# “Mini Project on – Page Replacement Policies”

## The Background Concept

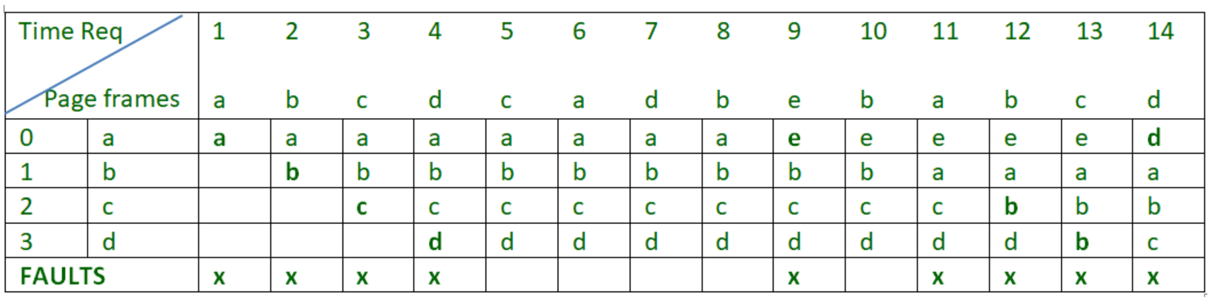
In a operating systems that use paging for memory management, page replacement algorithm are needed to decide which page needed to be replaced when new page comes in. Whenever a new page is referred and not present in memory, page fault occurs and Operating System replaces one of the existing pages with newly needed page. Different page replacement algorithms suggest different ways to decide which page to replace. The target for all algorithms is to reduce number of page faults.

Page Fault – A page fault is a type of interrupt, raised by the hardware when a running program accesses a memory page that is mapped into the virtual address space, but not loaded in physical memory.

## Page Replacement Algorithms :

### First In First Out (FIFO) –

This is the simplest page replacement algorithm. In this algorithm, operating system keeps track of all pages in the memory in a queue, oldest page is in the front of the queue. When a page needs to be replaced page in the front of the queue is selected for removal.
Let’s have a reference string: a, b, c, d, c, a, d, b, e, b, a, b, c, d and the size of the frame be 4.



| Time Req | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Page frames | | a | b | c | d | c | a | d | b | e | b | a | b | c | d |
| 0 | a | **a** | a | a | a | a | a | a | a | **e** | e | e | e | e | **d** |
| 1 | b | | **b** | b | b | b | b | b | b | b | b | a | a | a | a |
| 2 | c | | | **c** | c | c | c | c | c | c | c | c | **b** | b | b |
| 3 | d | | | | **d** | d | d | d | d | d | d | d | d | **b** | c |
| FAULTS | | x | x | x | x | | | | | x | | x | x | x | x |

There are 9 page faults using FIFO algorithm.

### Least Recently Used –

In this algorithm page will be replaced which is least recently used.

Let’s have a reference string: a, b, c, d, c, a, d, b, e, b, a, b, c, d and the size of the frame be 4.

| Time Req |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Page frames |  | a | b | c | d | c | a | d | b | e | b | a | b | c | d |
| 0 | a | **a** | a | a | a | a | a | a | a | a | e | e | e | e | e |
| 1 | b |  | **b** | b | b | b | b | b | b | b | b | a | a | a | a |
| 2 | c |  |  | **c** | c | c | c | c | c | **e** | c | c | b | b | **d** |
| 3 | d |  |  |  | **d** | d | d | d | d | d | d | d | d | **c** | c |
| **FAULTS** |  | **x** | **x** | **x** | **x** |  |  |  |  | **x** |  |  |  | **x** | **x** |

There are 7 page faults using LRU algorithm.

**OPTIMAL**

In this algorithm, pages are replaced which are not used for the longest duration of time in the future.

Optimal page replacement is perfect, but not possible in practice as operating system cannot know future requests. The use of Optimal Page replacement is to set up a benchmark so that other replacement algorithms can be analysed against it.

| Time Req |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Page frames |  | a | b | c | d | c | a | d | b | e | b | a | b | c | d |
| 0 | a | **a** | a | a | a | a | a | a | a | **a** | e | e | e | e | **d** |
| 1 | b |  | **b** | b | b | b | b | b | b | b | b | a | a | a | a |
| 2 | c |  |  | **c** | c | c | c | c | c | c | c | c | b | b | b |
| 3 | d |  |  |  | **d** | d | d | d | d | **e** | d | d | d | c | c |
| **FAULTS** |  | **x** | **x** | **x** | **x** |  |  |  |  | **x** |  |  |  |  | **x** |

There are 6 page faults using optimal algorithm.

