Ain Shams University / Faculty of Engineering

CSE Department

Operating Systems CSE223

Main Memory Algorithms

Submitted By

Omar Magdy Shaaban 16P6034

Seif El-Din Mohamed 16P8109

Group 2 Section 1

Submitted To

Dr. Gamal Abdel Shafy

Cairo, 2018

Abstract

This project implements some of the algorithms used for page replacement including FIFO, LRU, Optimal, LFU, Second Chance, and Enhanced Second Chance. It is implemented in Java using NetBeans IDE, subdivided into several functions, each of them expresses an algorithm.

Table of Contents

1. Implementation

```
package mainmemoryalgorithmsv12;


import java.util.Random;

import java.util.Scanner;


public class MainMemoryAlgorithmsv12 {


    private static final Scanner SC = new Scanner(System.in);

    private static final Random R = new Random();

    private static int[] pageNumbers;

    private static int pageFaults;

    private static int[][] pageFrames;

    private static int[] frameReferences;

    private static int SecondChancePointer = 0;

    private static int[][] frameReferenceModifyBits;

    private static int[] pageModifyBits;

    private static int ESCPointer = 0;


    public static void main(String[] args) {

        init();

        System.out.println("First In First Out:");

        FirstInFirstOut();

        printPageFaultsAndReset();
        System.out.println("Least Recently Used:");

        LeastRecentlyUsed();

        printPageFaultsAndReset();

        System.out.println("Optimal:");

        Optimal();

        printPageFaultsAndReset();

        System.out.println("Least Frequently Used:");

        LeastFrequentlyUsed();

        printPageFaultsAndReset();

        System.out.println("Second Chance:");

        SecondChance();

        printPageFaultsAndReset();

        SecondChanceReset();

        System.out.println("Enhanced Second Chance:");

        EnhancedSecondChance();

        printPageFaultsAndReset();

        EnhancedSecondChanceReset();

    }


    public static void init() {

        System.out.println("Enter the length of the page-reference string:");

        int pageReferenceLength = SC.nextInt();

        pageNumbers = new int[pageReferenceLength];

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

            pageNumbers[i] = R.nextInt(100);
        }


        System.out.print("Your page-reference string is ");
        for (int i = 0; i < pageNumbers.length - 1; i++) {
            System.out.print(pageNumbers[i] + ",");
        }
        System.out.println(pageNumbers[pageReferenceLength - 1]);


        int numberOfPageFrames = 1 + R.nextInt(20);
        //4 for Enhanced Second Chance,3 for the rest
        System.out.println("Number of available frames are: " + numberOfPageFrames);
        pageFrames = new int[numberOfPageFrames][2];
        for (int i = 0; i < pageFrames.length; i++) {
            pageFrames[i][0] = -1;
        }
    }


    public static void printPageFaultsAndReset() {
        System.out.println("Page faults are: " + pageFaults);
        pageFaults = 0;
        for (int i = 0; i < pageFrames.length; i++) {
            pageFrames[i][0] = -1;
            pageFrames[i][1] = 0;
        }
    }
```

```
public static void SecondChanceReset() {
    SecondChancePointer = 0;
    for (int i = 0; i < frameReferences.length; i++) {
        frameReferences[i] = 0;
    }
}

public static void EnhancedSecondChanceReset() {
    for (int i = 0; i < pageFrames.length; i++) {
        for (int j = 0; j < 2; j++) {
            frameReferenceModifyBits[i][j] = 0;
        }
    }
    for (int i = 0; i < pageNumbers.length; i++){
        pageModifyBits[i] = 0;
    }
    ESCPointer = 0;
}

public static void printPageFramesContents() {
    for (int i = 0; i < pageFrames.length - 1; i++) {
        System.out.print(pageFrames[i][0] + ",");
    }
    System.out.println(pageFrames[pageFrames.length - 1][0]);
```

```
}


public static int assignPageToFrameLRU(int pageNumber, int index) {
    for (int i = 0; i < pageFrames.length; i++) {
        if (pageFrames[i][0] == pageNumber) {
            pageFrames[i][1] = index;
            return i;
        }
    }


    for (int i = 0; i < pageFrames.length; i++) {
        if (pageFrames[i][0] == -1) {
            pageFrames[i][0] = pageNumber;
            pageFrames[i][1] = index;
            pageFaults++;
            return -2;
        }
    }
    return -1;
}


public static void LeastRecentlyUsedReplace(int pageNumber, int index) {
    int minPageIndex = 200;
