Name: Adwait S Purao

UID: 2021300101

Batch: B2

Branch: Computer Engineering

Experiment No. 7

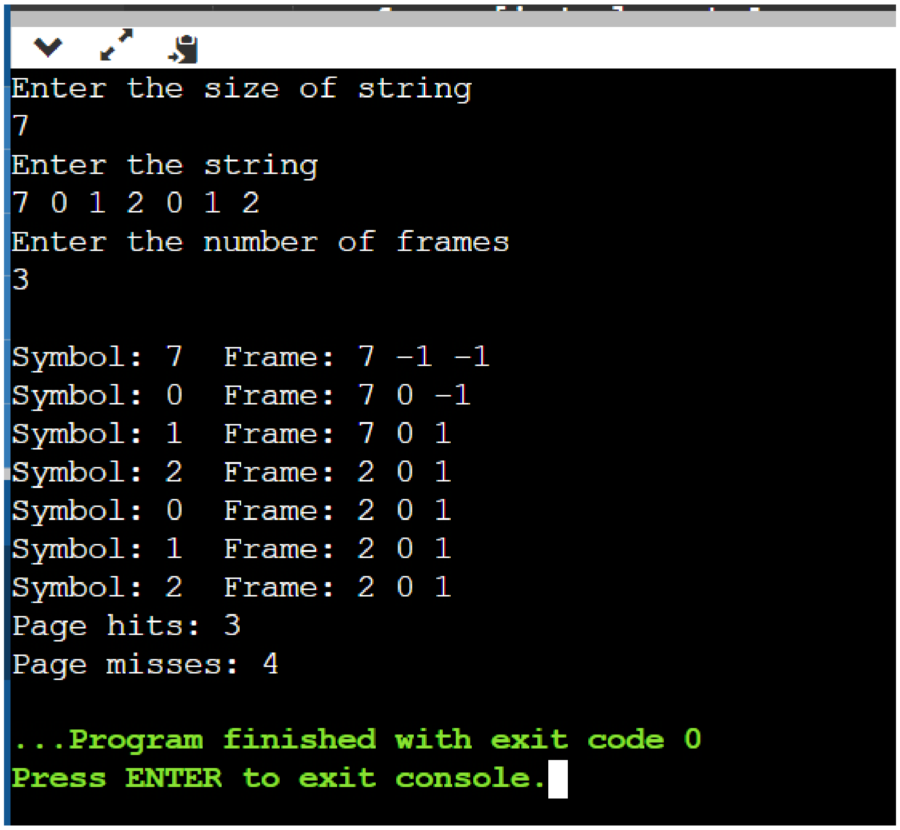
Page replacement algorithms:

1. **First In First out.**

Code :

```
#include<stdio.h>

void fifo(int string[20],int n,int size)
{

  int frames[n];

  for (int i=0;i<n;i++)
    frames[i]=-1;



  int index=-1;



  int page_miss=0;
  int page_hits=0;



  for (int i=0;i<size;i++)
  {
    int symbol=string[i];
    int flag=0;

    for(int j=0;j<n;j++)
    {
        if (symbol==frames[j])
        {
            flag=1;
            break;
        }
    }

    if (flag==1)
    {
        printf("\nSymbol: %d  Frame: ",symbol);
        for (int j=0;j<n;j++)
            printf("%d ",frames[j]);
        page_hits+=1;
    }
    else
    {
        index=(index+1)%n;
        frames[index]=symbol;
        page_miss+=1;
        printf("\nSymbol: %d  Frame: ",symbol);
        for (int j=0;j<n;j++)
            printf("%d ",frames[j]);
    }
  }
  printf("\nPage hits: %d",page_hits);
  printf("\nPage misses: %d",page_miss);
}

int main(void)
{
  int n;
  printf("Enter the size of string\n");
  scanf("%d",&n);
  int string[n];
  printf("Enter the string\n");
  for(int i=0;i<n;i++){
    scanf("%d",&string[i]);
  }
  int nf;
  printf("Enter the number of frames\n");
  scanf("%d",&nf);
  int size=sizeof(string)/sizeof(int);
  fifo(string,nf,size);
  return 0;
}
```