## The Implementation

This project has been very efficiently implemented on the Java Applet technology, serving end users top notch form of graphical user interface. This project provides all the above-mentioned page replacement policy algorithm with keeping all the constraints. This project requires only a JRE, to run the applet window. The checkInitFrame() method is used to check whether the element present currently in frame are all unique if not then place next element. Based on checkInitFrame() method, In the case of Fifo algo, the next string is being stored and if string don’t match any of frame element then replacing it and incrementing the fault. Likewise, in LRU also the only difference is here the string is replaced by means of unmatched element-framesize kth element .In optimal, a new array is used to check with frame content which string should be replaced and which shouldnt

## #Source Code

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;
import java.util.*;
import java.math.*;

//<applet code="OSmini.class" width="770" height="400"></applet>
public class OSmini extends JApplet implements ActionListener
{
        int i,psn,fs,fault=0,hit;
        static int count=0;
        int frame[]=new int[100];
        int temp[]=new int[100];
        int[] sseq = new int[100];
        int temp1[]=new int[100];
        Border br=BorderFactory.createLineBorder(Color.BLACK,5);
        StringBuilder builder = new StringBuilder();
        String choice,seq_str,s,st;
        JTextField jtf1, jtf2, jtf3;
        JLabel jlb1,jlb2,jlb3,jlb4,jlb5,jlb6,jlb7,jhit,jfault,jlbframe,jlbseq,opt,jlblast,jlbreset;
        JComboBox cb = new JComboBox();
        JComboBox cb1 = new JComboBox();
        JComboBox cb2 = new JComboBox();
        JButton jb1;
        Font font,font1,font2;
        Container co;
//**********************************************************************
//**************
        int checkInitFrame(int frame[],int fs,int sseq[],int k)
        {
                int i,j;
                for(i=0;i<fs;i++)
                {
                        for(j=i+1;j<fs;j++)
                        {
                                if(frame[i]==frame[j])
                                {
                                        frame[j]=-1;
                                }
                        }
                        if(frame[i]!=-1)
                        {
                                StringBuilder builder=new StringBuilder();
                                builder.append(frame[i]);
                                jlbframe.setText(jlbframe.getText()+builder.toString());
                                System.out.print(frame[i]);
                        }
                        if(frame[i]==-1)
                        {
                                frame[i]=sseq[i+k];
                        k++;
                        count++;
                        StringBuilder builder=new StringBuilder();
                        builder.append(" | ");
                        jlbframe.setText(jlbframe.getText()+builder.toString());
                        System.out.print("  ");
                        checkInitFrame(frame,fs,sseq,k);
                    }
                }
                return frame[i];
            }
//************************************************************************
//**************
            void fifo(int frame[],int sseq[],int fs,int fi,int fip,int psn,int fault)
            {
                int i,ji,k;
                temp[0]=sseq[fs-1+count+fi];
                for(i=0;i<psn;i++)
                {
                    for(ji=0;ji<fs;ji++)
                    {
                        if(temp[0]==frame[ji])
                        {
                            StringBuilder builder=new StringBuilder();
                            builder.append(" | ");
                            jlbframe.setText(jlbframe.getText()+builder.toString());
                            System.out.print("  ");
                            for(k=0;k<fs;k++){
                                StringBuilder builder1=new StringBuilder();
                                builder1.append(frame[k]);
                                 jlbframe.setText(jlbframe.getText()
                                +builder1.toString());
                                System.out.print(frame[k]);
                            }
                            fi+=1;
                            temp[0]=sseq[fs-1+count+fi];

                        }
                    }
                }
                if(sseq[fs-1+count+fi]!=sseq[fs-1+count+fi+1])
                {
                    fault=fault+1;
                    frame[fip]=temp[0];
                }
                fip=(fip+1)%fs;
                fi+=1;
                if(fi<psn-(fs-1+count))
                {
                        StringBuilder builder=new StringBuilder();
                        builder.append(" | ");
                        jlbframe.setText(jlbframe.getText()+builder.toString());
                        System.out.print("  ");
                        for(k=0;k<fs;k++)
                        {
                                StringBuilder builder2=new StringBuilder();
                                builder2.append(frame[k]);
                                jlbframe.setText(jlbframe.getText()+builder2.toString());
                                System.out.print(frame[k]);
                        }
                }
                try{
                if(fi<psn)
                        fifo(frame,sseq,fs,fi,fip,psn,fault);
                }
                catch(ArrayIndexOutOfBoundsException ae){
                                fault+=1;
                                frame[fip]=temp[0];
                                StringBuilder buildersl=new StringBuilder();
                                buildersl.append(" | ");
                                jlbframe.setText(jlbframe.getText()+buildersl.toString());
                                System.out.print("  ");
                                for(i=0;i<fs;i++)
                                {
                                        StringBuilder builderl=new StringBuilder();
                                        builderl.append(frame[i]);
                                        jlbframe.setText(jlbframe.getText()+
                                        builderl.toString());
                                        System.out.print(frame[i]);
                                }
                                jlb6.setText("Hit = "+Integer.toString(psn-fault));
        //displaying hit and fault count
                                jlb7.setText("Fault = "+Integer.toString(fault));
                }
        }
//****************************************************************************
//**************
        void lru_algo(int frame[],int sseq[],int fs,int fi,int fip,int psn,int fault)
        {
                int i,ji,lru,k;
                temp[0]=sseq[0+fi];
                for(i=0;i<psn;i++)
```

