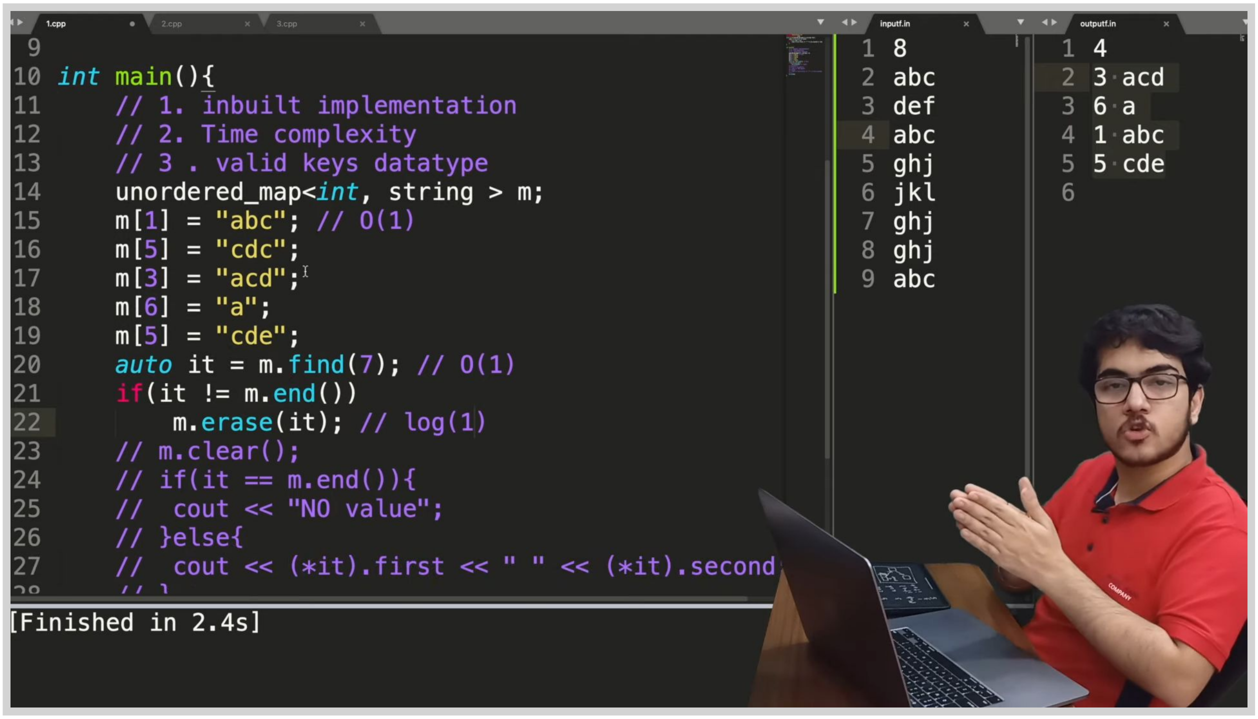
←

Lecture 26
Part 2: Unordered Maps & Multimaps

- **unordered_map complexity** : in any case ether its find , put , get ,anything is O(1) ;

thats way all time we use **unordered_map**

- Unordered_map as key not allowed container's like map<<vector> , int> is not allowed .
- **multimap** : it is used only that we store the duplicate key's in the map .



```
int main(){
    // 1. inbuilt implementation
    // 2. Time complexity
    // 3 . valid keys datatype
    unordered_map<int, string > m;
    m[1] = "abc"; // O(1)
    m[5] = "cdc";
    m[3] = "acd";
    m[6] = "a";
    m[5] = "cde";
    auto it = m.find(7); // O(1)
    if(it != m.end())
        m.erase(it); // log(1)
    // m.clear();
    // if(it == m.end()){
    //   cout << "NO value";
    // }else{
    //   cout << (*it).first << " " << (*it).second
```

```
[Finished in 2.4s]
```