    int toBeReplaced = 0;
    for (int i = 0; i < pageFrames.length; i++) {
        if (pageFrames[i][1] < minPageIndex) {
            minPageIndex = pageFrames[i][1];
            toBeReplaced = i;
        }
    }
    pageFrames[toBeReplaced][0] = pageNumber;
    pageFrames[toBeReplaced][1] = index;
}


public static void LeastRecentlyUsed() {
    int FreeFrame = -2;
    for (int i = 0; i < pageNumbers.length; i++) {
        FreeFrame = assignPageToFrameLRU(pageNumbers[i], i);
        switch (FreeFrame) {
            case -2:
                System.out.print("Current frames allocation: ");
                printPageFramesContents();
                break;
            case -1:
                LeastRecentlyUsedReplace(pageNumbers[i], i);
                pageFaults++;
                System.out.print("New frames allocation: ");
                printPageFramesContents();
                break;
            default:
```

```
            System.out.print("Change only in index " + FreeFrame + " with new"
                    + " index " + i + " : ");
            printPageFramesContents();
            break;
        }
    }
}

public static void FirstInFirstOut() {
    int FreeFrame = -2;
    for (int i = 0; i < pageNumbers.length; i++) {
        FreeFrame = assignPageToFrameFIFO(pageNumbers[i], i);
        switch (FreeFrame) {
            case -2:
                System.out.print("Current frames allocation: ");
                printPageFramesContents();
                break;
            case -1:
                FirstInFirstOutReplace(pageNumbers[i], i);
                pageFaults++;
                System.out.print("New frames allocation: ");
                printPageFramesContents();
                break;
            default:
                System.out.print("Index " + FreeFrame + " remains the same: ");
            printPageFramesContents();
            break;
        }
    }
}

public static void FirstInFirstOutReplace(int pageNumber, int index) {
    int minPageIndex = 200;
    int toBeReplaced = 0;
    for (int i = 0; i < pageFrames.length; i++) {
        if (pageFrames[i][1] < minPageIndex) {
            minPageIndex = pageFrames[i][1];
            toBeReplaced = i;
        }
    }
    pageFrames[toBeReplaced][0] = pageNumber;
    pageFrames[toBeReplaced][1] = index;
}

public static int assignPageToFrameFIFO(int pageNumber, int index) {
    for (int i = 0; i < pageFrames.length; i++) {
        if (pageFrames[i][0] == pageNumber) {
            return i;
        }
    }
    for (int i = 0; i < pageFrames.length; i++) {
        if (pageFrames[i][0] == -1) {
            pageFrames[i][0] = pageNumber;
            pageFrames[i][1] = index;
            pageFaults++;
            return -2;
        }
    }
    return -1;
}

public static void Optimal() {
    int FreeFrame = -2;
    for (int i = 0; i < pageNumbers.length; i++) {
        FreeFrame = assignPageToFrameOptimal(pageNumbers[i], i);
        switch (FreeFrame) {
            case -2:
                System.out.print("Current frames allocation: ");
                printPageFramesContents();
                break;
            case -1:
                OptimalReplace(pageNumbers[i], i);
                pageFaults++;
                System.out.print("New frames allocation: ");
                printPageFramesContents();
                break;
            default:
                System.out.print("Change only in index " + FreeFrame + " with new"
                        + " index " + i + " : ");
                printPageFramesContents();
                break;
        }
    }
}

public static void OptimalReplace(int pageNumber, int index) {
    int[] count = new int[pageFrames.length];
    for (int i = 0; i < pageFrames.length; i++) {
        for (int j = index + 1; j < pageNumbers.length; j++) {
            if (pageFrames[i][0] == pageNumbers[j]) {
                break;
            } else {
                count[i]++;
            }
        }
    }
    int maxPageCount = 0;
    int toBeReplaced = 0;
    for (int i = 0; i < count.length; i++) {
        if (count[i] > maxPageCount) {

            maxPageCount = count[i];

            toBeReplaced = i;


        }

    }

    pageFrames[toBeReplaced][0] = pageNumber;

    pageFrames[toBeReplaced][1] = index;

}


public static int assignPageToFrameOptimal(int pageNumber, int index) {

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

        if (pageFrames[i][0] == pageNumber) {

            pageFrames[i][1] = index;

            return i;