```
{
        for(ji=0;ji<fs;ji++)
        {
                if(temp[0]==frame[ji])
                {
                        if(fi>fs+1)
                        {
                                StringBuilder builder=new StringBuilder();
                                builder.append(" | ");
                                jlbframe.setText(jlbframe.getText()+
                                builder.toString());
                                System.out.print("  ");
                                for(k=0;k<fs;k++)
                                {
                                StringBuilder builder1=new StringBuilder();
                                builder1.append(frame[k]);
                                        jlbframe.setText(jlbframe.getText()+
                                        builder1.toString());
                                        System.out.print(frame[k]);
                                }
                        }
                        fi+=1;
                        temp[0]=sseq[0+fi];
                }
        }
}

lru=sseq[fi-fs];

for(i=0;i<fs;i++)
{
        if(lru==frame[i])
        {
                if(sseq[0+fi]!=sseq[0+fi+1])
                {
                        fault=fault+1;
                        frame[i]=temp[0];
                }
        }
}
fi+=1;
if(fi<psn)
{
        StringBuilder builder2=new StringBuilder();
        builder2.append(" | ");
        jlbframe.setText(jlbframe.getText()+builder2.toString());
```

```
                        System.out.print("  ");
                        for(k=0;k<fs;k++)
                        {
                                StringBuilder builder3=new StringBuilder();
                                builder3.append(frame[k]);
                                jlbframe.setText(jlbframe.getText()+builder3.toString());
                                System.out.print(frame[k]);
                        }
                }
                try{
                if(fi<psn)
                        lru_algo(frame,sseq,fs,fi,fip,psn,fault);
                }
                catch(ArrayIndexOutOfBoundsException ae){
                        lru=sseq[fi-fs];
                        StringBuilder buildersl=new StringBuilder();
                        buildersl.append(" | ");
                        jlbframe.setText(jlbframe.getText()+buildersl.toString());
                        System.out.print("  ");
                        for(i=0;i<fs;i++)
                        {
                                if(lru==frame[i])
                                {
                                        fault=fault+1;
                                        frame[i]=temp[0];
                                }
                                for(i=0;i<fs;i++)
                                {
                                        StringBuilder builderl=new StringBuilder();
                                        builderl.append(frame[i]);
                                        jlbframe.setText(jlbframe.getText()+
                                        builderl.toString());
                                        System.out.print(frame[i]);
                                }
                        }
                        jlb6.setText("Hit = "+Integer.toString(psn-fault));
                        jlb7.setText("Fault = "+Integer.toString(fault));
                }

        }
//***************************************************************************
**************
        void optimal(int frame[],int sseq[],int fs,int fi,int fip,int psn,int fault)
        {
                int i,ji,j,optim[]=new int[100],k=0,kl;
                temp[0]=sseq[0+fi];
```

```
for(i=0;i<psn;i++)
{
        for(ji=0;ji<fs;ji++)
        {
                if(temp[0]==frame[ji])
                {
                        if(fi>fs+1)
                        {
                                StringBuilder builder=new StringBuilder();
                                builder.append(" | ");
                                jlbframe.setText(jlbframe.getText()+
                                builder.toString());
                                System.out.print("  ");
                                for(kl=0;kl<fs;kl++){
                                StringBuilder builderl=new StringBuilder();
                                        builderl.append(frame[kl]);
                                        jlbframe.setText(jlbframe.getText()+
                                        builderl.toString());
                                        System.out.print(frame[kl]);
                                }
                        }
                        fi+=1;
                        temp[0]=sseq[0+fi];
                }
        }
}

for(i=0;i<fs;i++)
{
        optim[i]=sseq[fi+1+i];
}

for(i=0;i<fs;i++)
{
        for(j=0;j<fs;j++)
        {
                if(frame[i]==optim[j])
                {
                        temp1[i]=frame[i];
                }
        }
        if(temp1[i]==0)
        {
        k=i;
        }
        temp1[i]=0;
```