Output:



```
Enter the size of string
7
Enter the string
7 0 1 2 0 1 2
Enter the number of frames
3

Symbol: 7  Frame: 7 -1 -1
Symbol: 0  Frame: 7 0 -1
Symbol: 1  Frame: 7 0 1
Symbol: 2  Frame: 2 0 1
Symbol: 0  Frame: 2 0 1
Symbol: 1  Frame: 2 0 1
Symbol: 2  Frame: 2 0 1
Page hits: 3
Page misses: 4

...Program finished with exit code 0
Press ENTER to exit console.
```

2. **Optimal Page Replacement.**

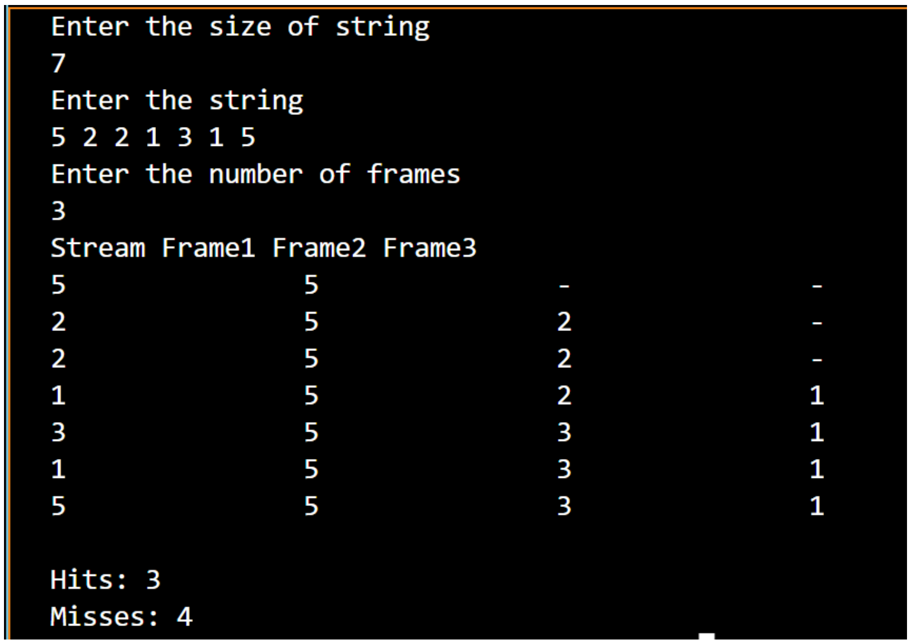
Code:

```
#include<stdio.h>


int search(int key, int frame_items[], int frame_occupied)
{
    for (int i = 0; i < frame_occupied; i++)
        if (frame_items[i] == key)
            return 1;
    return 0;
}
```

```
void printOuterStructure(int nf){
  printf("Stream ");

  for(int i = 0; i < nf; i++)
    printf("Frame%d ", i+1);
}
void printCurrFrames(int item, int frame_items[], int frame_occupied, int nf){


  printf("\n%d \t\t", item);


  for(int i = 0; i < nf; i++){
    if(i < frame_occupied)
      printf("%d \t\t", frame_items[i]);
    else
      printf("- \t\t");
  }
}

int predict(int string[], int frame_items[], int refStrLen, int index, int frame_occupied)
{

  int result = -1, farthest = index;
  for (int i = 0; i < frame_occupied; i++) {
    int j;
    for (j = index; j < refStrLen; j++)
    {
      if (frame_items[i] == string[j])
      {
        if (j > farthest) {
          farthest = j;
          result = i;
```

```
                }
                break;
            }
        }


        if (j == refStrLen)
            return i;
    }


    return (result == -1) ? 0 : result;
}

void optimalPage(int string[], int refStrLen, int frame_items[], int nf)
{

    int frame_occupied = 0;
    printOuterStructure(nf);


    int hits = 0;
    for (int i = 0; i < refStrLen; i++) {


        if (search(string[i], frame_items, frame_occupied)) {
            hits++;
            printCurrFrames(string[i], frame_items, frame_occupied, nf);
            continue;
        }


        if (frame_occupied < nf){
            frame_items[frame_occupied] = string[i];
            frame_occupied++;
            printCurrFrames(string[i], frame_items, frame_occupied, nf);
        }

        else {
            int pos = predict(string, frame_items, refStrLen, i + 1, frame_occupied);
            frame_items[pos] = string[i];
            printCurrFrames(string[i], frame_items, frame_occupied, nf);
        }

    }
    printf("\n\nHits: %d\n", hits);
    printf("Misses: %d", refStrLen - hits);
}


int main()
{
    int n;
    printf("Enter the size of string\n");
    scanf("%d",&n);
    int string[n];
    printf("Enter the string\n");
    for(int i=0;i<n;i++){
        scanf("%d",&string[i]);
    }
    int nf;
    printf("Enter the number of frames\n");
    scanf("%d",&nf);
    int refStrLen = sizeof(string) / sizeof(string[0]);
    int frame_items[nf];
    optimalPage(string, refStrLen, frame_items, nf);
    return 0;
}
```