        }

    }


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

        if (pageFrames[i][0] == -1) {

            pageFrames[i][0] = pageNumber;

            pageFrames[i][1] = index;

            pageFaults++;

            return -2;

        }
    }

    return -1;

}


public static void LeastFrequentlyUsed() {

    int FreeFrame = -2;

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

        FreeFrame = assignPageToFrameLFU(pageNumbers[i], i);

        switch (FreeFrame) {

            case -2:

                System.out.print("Current frames allocation: ");

                printPageFramesContents();

                break;

            case -1:

                LeastFrequentlyUsedReplace(pageNumbers[i], i);

                pageFaults++;

                System.out.print("New frames allocation: ");

                printPageFramesContents();

                break;

            default:

                System.out.print("Index " + FreeFrame + " remains the same: ");

                printPageFramesContents();

                break;

        }

    }
```

```
}


public static void LeastFrequentlyUsedReplace(int pageNumber, int index) {
   int[] reference = new int[pageFrames.length];
   for (int i = 0; i < pageFrames.length; i++) {
      for (int j = index - 1; j > 0; j--) {
         if (pageFrames[i][0] == pageNumbers[j]) {
            reference[i]++;
         }
      }
   }
   int minPageReference = 200;
   int toBeReplaced = 0;
   for (int i = 0; i < reference.length; i++) {
      if (reference[i] < minPageReference) {
         minPageReference = reference[i];
         toBeReplaced = i;


      }
   }
   pageFrames[toBeReplaced][0] = pageNumber;
   pageFrames[toBeReplaced][1] = index;
}


public static int assignPageToFrameLFU(int pageNumber, int index) {
    for (int i = 0; i < pageFrames.length; i++) {
        if (pageFrames[i][0] == pageNumber) {
            return i;
        }
    }

    for (int i = 0; i < pageFrames.length; i++) {
        if (pageFrames[i][0] == -1) {
            pageFrames[i][0] = pageNumber;
            pageFrames[i][1] = index;
            pageFaults++;
            return -2;
        }
    }
    return -1;
}

public static void SecondChance() {
    int FreeFrame = -2;
    frameReferences = new int[pageFrames.length];
    for (int i = 0; i < pageNumbers.length; i++) {
        FreeFrame = assignPageToFrameSC(pageNumbers[i], i);
        switch (FreeFrame) {
            case -2:
                System.out.print("Current frames allocation: ");
                printPageFramesContents();

                break;

            case -1:

                SecondChanceReplace(pageNumbers[i], i);

                pageFaults++;

                System.out.print("New frames allocation: ");

                printPageFramesContents();

                break;

            default:

                System.out.print("Index " + FreeFrame + " remains the same: ");

                printPageFramesContents();

                break;

        }

    }

}


public static int assignPageToFrameSC(int pageNumber, int index) {

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

        if (pageFrames[i][0] == pageNumber) {

            frameReferences[i] = 1;

            return i;

        }

    }


    for (int i = 0; i < pageFrames.length; i++) {
        if (pageFrames[i][0] == -1) {
            pageFrames[i][0] = pageNumber;
            pageFrames[i][1] = index;
            frameReferences[i] = 1;
            pageFaults++;
            return -2;
        }
    }
    return -1;
}

public static void SecondChanceReplace(int pageNumber, int index) {
    for (; SecondChancePointer < pageFrames.length;) {
        if (frameReferences[SecondChancePointer] == 0) {
            pageFrames[SecondChancePointer][0] = pageNumber;
            pageFrames[SecondChancePointer][1] = index;
            frameReferences[SecondChancePointer] = 1;
            SecondChancePointer++;
            SecondChancePointer %= pageFrames.length;
            return;
        } else {
            frameReferences[SecondChancePointer] = 0;
            SecondChancePointer++;
            SecondChancePointer %= pageFrames.length;
        }
    }
}

public static void EnhancedSecondChance() {
    int FreeFrame = -2;
    frameReferenceModifyBits = new int[pageFrames.length][2];
    pageModifyBits = new int[pageNumbers.length];
    fillPageModifyBits();
    printPageModifyBits();
    for (int i = 0; i < pageNumbers.length; i++) {
