concept	Туре	Purpose
Function <t, r=""></t,>	Interface	Represents a function taking T and returning R
<pre>new Function<>() { }</pre>	Anonymous class	Old-style implementation
k -> new HashSet<>()	Lambda	Short, modern version of same thing
Used in	<pre>computeIfAbsent , map , stream.map() , etc.</pre>	