```
}

if(sseq[0+fi]!=sseq[0+fi+1]&&sseq[fi]>0)
{
	fault=fault+1;
	frame[k]=temp[0];
}

fi+=1;
if(fi<psn)
{
	StringBuilder builder2=new StringBuilder();
	builder2.append(" | ");
	jlbframe.setText(jlbframe.getText()+builder2.toString());
	System.out.print("  ");
	for(kl=0;kl<fs;kl++)
	{
		StringBuilder builder3=new StringBuilder();
		builder3.append(frame[kl]);
		jlbframe.setText(jlbframe.getText()+builder3.toString());
		System.out.print(frame[kl]);
	}
}
try{
if(fi<psn)
	optimal(frame,sseq,fs,fi,fip,psn,fault);
}
catch(ArrayIndexOutOfBoundsException ae){
optim[0]=sseq[fi];
for(i=1;i<fs;i++)
{
	optim[i]=0;
}
for(i=0;i<fs;i++)
{
	for(j=0;j<fs;j++)
	{
		if(frame[i]==optim[j])
		{
			temp1[i]=frame[i];
		}
	}
	if(temp1[i]==0)
	{
	k=i;
	}
```

```
        temp1[i]=0;
}
if(sseq[0+fi]!=sseq[0+fi+1]&&sseq[fi]>0)
{
        fault=fault+1;
        frame[k]=temp[0];
}
StringBuilder buildersl=new StringBuilder();
buildersl.append(" | ");
jlbframe.setText(jlbframe.getText()+buildersl.toString());
System.out.print("  ");
for(i=0;i<fs;i++)
{
        StringBuilder builderssl=new StringBuilder();
        builderssl.append(frame[i]);
        jlbframe.setText(jlbframe.getText()+builderssl.toString());
        System.out.print(frame[i]);
}
temp[0]=sseq[psn-1];
k=0;
for(i=0;i<fs;i++)
{
        if(temp[0]==frame[i])
        {
                k=i;
        }
}
if(k==0)
{
        fault=fault+1;
        frame[k]=temp[0];
}
StringBuilder buildersll=new StringBuilder();
buildersll.append(" | ");
jlbframe.setText(jlbframe.getText()+buildersll.toString());
System.out.print("  ");
for(i=0;i<fs;i++)
{
        StringBuilder builderl=new StringBuilder();
        builderl.append(frame[i]);
        jlbframe.setText(jlbframe.getText()+builderl.toString());
        System.out.print(frame[i]);
}


if(fs<4)
```

```
		{
			temp[0]=sseq[psn-2];
			k=0;
			for(i=0;i<fs;i++)
			{
				if(temp[0]==frame[i])
				{
					k=i;
				}
			}
			if(k==0)
			{
				fault=fault+1;
				frame[k]=temp[0];
			}
			StringBuilder builderslll=new StringBuilder();
			builderslll.append(" | ");
			jlbframe.setText(jlbframe.getText()+builderslll.toString());
			System.out.print("  ");
			for(i=0;i<fs;i++)
			{
				StringBuilder builderll=new StringBuilder();
				builderll.append(frame[i]);
				jlbframe.setText(jlbframe.getText()+builderll.toString());
				System.out.print(frame[i]);
			}
		}

		jlb6.setText("Hit = "+Integer.toString(psn-fault));
		jlb7.setText("Fault = "+Integer.toString(fault));
		}

	}
//**************************************************************************
//**************
	public void init()
	{
		co = getContentPane();
		co.setLayout(null);
		setVisible(true);

		font = new Font("Times New Roman", Font.BOLD,14);
		font1 = new Font("Times New Roman", Font.BOLD,16);
		font2 = new Font("Times New Roman", Font.BOLD,24);

		jlb1=new JLabel("No. Of String ");
```

```
jlb1.setBounds(80,23,90,23);

cb1.addItem("Select");
cb1.addItem("1");cb1.addItem("2");cb1.addItem("3");cb1.addItem("4");
cb1.addItem("5");cb1.addItem("6");cb1.addItem("7");cb1.addItem("8");
cb1.addItem("9");cb1.addItem("10");cb1.addItem("11");cb1.addItem("12");
cb1.addItem("13");cb1.addItem("14");cb1.addItem("15");cb1.addItem("16");
cb1.addItem("16");cb1.addItem("18");cb1.addItem("19");cb1.addItem("20");
cb1.setBounds(165,23,125,23);

jlb2=new JLabel("Enter String ");
jlb2.setBounds(318,23,300,23);

jtf2 = new JTextField();
jtf2.setBounds(395,23,300,23);

jlb3=new JLabel("Frame Size ");
jlb3.setBounds(80,50,70,50);

cb2.addItem("Select");cb2.addItem("3");cb2.addItem("4");
cb2.setBounds(165,65,125,22);

jlb4=new JLabel("Choose Algo ");
jlb4.setBounds(318,50,300,50);

cb.addItem("Select");
cb.addItem("FIFO");
cb.addItem("LRU");
cb.addItem("Optimal");
cb.setBounds(395,65,70,22);

jb1 = new JButton("Submit");
jb1.setBounds(80,100,616,30);

