Title: Memory Management-I

Experiment No: 9 a

Experiment Name: Memory Management using Worst Fit.

Algorithm:

1- Input memory blocks and processes with sizes.
2- Initialize all memory blocks as free.
3- Start by picking each process and find the minimum block size that can be assigned to current process i.e., find min(bockSize[1], blockSize[2],.....blockSize[n]) > processSize[current], if found then assign it to the current process.
4- If not then leave that process and keep checking the further processes.

Source Code:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int max=25
 int frag[max],b[max],f[max],i,j,nb,nf,temp,highest=0;
 static int bf[max],ff[max];
 clrscr();

 printf("\n\tMemory Management Scheme - Worst Fit");
 printf("\nEnter the number of blocks:");
 scanf("%d",&nb);
 printf("Enter the number of files:");
 scanf("%d",&nf);
 printf("\nEnter the size of the blocks:-\n");
 for(i=1;i<=nb;i++) {printf("Block %d:",i);scanf("%d",&b[i]);}
 printf("Enter the size of the files :-\n");
 for(i=1;i<=nf;i++) {printf("File %d:",i);scanf("%d",&f[i]);}

 for(i=1;i<=nf;i++)
```

```
{
 for(j=1;j<=nb;j++)
 {
  if(bf[j]!=1)    //if bf[j] is not allocated
  {
   temp=b[j]-f[i];
   if(temp>=0)
   if(highest<temp)
   {
    ff[i]=j;
    highest=temp;
   }
  }
 }
 frag[i]=highest;
 bf[ff[i]]=1;
 highest=0;
}
 printf("\nFile_no:\tFile_size :\tBlock_no:\tBlock_size:\tFragement");
 for(i=1;i<=nf;i++)
 printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d",i,f[i],ff[i],b[ff[i]],frag[i]);
 getch();
}
```

Output:

```
File  Edit  View  Search  Terminal  Help
sachin@sachin-VirtualBox:~/Desktop/programs$ cc worstfit.c
sachin@sachin-VirtualBox:~/Desktop/programs$ ./a.out

        Memory Management Scheme - Worst Fit
Enter the number of blocks:5
Enter the number of files:4

Enter the size of the blocks:-
Block 1:100
Block 2:500
Block 3:200
Block 4:300
Block 5:600
Enter the size of the files :-
File 1:212
File 2:417
File 3:12
File 4:100

File_no:        File_size :     Block_no:       Block_size:     Fragement
1               212             5               600             388

2               417             2               500             83

3               12              4               300             288

4               100             3               200             100
sachin@sachin-VirtualBox:~/Desktop/programs$
```

VIVA QUESTION:

- What is the FIRST FIT and BEST FIT algorithm
- Define Paging.
- Define Hole.
- Define External and Internal Fragmentation.

Title: Memory Management-II- Page Replacement

Experiment No: 9 b

Experiment Name: Memory Management using First In First Out Replacement Algorithm.

Algorithm:

1.Start the process

2. Declare the size with respect to page length

3. Check the need of replacement from the page to memory

4. Check the need of replacement from old page to new page in memory

5. Forma queue to hold all pages

6. Insert the page require memory into the queue

7. Check for bad replacement and page fault

8. Get the number of processes to be inserted

9. Display the values

10. Stop the process

Source Code:

```
#include<stdio.h>
int main()
{
int i,j,n,a[50],frame[10],no,k,avail,count=0;
        printf("\n ENTER THE NUMBER OF PAGES:\n");
scanf("%d",&n);
        printf("\n ENTER THE PAGE NUMBER :\n");
        for(i=1;i<=n;i++)
        scanf("%d",&a[i]);
        printf("\n ENTER THE NUMBER OF FRAMES :");
        scanf("%d",&no);
for(i=0;i<no;i++)
        frame[i]= -1;
                j=0;
                printf("\tref string\t page frames\n");
for(i=1;i<=n;i++)
                {
                        printf("%d\t\t",a[i]);
                        avail=0;
                        for(k=0;k<no;k++)
if(frame[k]==a[i])
                                avail=1;
                        if (avail==0)
                        {
                                frame[j]=a[i];
                                j=(j+1)%no;
                                count++;
                                for(k=0;k<no;k++)
                                printf("%d\t",frame[k]);
}
                        printf("\n");
}
                printf("Page Fault Is %d",count);
                return 0;
}
```

Output:

```
File  Edit  View  Search  Terminal  Help
sachin@sachin-VirtualBox:~/Desktop$ cc fi.c
sachin@sachin-VirtualBox:~/Desktop$ ./a.out

 ENTER THE NUMBER OF PAGES:
5

 ENTER THE PAGE NUMBER :
7
2
0
1
0

 ENTER THE NUMBER OF FRAMES :3
        ref string       page frames
7               7       -1      -1
2               7       2       -1
0               7       2       0
1               1       2       0
0
Page Fault Is 4
```

VIVA QUESTION:

- What is the criteria for the best page replacement algorithm?
- What is Belady's anomaly and why does it occur ? How would you avoid this ..?
- A memory page containing a heavily used variable that was initialized very early and is in constant use is removed, then the page replacement algorithm used is .
- If no frames are free, _____ page transfer(s) is/are required.
- For 3 page frames, the following is the reference string :
- 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1. How many page faults does the LRU page replacement algorithm produce ?