CS266 LAB 11

NAME:

ARCHIT AGRAWAL

ROLL NO. :

202051213

SECTION:

2

Question

Problem

1. Generate a random 20 (at least) page reference string from 0 to 9.
2. For a fixed sequence, compare the performance of all the algorithms given below.
3. Repeat whole experiment minimum five times to check performance of all the algorithms.

Approach 1

Random Page Replacement (RPR) is an easy to implement low overhead page replacement strategy. Each page in main memory has an equal likelihood of being selected for replacement. The underlying belief is there are many page frames from which to choose, there is only a small probability of replacing a page likely to be referenced again almost immediately. Consider minimum 5 page frames available and empty initially.

Approach 2

Let's consider another approach where we give priority or weights to frame location. In case of 5 frames, frame 1 and 5 has weight 10, frame 2 and 4 has weight 20 and frame 3 has weight 40. Generate a random number from Uniform Distribution (0,1) to decide the frame location to replace a page. (This is irrespective of the page sequence.)

Code

```c
#include<stdio.h>
#include<stdlib.h>
#include<time.h>


void main(){

    int n = 5; //number of page frames, fixed

    int s; //length of referenceString
    printf("Enter the length of reference string (>= 20) : ");
    scanf("%d", &s);
    int referenceString[s];

    srand(time(0));
    printf("Reference String : ");
    for(int i = 0; i < s; i++){
        referenceString[i] = rand() % 10;
        printf("%d", referenceString[i]);
        if(i != s-1) printf(", ");
    }

    printf("\n");

    //same referenceString is passed to both the approaches
```

```
    printf("Page Faults in Equally Likely RPR (Approach 1) = %d\n",
equallyLikelyRPR(s, referenceString, n));
    printf("Page Faults in Weighted RPR (Approach 2) = %d\n", weightedRPR(s,
referenceString, n));

}

//Approach 1
int equallyLikelyRPR(int s, int referenceString[s], int n){
    int frames[n];
    int vacantFrames = 0; //stores the number of frames vacant at any moment

    for(int i = 0; i < n; i++) frames[i] = -1; //frames[i] == -1 indicates
that frame is empty
    //frames[i] != -1 indicates that page number = frames[i] is stored at
frames[i];

    int pageFaults = 0;

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

        int flag = 0;   //a flag to check if the new page is already in the
frames or not
        //if flag == 1, page is present already
        //else it is required to store it

        //this loop checks if new page is already present or not
        for(int j = 0; j < n; j++){
            if(frames[j] == referenceString[i]){
                flag = 1;
                break;
            }
        }

        //if new page is already present, there is no page fault, hence we can
continue to next page
        if(flag == 1) continue;
        //if there are empty frames, and a page is not already present
        //we need to put this page in the empty space.
        //this will cause a page fault
        else if(vacantFrames < n){
            frames[vacantFrames] = referenceString[i];
            vacantFrames++;
            pageFaults++;
        }
        //else, if page is not present and no frames are vacant, we need to
replace the page, and it will also cause a page fault
```

```
        else{
            int frameNum = rand() % 5; //each frame is equally likely to be
replaced
            //printf("%d ", frameNum);
            frames[frameNum] = referenceString[i];
            pageFaults++;
        }
    }
    printf("\n");

    return pageFaults;
}


//Approach 2
int weightedRPR(int s, int referenceString[s], int n){

    int frames[n];
    int vacantFrames = 0;

    for(int i = 0; i < n; i++) frames[i] = -1; //frames[i] == -1 indicates
that frame is empty
    //frames[i] != -1 indicates that page number = frames[i] is stored at
frames[i];

    int pageFaults = 0;

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

        int flag = 0;   //a flag to check if the new page is already in the
frames or not
        //if flag == 1, page is present already
        //else it is required to store it

        //this loop checks if new page is already present or not
        for(int j = 0; j < n; j++){
            if(frames[j] == referenceString[i]){
                flag = 1;
                break;
            }
        }

        //if new page is already present, there is no page fault, hence we can
continue to next page
        if(flag == 1) continue;
        //if there are empty frames, and a page is not already present
        //we need to put this page in the empty space.
```

```
        //this will cause a page fault
        else if(vacantFrames < n){
            frames[vacantFrames] = referenceString[i];
            vacantFrames++;
            pageFaults++;
        }
        //else, if page is not present and no frames are vacant, we need to
replace the page, and it will also cause a page fault
        else{
            int randomNumber = rand() % 10;
            int frameNum = -1;


            //using switch case to create an distribution among the frame to
be replaced
            //given frame 0 and 4 has 10% chances to be replaced
            //frame 1 and 3 has 20% chances to be replaced
            //frame 2 has 40% chances to be replaced

            //giving 1 number to frame 0 out of {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
generated by random function
            //will set its probability to be 10%.

            //Similarly giving 2 numbers (1, 2) to frame 1 will set its
probability to be 20%;

            switch(randomNumber){
                case 0:
                    frameNum = 0;
                    break;
                case 1:
                case 2: //no break used, so randomNumber == 1 or randomNumber
== 2, both points to frame 1
                    frameNum = 1;
                    break;
                case 3:
                case 4:
                case 5:
                case 6:  //no break used, so randomNumber in {3, 4, 5, 6}
points to frame 2
                    frameNum = 2;
                    break;
                case 7:
                case 8: //no break used, so randomNumber == 7 or randomNumber
== 8, both points to frame 3
                    frameNum = 3;
                    break;
```

```
                case 9:
                    frameNum = 4;
                    break;
                default:  //not really required
                    frameNum = -1;
            }

            //printf("%d ", frameNum);
            frames[frameNum] = referenceString[i];
            pageFaults++;
        }
    }
    printf("\n");

    return pageFaults;
}
```