String dash="-";
for(int i=0;i<250;i++)
        dash=dash+'-';

jlb5=new JLabel(dash);
jlb5.setBounds(0,137,775,5);

jlb6=new JLabel("");
jlb6.setBounds(290,285,54,30);
jlb6.setOpaque(true);
jlb6.setBackground(Color.CYAN);
```

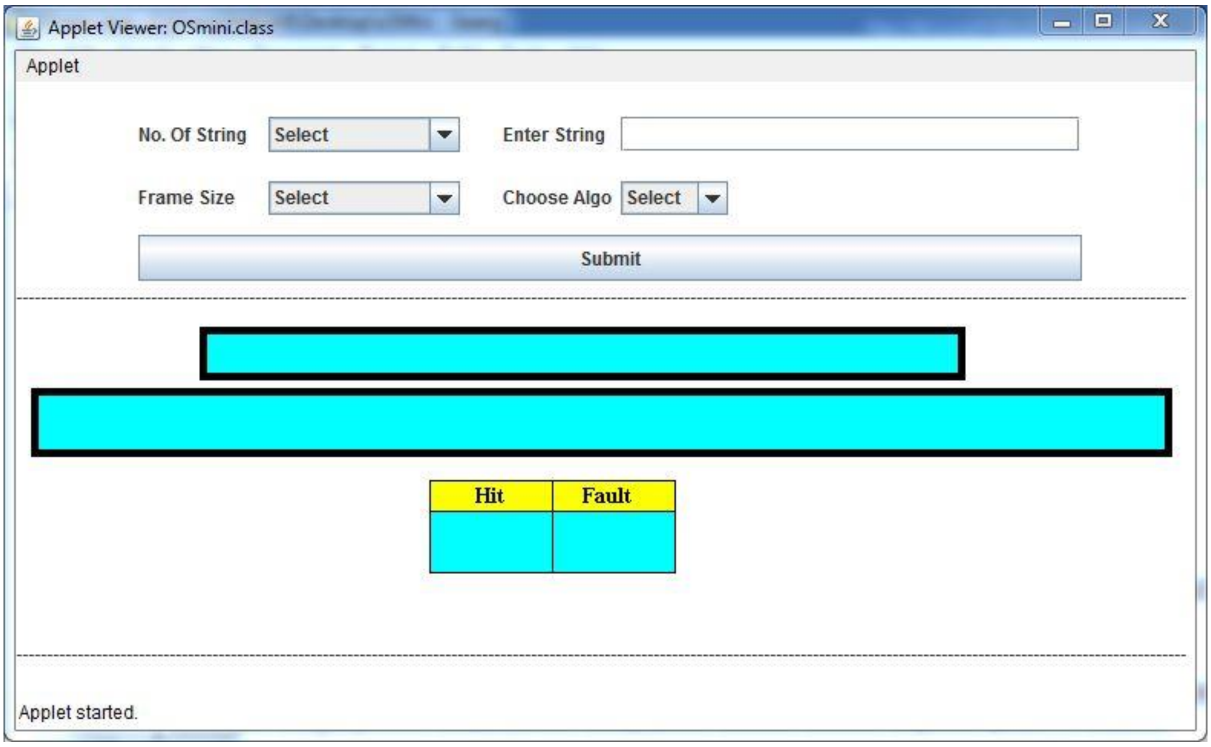
```
jlb7=new JLabel("");
jlb7.setBounds(360,285,54,30);
jlb7.setOpaque(true);
jlb7.setBackground(Color.CYAN);

jlbseq=new JLabel();
jlbseq.setBounds(120,160,500,35);
jlbseq.setOpaque(true);
jlbseq.setBackground(Color.CYAN);
jlbseq.setBorder(br);
jlbseq.setFont(font);

jlbframe=new JLabel();
jlbframe.setBounds(10,200,745,45);
jlbframe.setOpaque(true);
jlbframe.setBackground(Color.CYAN);
jlbframe.setBorder(br);
jlbframe.setFont(font);

jb1.addActionListener(this);

opt=new JLabel();
opt.setBounds(210,330,400,50);
opt.setFont(font2);
opt.setForeground(Color.BLUE);

jlblast=new JLabel(dash);
jlblast.setBounds(0,370,775,5);

co.add(jlb1);
co.add(cb1);
co.add(jlb2);
co.add(jtf2);
co.add(jlb3);
co.add(cb2);
co.add(jlb4);
co.add(cb);
co.add(jb1);
co.add(jlb5);
co.add(jlb6);
co.add(jlb7);
co.add(jlbframe);
co.add(jlbseq);
co.add(opt);
co.add(jlblast);
co.setBackground(Color.WHITE);
```

```
                repaint();
        }
        public void paint(Graphics g)
        {
    super.paint(g);  // fixes the immediate problem.