        FreeFrame = assignPageToFrameESC(pageNumbers[i], i);
        switch (FreeFrame) {
            case -2:
                System.out.print("Current frames allocation: ");
                printPageFramesContents();
                break;
            case -1:
                EnhancedSecondChanceReplace(pageNumbers[i], i);
                pageFaults++;
                System.out.print("New frames allocation: ");
                printPageFramesContents();
                break;
            default:
                System.out.print("Index " + FreeFrame + " remains the same: ");
                printPageFramesContents();
                break;
            }
        }
    }

    public static void fillPageModifyBits() {
        for(int i = 0; i < pageNumbers.length; i++){
            pageModifyBits[i] = R.nextInt(2);
        }
    }

    public static void printPageModifyBits(){
        System.out.print("Pages modify bits are: ");
        for(int i = 0; i < pageNumbers.length - 1; i++){
            System.out.print(pageModifyBits[i] + ",");
        }
        System.out.println(pageModifyBits[pageNumbers.length - 1]);
    }

    public static int assignPageToFrameESC(int pageNumber, int index) {
        for (int i = 0; i < pageFrames.length; i++) {
            if (pageFrames[i][0] == pageNumber) {
//                frameReferenceModifyBits[i][0] = 1;
                return i;
            }
    }

    for (int i = 0; i < pageFrames.length; i++) {
        if (pageFrames[i][0] == -1) {
            pageFrames[i][0] = pageNumber;
            pageFrames[i][1] = index;
            frameReferenceModifyBits[i][0] = 1;
            frameReferenceModifyBits[i][1] = pageModifyBits[index];
            pageFaults++;
            return -2;
        }
    }
    return -1;
}

public static void EnhancedSecondChanceReplace(int pageNumber, int index) {
    int[] ReferenceModifyToInteger = new int[pageFrames.length];
    int reference;
    int modify;
    int ZeroIndex = -1;
    boolean isReplaced = false;
    for (int i = 0; i < pageFrames.length; i++) {
        reference = frameReferenceModifyBits[i][0];
        modify = frameReferenceModifyBits[i][1];
        ReferenceModifyToInteger[i] = 1 * modify + 2 * reference;
```

```
}

while (ZeroIndex == -1 && !isReplaced) {

  isReplaced = false;

  ZeroIndex = checkZeroClass(ReferenceModifyToInteger);

  if (ZeroIndex != -1) {

    pageFrames[ZeroIndex][0] = pageNumber;

    pageFrames[ZeroIndex][1] = index;

    frameReferenceModifyBits[ZeroIndex][0] = 1;

    frameReferenceModifyBits[ZeroIndex][1] = pageModifyBits[index];

    ReferenceModifyToInteger[ZeroIndex] = 2 + pageModifyBits[index];

    ESCPointer = ZeroIndex + 1;

    ESCPointer %= pageFrames.length;

  } else {

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

      if (ReferenceModifyToInteger[ESCPointer] == 1 && !isReplaced) {

        pageFrames[ESCPointer][0] = pageNumber;

        pageFrames[ESCPointer][1] = index;

        frameReferenceModifyBits[ESCPointer][0] = 1;

        frameReferenceModifyBits[ESCPointer][1] = pageModifyBits[index];

        ReferenceModifyToInteger[ESCPointer] = 2 + pageModifyBits[index];

        ESCPointer++;

        ESCPointer %= pageFrames.length;

        //ZeroIndex = -2;

        isReplaced = true;

        //break;
```

```
            } else {
                if(ReferenceModifyToInteger[ESCPointer] > 1){
                frameReferenceModifyBits[ESCPointer][0] = 0;
                ReferenceModifyToInteger[ESCPointer] -= 2;
                }
                ESCPointer++;
                ESCPointer %= pageFrames.length;
            }
        }
    }
}
for (int i = 0; i < pageFrames.length; i++) {
    switch (ReferenceModifyToInteger[i]) {
        case 3:
            frameReferenceModifyBits[i][0] = 1;
            frameReferenceModifyBits[i][1] = 1;
            break;
        case 2:
            frameReferenceModifyBits[i][0] = 1;
            frameReferenceModifyBits[i][1] = 0;
            break;
        case 1:
            frameReferenceModifyBits[i][0] = 0;
            frameReferenceModifyBits[i][1] = 1;
            break;
            case 0:
                frameReferenceModifyBits[i][0] = 0;
                frameReferenceModifyBits[i][1] = 0;
                break;
        }
    }
  }

  public static int checkZeroClass(int ReferenceModifyToInteger[]) {
    for (int i = 0; i < pageFrames.length; i++) {
        if (ReferenceModifyToInteger[i] == 0) {
            return i;
        }
    }
    return -1;
  }
}
```