Output 1

```
archit@archit-VirtualBox:~/Desktop$ ./RPR
Enter the length of reference string (>= 20) : 25
Reference String : 2, 5, 8, 6, 8, 0, 2, 3, 2, 7, 3, 0, 7, 2, 8, 9, 7, 7, 2, 5, 2, 3, 0, 3, 9

Page Faults in Equally Likely RPR (Approach 1) = 13

Page Faults in Weighted RPR (Approach 2) = 15
archit@archit-VirtualBox:~/Desktop$
```

Output 2

```
archit@archit-VirtualBox:~/Desktop$ ./RPR
Enter the length of reference string (>= 20) : 30
Reference String : 7, 9, 5, 0, 0, 6, 0, 9, 1, 4, 5, 4, 8, 3, 2, 6, 4, 3, 6, 1, 0, 2, 7, 9, 4, 4, 3, 3, 5, 3

Page Faults in Equally Likely RPR (Approach 1) = 22

Page Faults in Weighted RPR (Approach 2) = 18
archit@archit-VirtualBox:~/Desktop$
```

Output 3

```
architagarchit-VirtualBox:~/Desktop$ ./RPR
Enter the length of reference string (>= 20) : 35
Reference String : 1, 3, 0, 5, 7, 0, 2, 2, 4, 8, 1, 5, 8, 1, 8, 4, 5, 8, 4, 8, 9, 0, 0, 3, 6, 7, 1, 6, 1, 6, 3, 4, 9, 5, 9

Page Faults in Equally Likely RPR (Approach 1) = 21

Page Faults in Weighted RPR (Approach 2) = 19
architagarchit-VirtualBox:~/Desktop$
```

Output 4

```
architagarchit-VirtualBox:~/Desktop$ ./RPR
Enter the length of reference string (>= 20) : 34
Reference String : 9, 4, 9, 1, 8, 1, 0, 3, 2, 7, 5, 6, 5, 4, 2, 5, 1, 3, 5, 4, 1, 4, 3, 1, 8, 4, 0, 8, 8, 8, 1, 9, 5, 1

Page Faults in Equally Likely RPR (Approach 1) = 18

Page Faults in Weighted RPR (Approach 2) = 21
architagarchit-VirtualBox:~/Desktop$
```

Output 5

```
architagarchit-VirtualBox:~/Desktop$ ./RPR
Enter the length of reference string (>= 20) : 20
Reference String : 6, 6, 2, 7, 8, 3, 2, 6, 4, 6, 2, 2, 0, 3, 8, 7, 5, 2, 3, 9

Page Faults in Equally Likely RPR (Approach 1) = 10

Page Faults in Weighted RPR (Approach 2) = 14
architagarchit-VirtualBox:~/Desktop$
```