    g.setColor(Color.YELLOW);
    g.fillRect(270,260,160,20);                    //hit fault header

    g.setColor(Color.CYAN);
    g.fillRect(270,280,160,40);                    //hit fault content

    g.setColor(Color.BLACK);
    g.drawLine(350,260,350,320);          //line inner rect

    g.setColor(Color.BLACK);
    g.drawLine(270,260,270,320);          //line left most inner rect

    g.setColor(Color.BLACK);
    g.drawLine(430,260,430,320);          //line right most inner rect

    g.setColor(Color.BLACK);
    g.drawLine(270,260,430,260);          //line uppermost most inner rect

    g.setColor(Color.BLACK);
    g.drawLine(270,280,430,280);          //line middlemost most inner rect

    g.setColor(Color.BLACK);
    g.drawLine(270,320,430,320);          //line Bottommost most inner rect

    g.setFont(font);
    g.drawString("Hit",300,275);
    g.drawString("Fault",370,275);
}




        public void actionPerformed(ActionEvent ke)
        {
                if(ke.getSource() == jb1)
                {
                        st=(String)(cb1.getSelectedItem());
                        psn = Integer.parseInt(st);
                        seq_str = jtf2.getText();
```

```
String[] temp_str = seq_str.split("");
int[] sseq = new int[temp_str.length];

StringBuilder bs=new StringBuilder();
for (int i = 0; i < sseq.length; i++)
{
        sseq[i] = Integer.parseInt(temp_str[i]);
        bs.append("   "+sseq[i]);

}
jlbseq.setText("String Sequence ");
jlbseq.setText(jlbseq.getText()+bs.toString());

st=(String)(cb2.getSelectedItem());
fs = Integer.parseInt(st);
choice = (String)cb.getSelectedItem();

for(i=0;i<fs;i++)
{
        if(frame[i]==0)
        {
                frame[i]=sseq[i];
                fault+=1;
        }
}

if(frame[0]!=frame[1]&&frame[1]!=frame[2]&&fs<4)
{
        builder.append(frame[0]+" | "+frame[0]+frame[1]+" | ");

        jlbframe.setText(jlbframe.getText()+builder.toString());
        System.out.print(frame[0]+" | "+frame[0]+frame[1]+" | ");
}




if(frame[0]!=frame[1]&&frame[1]!=frame[2]&&fs>=4)
{
        builder.append(frame[0]+" | "+frame[0]+frame[1]+" |
        "+frame[0]+frame[1]+frame[2]+" | ");

        jlbframe.setText(jlbframe.getText()+builder.toString());
```

```
                    System.out.print(frame[0]+" | "+frame[0]+frame[1]+" |
                    "+frame[0]+frame[1]+frame[2]+" | ");
                }
                else
                {
                    builder.append(frame[0]+" | ");
                    System.out.print(frame[0]+" | ");
                }