2. Test Cases

1) 15, Output:

```
Enter the length of the page-reference string:
15
Your page-reference string is 62,81,12,28,80,58,67,46,76,9,36,71,98,15,73
Number of available frames are: 4
First In First Out:
Current frames allocation: 62,-1,-1,-1
Current frames allocation: 62,81,-1,-1

Current frames allocation: 62,81,12,-1

Current frames allocation: 62,81,12,28

New frames allocation: 80,81,12,28

New frames allocation: 80,58,12,28

New frames allocation: 80,58,67,28

New frames allocation: 80,58,67,46

New frames allocation: 76,58,67,46

New frames allocation: 76,9,67,46

New frames allocation: 76,9,36,46

New frames allocation: 76,9,36,71

New frames allocation: 98,9,36,71

New frames allocation: 98,15,36,71

New frames allocation: 98,15,73,71

Page faults are: 15

Least Recently Used:

Current frames allocation: 62,-1,-1,-1

Current frames allocation: 62,81,-1,-1

Current frames allocation: 62,81,12,-1

Current frames allocation: 62,81,12,28

New frames allocation: 80,81,12,28

New frames allocation: 80,58,12,28

New frames allocation: 80,58,67,28

New frames allocation: 80,58,67,46

New frames allocation: 76,58,67,46

New frames allocation: 76,9,67,46

New frames allocation: 76,9,36,46

New frames allocation: 76,9,36,71

New frames allocation: 98,9,36,71
New frames allocation: 98,15,36,71

New frames allocation: 98,15,73,71

Page faults are: 15

Optimal:

Current frames allocation: 62,-1,-1,-1

Current frames allocation: 62,81,-1,-1

Current frames allocation: 62,81,12,-1

Current frames allocation: 62,81,12,28

New frames allocation: 80,81,12,28

New frames allocation: 58,81,12,28

New frames allocation: 67,81,12,28

New frames allocation: 46,81,12,28

New frames allocation: 76,81,12,28

New frames allocation: 9,81,12,28

New frames allocation: 36,81,12,28

New frames allocation: 71,81,12,28

New frames allocation: 98,81,12,28

New frames allocation: 15,81,12,28

New frames allocation: 73,81,12,28

Page faults are: 15

Least Frequently Used:

Current frames allocation: 62,-1,-1,-1

Current frames allocation: 62,81,-1,-1

Current frames allocation: 62,81,12,-1

Current frames allocation: 62,81,12,28

New frames allocation: 80,81,12,28

New frames allocation: 58,81,12,28

New frames allocation: 67,81,12,28

New frames allocation: 46,81,12,28
New frames allocation: 76,81,12,28

New frames allocation: 9,81,12,28

New frames allocation: 36,81,12,28

New frames allocation: 71,81,12,28

New frames allocation: 98,81,12,28

New frames allocation: 15,81,12,28

New frames allocation: 73,81,12,28

Page faults are: 15

Second Chance:

Current frames allocation: 62,-1,-1,-1

Current frames allocation: 62,81,-1,-1

Current frames allocation: 62,81,12,-1

Current frames allocation: 62,81,12,28

New frames allocation: 80,81,12,28

New frames allocation: 80,58,12,28

New frames allocation: 80,58,67,28

New frames allocation: 80,58,67,46

New frames allocation: 76,58,67,46

New frames allocation: 76,9,67,46

New frames allocation: 76,9,36,46

New frames allocation: 76,9,36,71

New frames allocation: 98,9,36,71

New frames allocation: 98,15,36,71

New frames allocation: 98,15,73,71

Page faults are: 15

Enhanced Second Chance:

Pages modify bits are: 0,1,0,1,0,1,1,1,1,0,1,0,0,1,0

Current frames allocation: 62,-1,-1,-1

Current frames allocation: 62,81,-1,-1
Current frames allocation: 62,81,12,-1

Current frames allocation: 62,81,12,28

New frames allocation: 80,81,12,28

New frames allocation: 80,81,58,28

New frames allocation: 80,81,58,67

New frames allocation: 46,81,58,67

New frames allocation: 46,76,58,67

New frames allocation: 46,76,9,67

New frames allocation: 46,36,9,67

New frames allocation: 46,36,71,67

New frames allocation: 46,36,71,98

New frames allocation: 46,36,15,98

New frames allocation: 73,36,15,98

Page faults are: 15
```