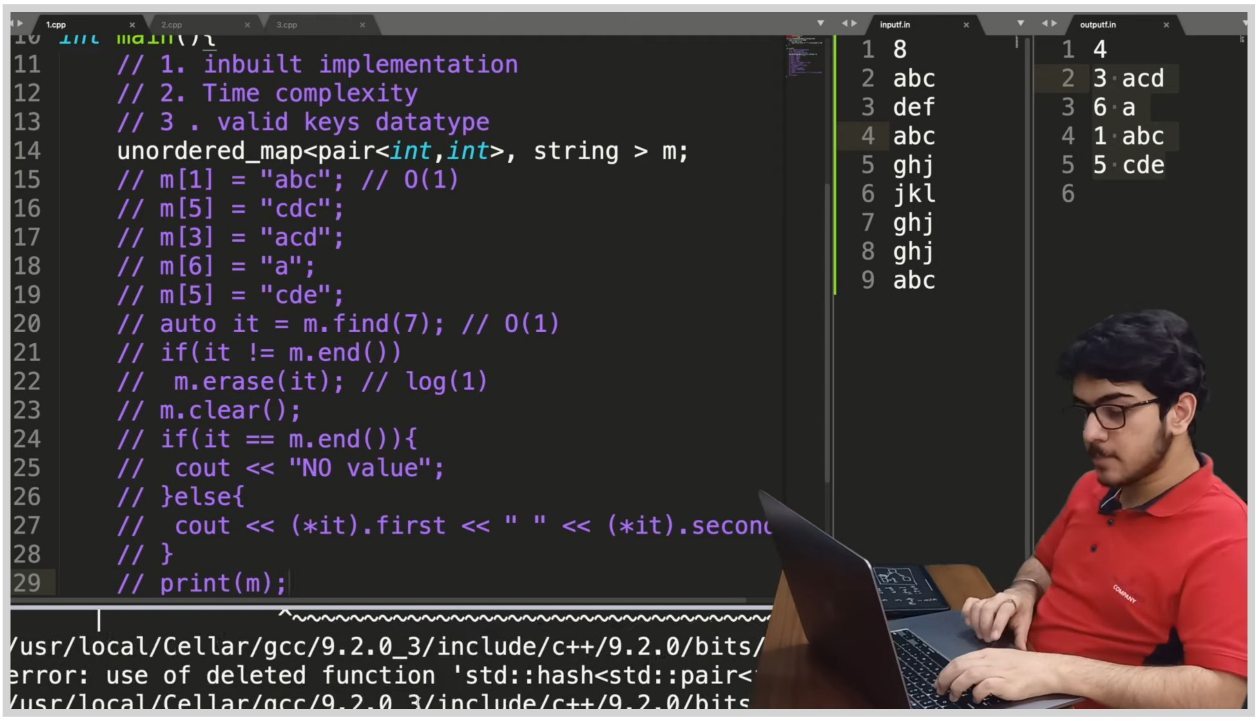
input.in:

```
8
abc
def
abc
ghj
jkl
ghj
ghj
abc
```

output.in:

```
4
3 acd
6 a
1 abc
5 cde
```

container's not allowed as a key.

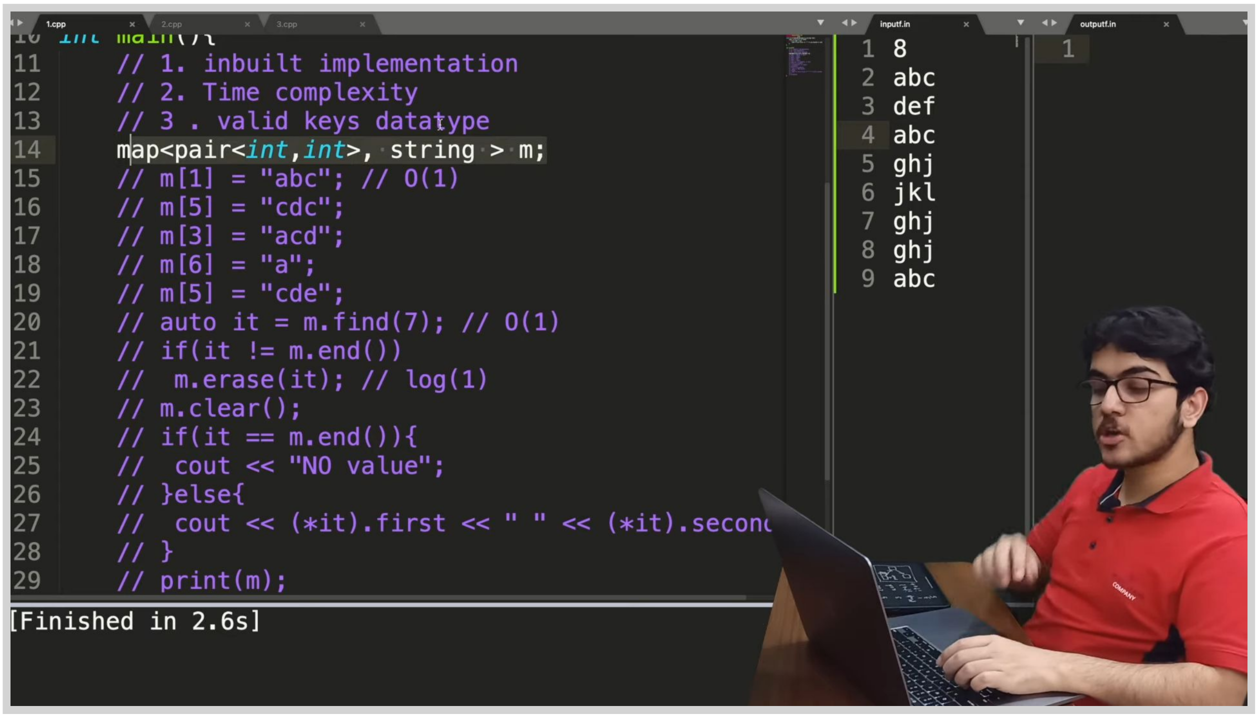


```
int main(){
    // 1. inbuilt implementation
    // 2. Time complexity
    // 3 . valid keys datatype
    unordered_map<pair<int,int>, string > m;
    // m[1] = "abc"; // O(1)
    // m[5] = "cdc";
    // m[3] = "acd";
    // m[6] = "a";
    // m[5] = "cde";
    // auto it = m.find(7); // O(1)
    // if(it != m.end())
    //  m.erase(it); // log(1)
    // m.clear();
    // if(it == m.end()){
    //  cout << "NO value";
    // }else{
    //  cout << (*it).first << " " << (*it).seconc
    // }
    // print(m);
```

```
8
abc
def
abc
ghj
jkl
ghj
ghj
abc
```

```
4
3 acd
6 a
1 abc
5 cde
```