Output:



```
Enter the size of string
7
Enter the string
5 2 2 1 3 1 5
Enter the number of frames
3
Stream Frame1 Frame2 Frame3
5               5               -               -
2               5               2               -
2               5               2               -
1               5               2               1
3               5               3               1
1               5               3               1
5               5               3               1

Hits: 3
Misses: 4
```

3. Least recently Used

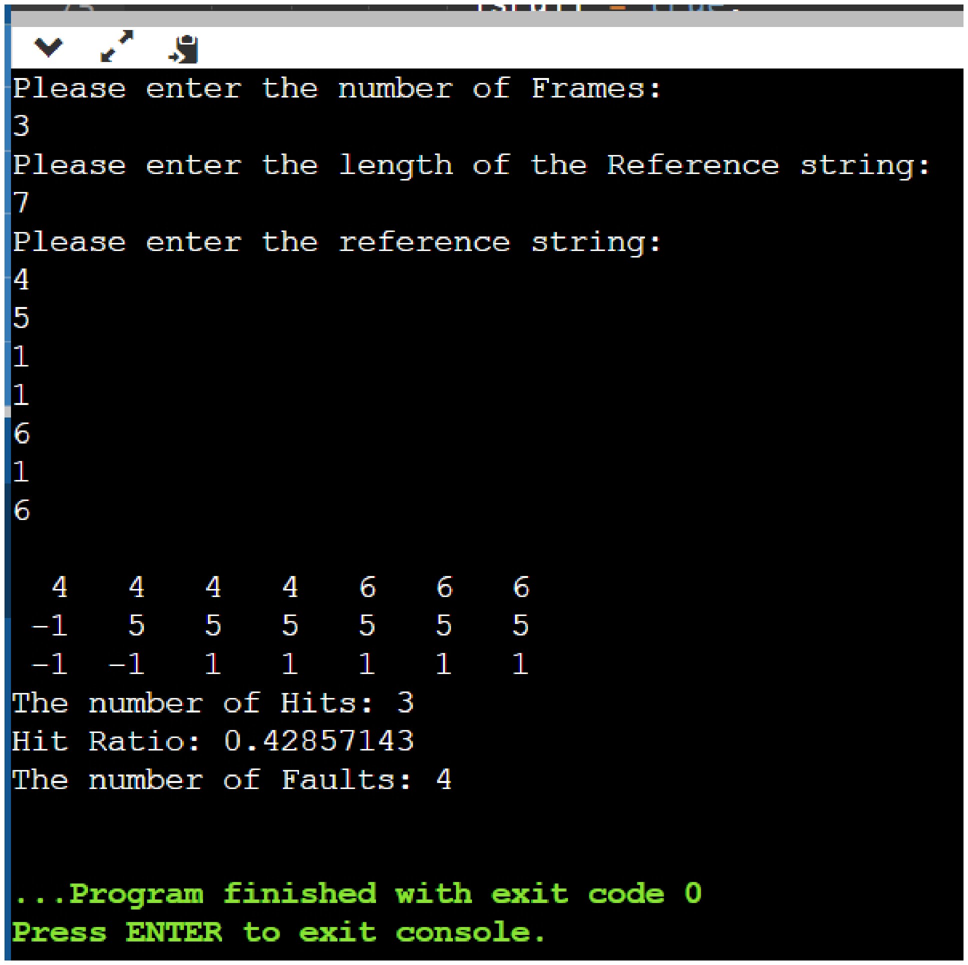
Code:

```
import java.io.*;

import java.util.*;


public class Main {


  public static void main(String[] args) throws IOException

  {

    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

    int frames,pointer = 0, hit = 0, fault = 0,ref_len;

    Boolean isFull = false;

    int buffer[];

    ArrayList<Integer> stack = new ArrayList<Integer>();

    int reference[];

    int mem_layout[][];


    System.out.println("Please enter the number of Frames: ");

    frames = Integer.parseInt(br.readLine());


    System.out.println("Please enter the length of the Reference string: ");

    ref_len = Integer.parseInt(br.readLine());
```

```
reference = new int[ref_len];
mem_layout = new int[ref_len][frames];
buffer = new int[frames];
for(int j = 0; j < frames; j++)
      buffer[j] = -1;

System.out.println("Please enter the reference string: ");
for(int i = 0; i < ref_len; i++)
{
   reference[i] = Integer.parseInt(br.readLine());
}
System.out.println();
for(int i = 0; i < ref_len; i++)
{
   if(stack.contains(reference[i]))
   {
    stack.remove(stack.indexOf(reference[i]));
   }
   stack.add(reference[i]);
   int search = -1;