                frame[fs]=checkInitFrame(frame,fs,sseq,1);  //will give the initial
                                                            //unique frame
                if(choice=="FIFO")
                {
                    opt.setText("FIFO Page Replacement Policy");
                    fifo(frame,sseq,fs,1,0,psn,fault);      //applying fifo algo
                }
                else if(choice=="LRU")
                {
                    opt.setText("LRU Page Replacement Policy");
                    lru_algo(frame,sseq,fs,0,0,psn,fault); //applying LRU algo
                }
                else
                {
                    if(choice=="Optimal")
                    {
                        opt.setText("OPTIMAL Page Replacement Policy");
                        optimal(frame,sseq,fs,0,0,psn,fault);  //applying
                                                                //OPTIMAL algo
                    }
                }
            }
        }
}
```

**#Output [initial window]**



No. Of String Select

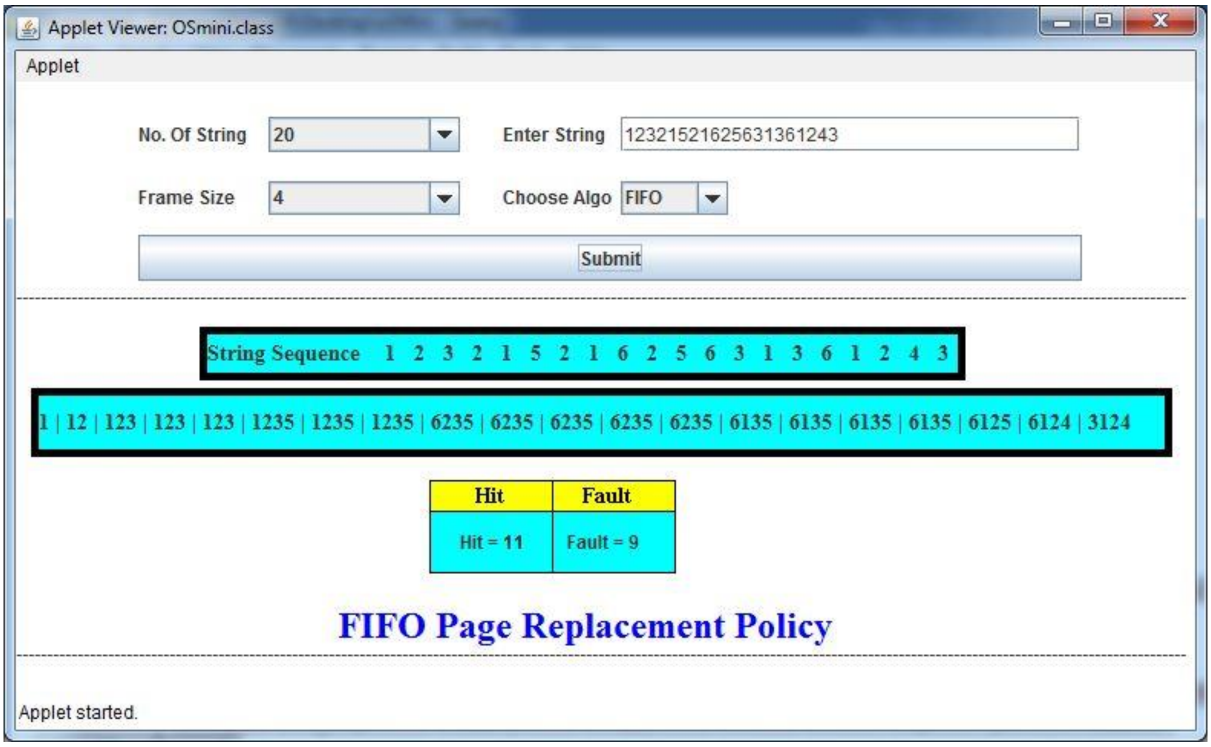
Enter String

Frame Size Select

Choose Algo Select

| Hit | Fault |
|---|---|
| | |

## #Output [FIFO Algo]



No. Of String 20

Enter String 12321521625631361243

Frame Size 4

Choose Algo FIFO

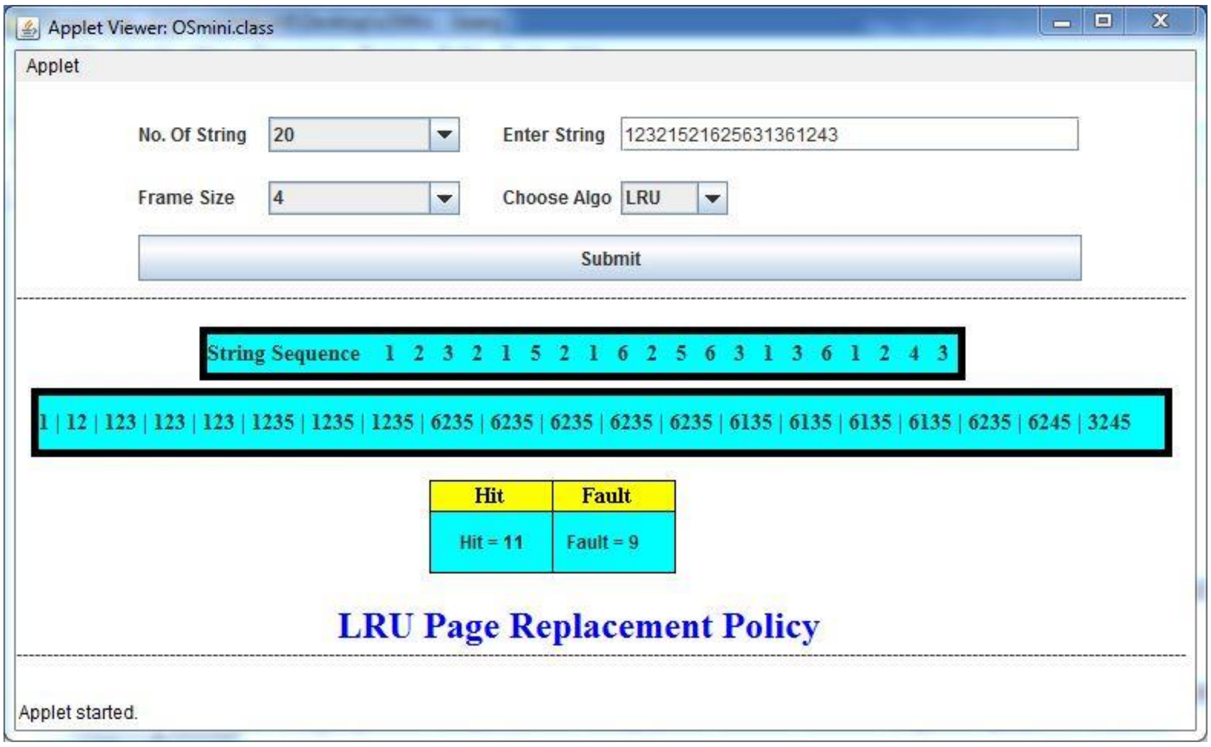
```
String Sequence   1  2  3  2  1  5  2  1  6  2  5  6  3  1  3  6  1  2  4  3
```