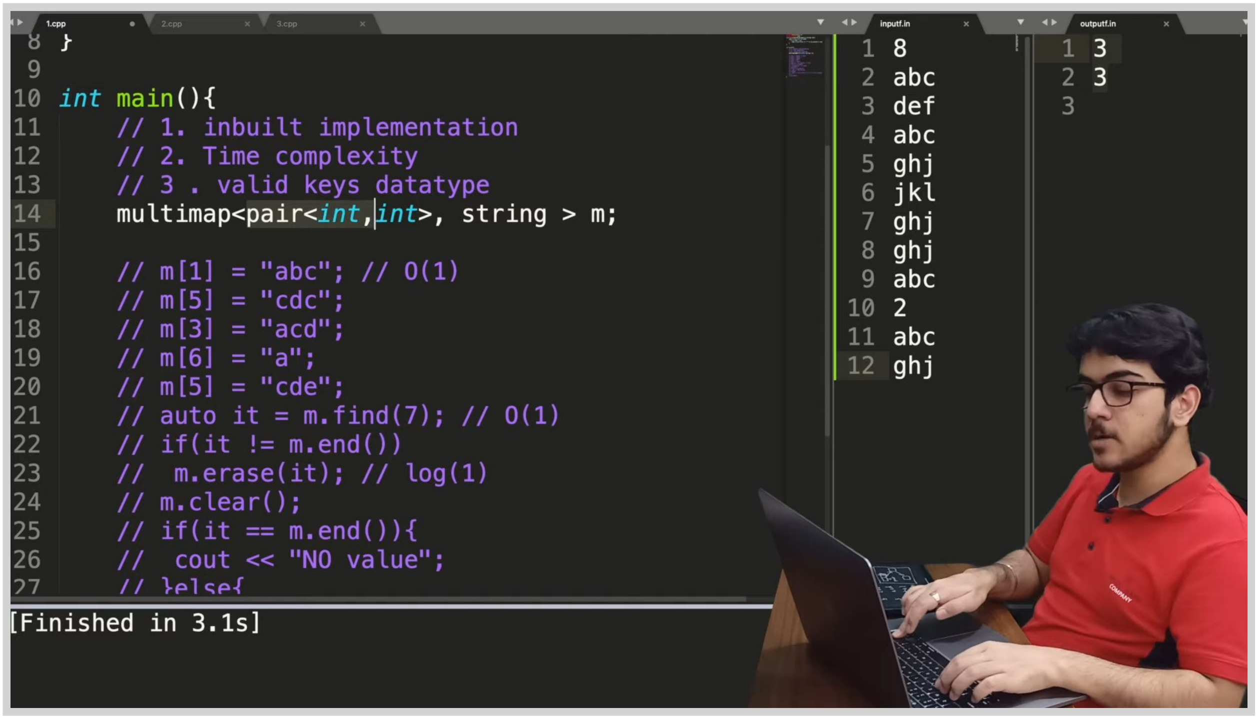
```
/usr/local/Cellar/gcc/9.2.0_3/include/c++/9.2.0/bits/
error: use of deleted function 'std::hash<std::pair<
/usr/local/Cellar/gcc/9.2.0_3/include/c++/9.2.0/bits
```



```
int main(){
    // 1. inbuilt implementation
    // 2. Time complexity
    // 3 . valid keys datatype
    map<pair<int,int>, string > m;
    // m[1] = "abc"; // O(1)
    // m[5] = "cdc";
    // m[3] = "acd";
    // m[6] = "a";
    // m[5] = "cde";
    // auto it = m.find(7); // O(1)
    // if(it != m.end())
    //  m.erase(it); // log(1)
    // m.clear();
    // if(it == m.end()){
    //  cout << "NO value";
    // }else{
    //  cout << (*it).first << " " << (*it).seconc
    // }
    // print(m);
```

```
8
abc
def
abc
ghj
jkl
ghj
ghj
abc
```

```
[Finished in 2.6s]
```