   for(int j = 0; j < frames; j++)
   {
      if(buffer[j] == reference[i])
      {
         search = j;
         hit++;
         break;
      }
   }
   if(search == -1)
   {
    if(isFull)
    {
      int min_loc = ref_len;
            for(int j = 0; j < frames; j++)
            {
             if(stack.contains(buffer[j]))
                {
                    int temp = stack.indexOf(buffer[j]);
                    if(temp < min_loc)
                    {
                        min_loc = temp;
                        pointer = j;
                    }
                }
            }
     }
        buffer[pointer] = reference[i];
        fault++;
        pointer++;
        if(pointer == frames)
        {
         pointer = 0;
         isFull = true;
        }
    }
    for(int j = 0; j < frames; j++)
        mem_layout[i][j] = buffer[j];
}

for(int i = 0; i < frames; i++)
{
    for(int j = 0; j < ref_len; j++)
        System.out.printf("%3d ",mem_layout[j][i]);
    System.out.println();
}
```

```
        System.out.println("The number of Hits: " + hit);

        System.out.println("Hit Ratio: " + (float)((float)hit/ref_len));

        System.out.println("The number of Faults: " + fault);

    }


}
```

Output:



```
Please enter the number of Frames:
3
Please enter the length of the Reference string:
7
Please enter the reference string:
4
5
1
1
6
1
6

  4   4   4   4   6   6   6
 -1   5   5   5   5   5   5
 -1  -1   1   1   1   1   1
The number of Hits: 3
Hit Ratio: 0.42857143
The number of Faults: 4


...Program finished with exit code 0
Press ENTER to exit console.
```

Conclusion:

In the above experiment we learnt about the various page replacement algorithms and implemented the code in c and java.