import java.util.HashSet;

import java.util.LinkedList;

import java.util.Queue;

import java.util.Scanner;

class OPTIMAL {

static int pageFaults(int n, int capacity) {

HashSet<Integer> s = new HashSet<>(capacity);

Queue<Integer> indexes = new LinkedList<>();

int page\_faults = 0;

int page\_hits = 0;

Scanner scanner = new Scanner(System.in);

int[] pages = new int[n];

System.out.println("Enter the page references:");

for (int i = 0; i < n; i++) {

pages[i] = scanner.nextInt();

}

for (int i = 0; i < n; i++) {

int page = pages[i];

if (s.contains(page)) {

page\_hits++;

} else {

if (s.size() < capacity) {

s.add(page);

page\_faults++;

indexes.add(page);

} else {

int val = indexes.poll();

s.remove(val);

s.add(page);

indexes.add(page);

page\_faults++;

}

}

}

System.out.println("Page Faults: " + page\_faults);

System.out.println("Page Hits: " + page\_hits);

return page\_faults;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the memory capacity: ");

int capacity = scanner.nextInt();

System.out.print("Enter the total number of page references: ");

int n = scanner.nextInt();

int totalFaults = pageFaults(n, capacity);

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

}

}