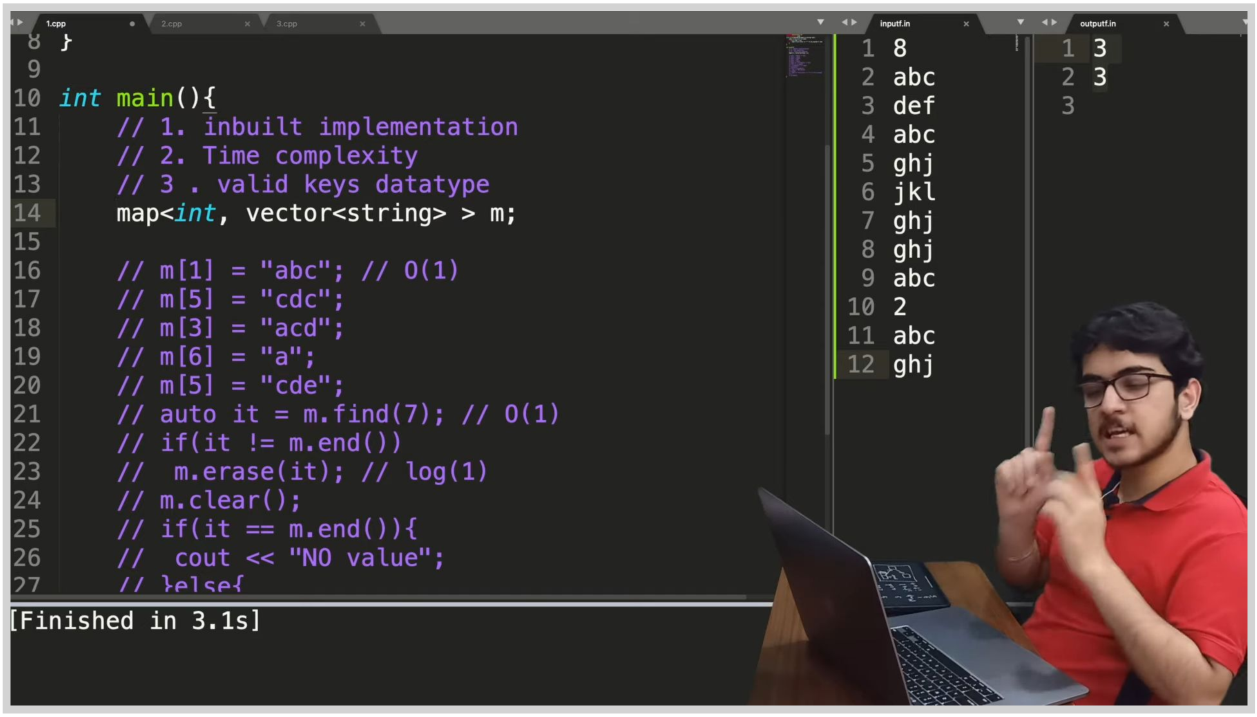


```
}

int main(){
    // 1. inbuilt implementation
    // 2. Time complexity
    // 3 . valid keys datatype
    multimap<pair<int,int>, string > m;

    // m[1] = "abc"; // O(1)
    // m[5] = "cdc";
    // m[3] = "acd";
    // m[6] = "a";
    // m[5] = "cde";
    // auto it = m.find(7); // O(1)
    // if(it != m.end())
    //  m.erase(it); // log(1)
    // m.clear();
    // if(it == m.end()){
    //  cout << "NO value";
    // }else{
[Finished in 3.1s]
```

```
8
abc
def
abc
ghj
jkl
ghj
ghj
abc
2
abc
ghj
```

```
3
3
```

that's way to replace the multimap



```
}

int main(){
    // 1. inbuilt implementation
    // 2. Time complexity
    // 3 . valid keys datatype
    map<int, vector<string> > m;

    // m[1] = "abc"; // O(1)
    // m[5] = "cdc";
    // m[3] = "acd";
    // m[6] = "a";
    // m[5] = "cde";
    // auto it = m.find(7); // O(1)
    // if(it != m.end())
    //  m.erase(it); // log(1)
    // m.clear();
    // if(it == m.end()){
    //  cout << "NO value";
    // }else{
[Finished in 3.1s]
```

```
8
abc
def
abc
ghj
jkl
ghj
ghj
abc
2
abc
ghj
```

```
3
3
```