**Answer question2**

**The code :**

import java.util.\*;

public class OptimalPageReplacement {

public static void main(String[] args) {

int[] referenceString = {1, 0, 2, 0, 3, 2, 1, 0, 1, 2, 3, 2, 1, 0, 2};

int frames = 3;

Set<Integer> memory = new HashSet<>();

List<Integer> frameList = new ArrayList<>();

int pageFaults = 0;

for (int i = 0; i < referenceString.length; i++) {

int page = referenceString[i];

if (!memory.contains(page)) {

pageFaults++;

if (memory.size() < frames) {

memory.add(page);

frameList.add(page);

} else {

// Optimal replacement: find the page not used for longest time in future

int indexToReplace = -1, farthest = i + 1;

for (int j = 0; j < frameList.size(); j++) {

int nextUse = Integer.MAX\_VALUE;

for (int k = i + 1; k < referenceString.length; k++) {

if (referenceString[k] == frameList.get(j)) {

nextUse = k;

break;

}

}

if (nextUse > farthest) {

farthest = nextUse;

indexToReplace = j;

}

}

if (indexToReplace == -1) {

// None used again, replace the first

memory.remove(frameList.get(0));

frameList.remove(0);

} else {

memory.remove(frameList.get(indexToReplace));

frameList.remove(indexToReplace);

}

memory.add(page);

frameList.add(page);

}

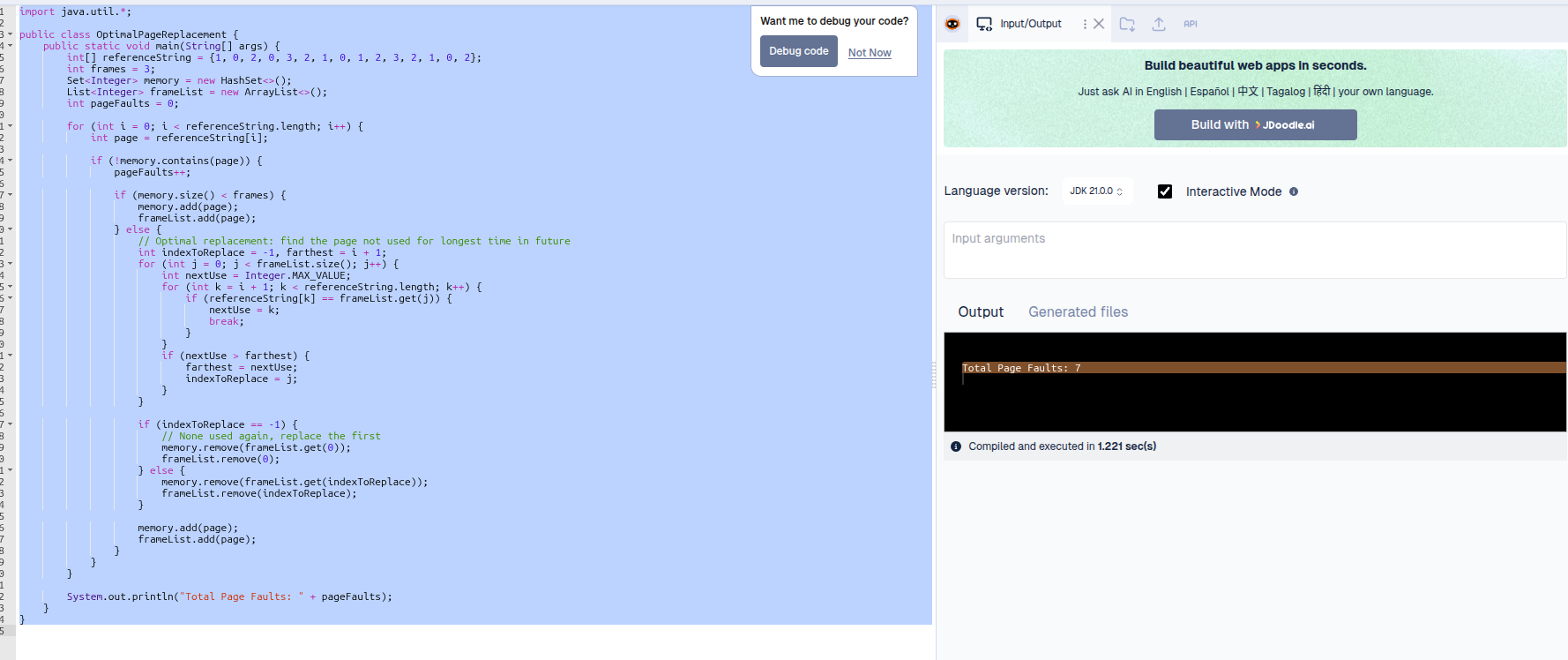
}

}

System.out.println("Total Page Faults: " + pageFaults);

}

}

**The output screenshot is :**   


That means with **3 frames** and the **Optimal Page Replacement algorithm**, the reference string:   
1, 0, 2, 0, 3, 2, 1, 0, 1, 2, 3, 2, 1, 0, 2

results in:

### **Total Page Faults: 7**