```
1 | 12 | 123 | 123 | 123 | 1235 | 1235 | 1235 | 6235 | 6235 | 6235 | 6235 | 6235 | 6135 | 6135 | 6135 | 6135 | 6125 | 6124 | 3124
```

| Hit | Fault |
|---|---|
| Hit = 11 | Fault = 9 |

FIFO Page Replacement Policy

## #Output [LRU Algo]


Applet Viewer: OSmini.class

Applet

No. Of String: 20
Enter String: 12321521625631361243
Frame Size: 4
Choose Algo: LRU

Submit

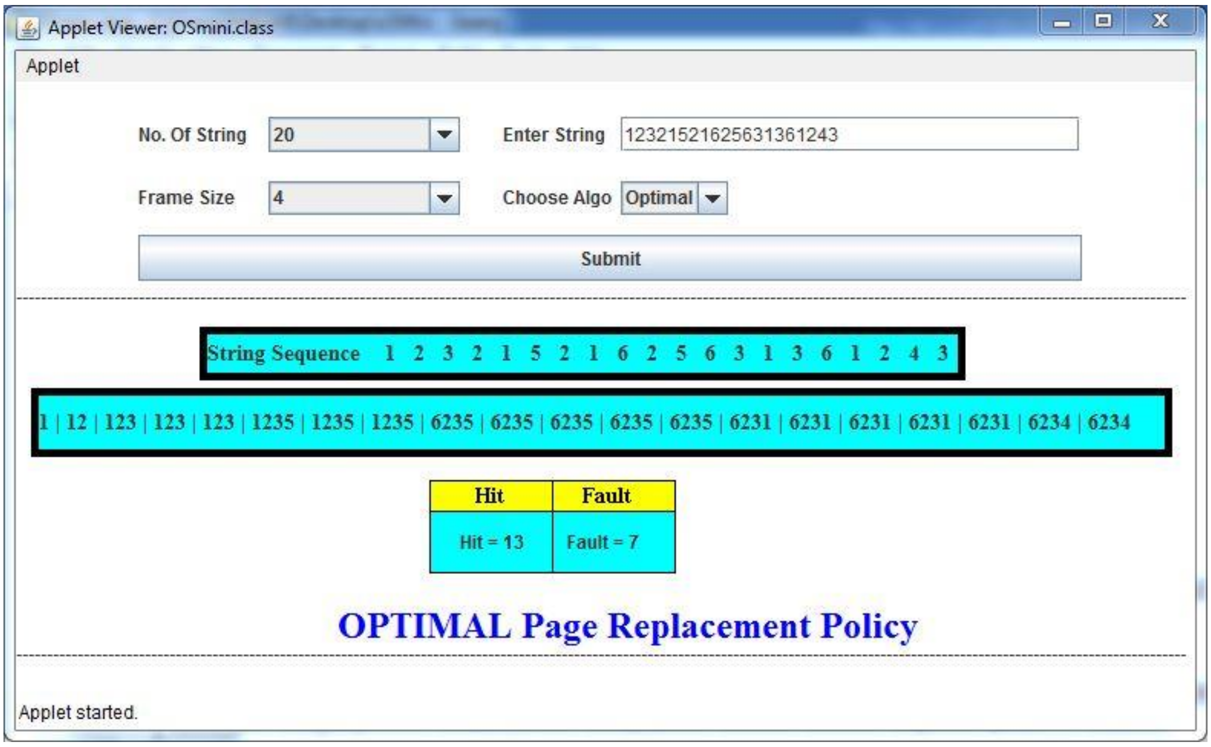
```
String Sequence  1  2  3  2  1  5  2  1  6  2  5  6  3  1  3  6  1  2  4  3
1 | 12 | 123 | 123 | 123 | 1235 | 1235 | 1235 | 6235 | 6235 | 6235 | 6235 | 6235 | 6135 | 6135 | 6135 | 6135 | 6235 | 6245 | 3245
```

| Hit | Fault |
|---|---|
| Hit = 11 | Fault = 9 |

LRU Page Replacement Policy

Applet started.

## #Output [Optimal Algo]


Applet Viewer: OSmini.class

Applet

No. Of String: 20
Enter String: 12321521625631361243
Frame Size: 4
Choose Algo: Optimal

Submit

```
String Sequence  1  2  3  2  1  5  2  1  6  2  5  6  3  1  3  6  1  2  4  3
1 | 12 | 123 | 123 | 123 | 1235 | 1235 | 1235 | 6235 | 6235 | 6235 | 6235 | 6235 | 6231 | 6231 | 6231 | 6231 | 6231 | 6234 | 6234
```

| Hit | Fault |
|---|---|
| Hit = 13 | Fault = 7 |

OPTIMAL Page Replacement Policy

Applet started.