ASSIGNMENT NO – 7

PagingSimulation.java file

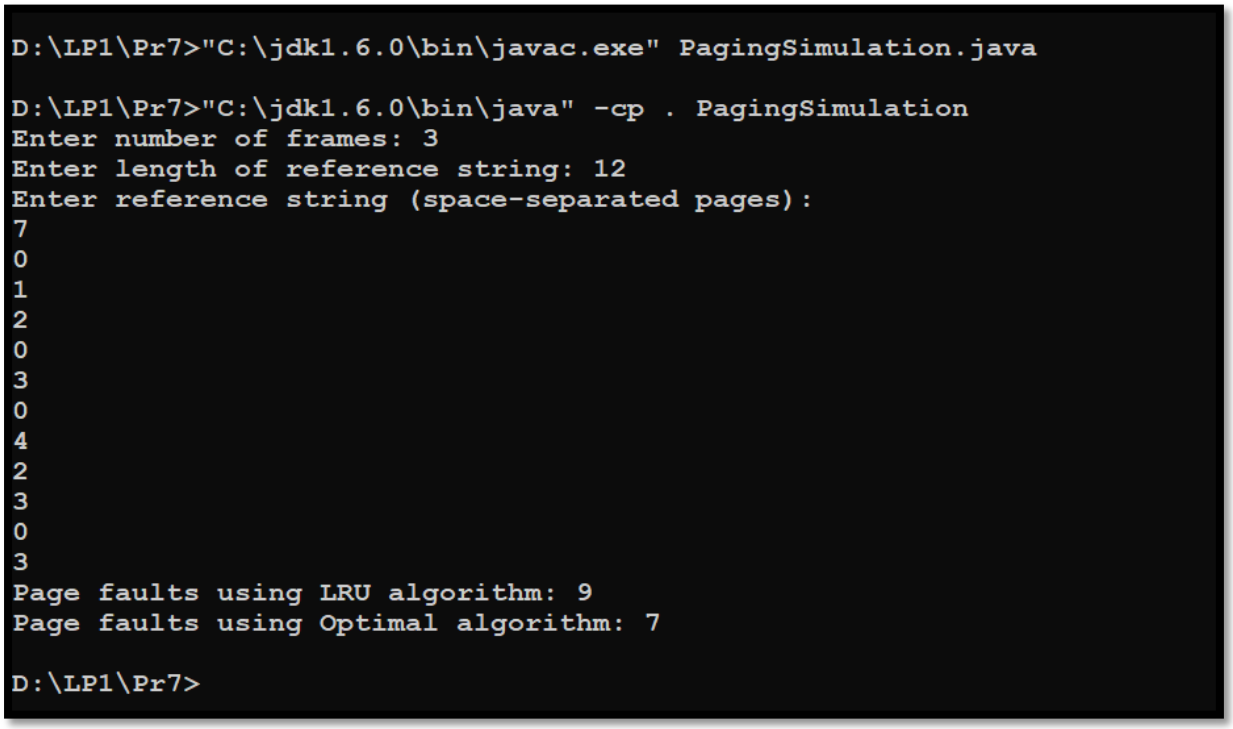
```
import java.util.*;

public class PagingSimulation {

   private int frames;

   private int[] referenceString;

   public PagingSimulation(int frames, int[] referenceString) {

      this.frames = frames;

      this.referenceString = referenceString;

   }

   public int simulateLRU() {

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

      Map<Integer, Integer> indexes = new HashMap<Integer, Integer>();

      int pageFaults = 0;

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

         int page = referenceString[i];

         if (memory.size() < frames) {

            if (!memory.contains(page)) {

               memory.add(page);

               pageFaults++;

            }

            indexes.put(page, i);

         } else {

            if (!memory.contains(page)) {

               int lruPage = Integer.MAX_VALUE;

               int minIndex = Integer.MAX_VALUE;

               for (int p : memory) {

                  if (indexes.get(p) < minIndex) {
```

```
                    minIndex = indexes.get(p);
                    lruPage = p;
                }
            }
            memory.remove(lruPage);
            memory.add(page);
            pageFaults++;
        }
        indexes.put(page, i);
    }
  }
  return pageFaults;
}
public int simulateOptimal() {
  Set<Integer> memory = new HashSet<Integer>(frames);
  int pageFaults = 0;
  for (int i = 0; i < referenceString.length; i++) {
    int page = referenceString[i];
    if (memory.size() < frames) {
      if (!memory.contains(page)) {
        memory.add(page);
        pageFaults++;
      }
    } else {
      if (!memory.contains(page)) {
        int pageToReplace = -1;
        int farthestIndex = i;
        for (int p : memory) {
          int j;
```

```
                for (j = i + 1; j < referenceString.length; j++) {
                    if (referenceString[j] == p)
                        break;
                }
                if (j == referenceString.length) {
                    pageToReplace = p;
                    break;
                }
                if (j > farthestIndex) {
                    farthestIndex = j;
                    pageToReplace = p;
                }
            }
            memory.remove(pageToReplace);
            memory.add(page);
            pageFaults++;
        }
    }
    return pageFaults;
}
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter number of frames: ");
    int frames = sc.nextInt();
    System.out.print("Enter length of reference string: ");
    int n = sc.nextInt();
    int[] referenceString = new int[n];
    System.out.println("Enter reference string (space-separated pages):");
```

```
        for (int i = 0; i < n; i++) {
            referenceString[i] = sc.nextInt();
        }
        PagingSimulation simulation = new PagingSimulation(frames, referenceString);
        int lruFaults = simulation.simulateLRU();
        int optimalFaults = simulation.simulateOptimal();
        System.out.println("Page faults using LRU algorithm: " + lruFaults);
        System.out.println("Page faults using Optimal algorithm: " + optimalFaults);
        sc.close();
    }
}
```

OUTPUT:



```
D:\LP1\Pr7>"C:\jdk1.6.0\bin\javac.exe" PagingSimulation.java

D:\LP1\Pr7>"C:\jdk1.6.0\bin\java" -cp . PagingSimulation
Enter number of frames: 3
Enter length of reference string: 12
Enter reference string (space-separated pages):
7
0
1
2
0
3
0
4
2
3
0
3
Page faults using LRU algorithm: 9
Page faults using Optimal algorithm: 7

D:\LP1\Pr7>
```