2) 17, Output:

```
Enter the length of the page-reference string:

17

Your page-reference string is 82,56,43,91,91,21,20,17,85,58,44,66,40,90,26,92,53

Number of available frames are: 12

First In First Out:

Current frames allocation: 82,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,-1,-1,-1,-1,-1,-1,-1,-1

Index 3 remains the same: 82,56,43,91,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,-1,-1,-1,-1
Current frames allocation: 82,56,43,91,21,20,17,85,58,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,44,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,44,66,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,44,66,40

New frames allocation: 90,56,43,91,21,20,17,85,58,44,66,40

New frames allocation: 90,26,43,91,21,20,17,85,58,44,66,40

New frames allocation: 90,26,92,91,21,20,17,85,58,44,66,40

New frames allocation: 90,26,92,53,21,20,17,85,58,44,66,40

Page faults are: 16

Least Recently Used:

Current frames allocation: 82,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,-1,-1,-1,-1,-1,-1,-1,-1

Change only in index 3 with new index 4 : 82,56,43,91,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,44,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,44,66,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,44,66,40

New frames allocation: 90,56,43,91,21,20,17,85,58,44,66,40

New frames allocation: 90,26,43,91,21,20,17,85,58,44,66,40

New frames allocation: 90,26,92,91,21,20,17,85,58,44,66,40

New frames allocation: 90,26,92,53,21,20,17,85,58,44,66,40

Page faults are: 16

Optimal:
```

```
Current frames allocation: 82,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,-1,-1,-1,-1,-1,-1,-1,-1

Change only in index 3 with new index 4 : 82,56,43,91,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,44,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,44,66,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,44,66,40

New frames allocation: 90,56,43,91,21,20,17,85,58,44,66,40

New frames allocation: 26,56,43,91,21,20,17,85,58,44,66,40

New frames allocation: 92,56,43,91,21,20,17,85,58,44,66,40

New frames allocation: 53,56,43,91,21,20,17,85,58,44,66,40

Page faults are: 16

Least Frequently Used:

Current frames allocation: 82,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,-1,-1,-1,-1,-1,-1,-1,-1

Index 3 remains the same: 82,56,43,91,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,-1,-1,-1
Current frames allocation: 82,56,43,91,21,20,17,85,58,44,-1,-1
Current frames allocation: 82,56,43,91,21,20,17,85,58,44,66,-1
Current frames allocation: 82,56,43,91,21,20,17,85,58,44,66,40
New frames allocation: 90,56,43,91,21,20,17,85,58,44,66,40
New frames allocation: 26,56,43,91,21,20,17,85,58,44,66,40
New frames allocation: 92,56,43,91,21,20,17,85,58,44,66,40
New frames allocation: 53,56,43,91,21,20,17,85,58,44,66,40
Page faults are: 16
Second Chance:
Current frames allocation: 82,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
Current frames allocation: 82,56,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
Current frames allocation: 82,56,43,-1,-1,-1,-1,-1,-1,-1,-1,-1
Current frames allocation: 82,56,43,91,-1,-1,-1,-1,-1,-1,-1,-1
Index 3 remains the same: 82,56,43,91,-1,-1,-1,-1,-1,-1,-1,-1
Current frames allocation: 82,56,43,91,21,-1,-1,-1,-1,-1,-1,-1
Current frames allocation: 82,56,43,91,21,20,-1,-1,-1,-1,-1,-1
Current frames allocation: 82,56,43,91,21,20,17,-1,-1,-1,-1,-1
Current frames allocation: 82,56,43,91,21,20,17,85,-1,-1,-1,-1
Current frames allocation: 82,56,43,91,21,20,17,85,58,-1,-1,-1
Current frames allocation: 82,56,43,91,21,20,17,85,58,44,-1,-1
Current frames allocation: 82,56,43,91,21,20,17,85,58,44,66,-1
Current frames allocation: 82,56,43,91,21,20,17,85,58,44,66,40
New frames allocation: 90,56,43,91,21,20,17,85,58,44,66,40
New frames allocation: 90,26,43,91,21,20,17,85,58,44,66,40
New frames allocation: 90,26,92,91,21,20,17,85,58,44,66,40
New frames allocation: 90,26,92,53,21,20,17,85,58,44,66,40
Page faults are: 16
Enhanced Second Chance:
Pages modify bits are: 0,0,1,1,0,1,0,0,0,0,0,1,0,0,0,1,1
```

```
Current frames allocation: 82,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,-1,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,-1,-1,-1,-1,-1,-1,-1,-1

Index 3 remains the same: 82,56,43,91,-1,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,-1,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,-1,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,-1,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,44,-1,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,44,66,-1

Current frames allocation: 82,56,43,91,21,20,17,85,58,44,66,40

New frames allocation: 90,56,43,91,21,20,17,85,58,44,66,40

New frames allocation: 90,26,43,91,21,20,17,85,58,44,66,40

New frames allocation: 90,26,43,91,21,92,17,85,58,44,66,40

New frames allocation: 90,26,43,91,21,92,53,85,58,44,66,40

Page faults are: 16
```

3) 20, Output:

```
Enter the length of the page-reference string:

20

Your page-reference string is 82,42,35,40,40,40,45,88,7,50,88,58,64,58,88,71,2,6,79,65

Number of available frames are: 8

First In First Out:

Current frames allocation: 82,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,42,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,42,35,-1,-1,-1,-1,-1

Current frames allocation: 82,42,35,40,-1,-1,-1,-1

Index 3 remains the same: 82,42,35,40,-1,-1,-1,-1
Index 3 remains the same: 82,42,35,40,-1,-1,-1,-1

Current frames allocation: 82,42,35,40,45,-1,-1,-1

Current frames allocation: 82,42,35,40,45,88,-1,-1

Current frames allocation: 82,42,35,40,45,88,7,-1

Current frames allocation: 82,42,35,40,45,88,7,50

Index 5 remains the same: 82,42,35,40,45,88,7,50

New frames allocation: 58,42,35,40,45,88,7,50

New frames allocation: 58,64,35,40,45,88,7,50

Index 0 remains the same: 58,64,35,40,45,88,7,50

Index 5 remains the same: 58,64,35,40,45,88,7,50

New frames allocation: 58,64,71,40,45,88,7,50

New frames allocation: 58,64,71,2,45,88,7,50

New frames allocation: 58,64,71,2,6,88,7,50

New frames allocation: 58,64,71,2,6,79,7,50

New frames allocation: 58,64,71,2,6,79,65,50

Page faults are: 15

Least Recently Used:

Current frames allocation: 82,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,42,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,42,35,-1,-1,-1,-1,-1

Current frames allocation: 82,42,35,40,-1,-1,-1,-1

Change only in index 3 with new index 4 : 82,42,35,40,-1,-1,-1,-1

Change only in index 3 with new index 5 : 82,42,35,40,-1,-1,-1,-1

Current frames allocation: 82,42,35,40,45,-1,-1,-1

Current frames allocation: 82,42,35,40,45,88,-1,-1

Current frames allocation: 82,42,35,40,45,88,7,-1

Current frames allocation: 82,42,35,40,45,88,7,50

Change only in index 5 with new index 10 : 82,42,35,40,45,88,7,50

New frames allocation: 58,42,35,40,45,88,7,50
New frames allocation: 58,64,35,40,45,88,7,50

Change only in index 0 with new index 13 : 58,64,35,40,45,88,7,50

Change only in index 5 with new index 14 : 58,64,35,40,45,88,7,50

New frames allocation: 58,64,71,40,45,88,7,50

New frames allocation: 58,64,71,2,45,88,7,50

New frames allocation: 58,64,71,2,6,88,7,50

New frames allocation: 58,64,71,2,6,88,79,50

New frames allocation: 58,64,71,2,6,88,79,65

Page faults are: 15

Optimal:

Current frames allocation: 82,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,42,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,42,35,-1,-1,-1,-1,-1

Current frames allocation: 82,42,35,40,-1,-1,-1,-1

Change only in index 3 with new index 4 : 82,42,35,40,-1,-1,-1,-1

Change only in index 3 with new index 5 : 82,42,35,40,-1,-1,-1,-1

Current frames allocation: 82,42,35,40,45,-1,-1,-1

Current frames allocation: 82,42,35,40,45,88,-1,-1

Current frames allocation: 82,42,35,40,45,88,7,-1

Current frames allocation: 82,42,35,40,45,88,7,50

Change only in index 5 with new index 10 : 82,42,35,40,45,88,7,50

New frames allocation: 58,42,35,40,45,88,7,50

New frames allocation: 58,64,35,40,45,88,7,50

Change only in index 0 with new index 13 : 58,64,35,40,45,88,7,50

Change only in index 5 with new index 14 : 58,64,35,40,45,88,7,50

New frames allocation: 71,64,35,40,45,88,7,50

New frames allocation: 2,64,35,40,45,88,7,50

New frames allocation: 6,64,35,40,45,88,7,50

New frames allocation: 79,64,35,40,45,88,7,50
New frames allocation: 65,64,35,40,45,88,7,50

Page faults are: 15

Least Frequently Used:

Current frames allocation: 82,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,42,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,42,35,-1,-1,-1,-1,-1

Current frames allocation: 82,42,35,40,-1,-1,-1,-1

Index 3 remains the same: 82,42,35,40,-1,-1,-1,-1

Index 3 remains the same: 82,42,35,40,-1,-1,-1,-1

Current frames allocation: 82,42,35,40,45,-1,-1,-1

Current frames allocation: 82,42,35,40,45,88,-1,-1

Current frames allocation: 82,42,35,40,45,88,7,-1

Current frames allocation: 82,42,35,40,45,88,7,50

Index 5 remains the same: 82,42,35,40,45,88,7,50

New frames allocation: 58,42,35,40,45,88,7,50

New frames allocation: 64,42,35,40,45,88,7,50

New frames allocation: 58,42,35,40,45,88,7,50

Index 5 remains the same: 58,42,35,40,45,88,7,50

New frames allocation: 58,71,35,40,45,88,7,50

New frames allocation: 58,2,35,40,45,88,7,50

New frames allocation: 58,6,35,40,45,88,7,50

New frames allocation: 58,79,35,40,45,88,7,50

New frames allocation: 58,65,35,40,45,88,7,50

Page faults are: 16

Second Chance:

Current frames allocation: 82,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,42,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,42,35,-1,-1,-1,-1,-1

Current frames allocation: 82,42,35,40,-1,-1,-1,-1
```

```
Index 3 remains the same: 82,42,35,40,-1,-1,-1,-1

Index 3 remains the same: 82,42,35,40,-1,-1,-1,-1

Current frames allocation: 82,42,35,40,45,-1,-1,-1

Current frames allocation: 82,42,35,40,45,88,-1,-1

Current frames allocation: 82,42,35,40,45,88,7,-1

Current frames allocation: 82,42,35,40,45,88,7,50

Index 5 remains the same: 82,42,35,40,45,88,7,50

New frames allocation: 58,42,35,40,45,88,7,50

New frames allocation: 58,64,35,40,45,88,7,50

Index 0 remains the same: 58,64,35,40,45,88,7,50

Index 5 remains the same: 58,64,35,40,45,88,7,50

New frames allocation: 58,64,71,40,45,88,7,50

New frames allocation: 58,64,71,2,45,88,7,50

New frames allocation: 58,64,71,2,6,88,7,50

New frames allocation: 58,64,71,2,6,88,79,50

New frames allocation: 58,64,71,2,6,88,79,65

Page faults are: 15

Enhanced Second Chance:

Pages modify bits are: 1,1,1,1,1,1,1,0,0,1,1,0,1,0,0,0,1,1,0,1

Current frames allocation: 82,-1,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,42,-1,-1,-1,-1,-1,-1

Current frames allocation: 82,42,35,-1,-1,-1,-1,-1

Current frames allocation: 82,42,35,40,-1,-1,-1,-1

Index 3 remains the same: 82,42,35,40,-1,-1,-1,-1

Index 3 remains the same: 82,42,35,40,-1,-1,-1,-1

Current frames allocation: 82,42,35,40,45,-1,-1,-1

Current frames allocation: 82,42,35,40,45,88,-1,-1

Current frames allocation: 82,42,35,40,45,88,7,-1

Current frames allocation: 82,42,35,40,45,88,7,50
```

```
Index 5 remains the same: 82,42,35,40,45,88,7,50

New frames allocation: 82,42,35,40,45,58,7,50

New frames allocation: 82,42,35,40,45,58,64,50

Index 5 remains the same: 82,42,35,40,45,58,64,50

New frames allocation: 82,42,35,40,45,58,64,88

New frames allocation: 82,42,35,40,45,71,64,88

New frames allocation: 82,42,35,40,45,71,2,88

New frames allocation: 82,42,35,40,45,6,2,88

New frames allocation: 82,42,35,40,45,6,2,79

New frames allocation: 65,42,35,40,45,6,2,79

Page faults are: 16
```

Note: Choosing random values don't reflect really much differences between the algorithms. Another version of the code is available upon request, in which you can enter specific values for the reference string and number of frames. Lecture examples were used to prove correctness of the implementation and results are available if asked for it.