VISVESVARAYA TECHNOLOGICAL UNIVERSITY
JNANA SANGAMA, BELGAUM - 590014

SYSTEM SOFTWARE AND COMPILER DESIGN LAB REPORT

Submitted in the partial fulfillment for System Software and Compiler Design Course

Submitted By :

NAME:-MUSADDIQ SHARIFF USN:-1SI19CS082

BATCH : B1

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SIDDAGANGA INSTITUTE OF TECHNOLOGY, TUMKUR-572103

(An Autonomous Institute, Affiliated to Visvesvaraya Technological University, Belgaum, Recognized by AICTE and Accredited by NBA, New Delhi)

2021-2022

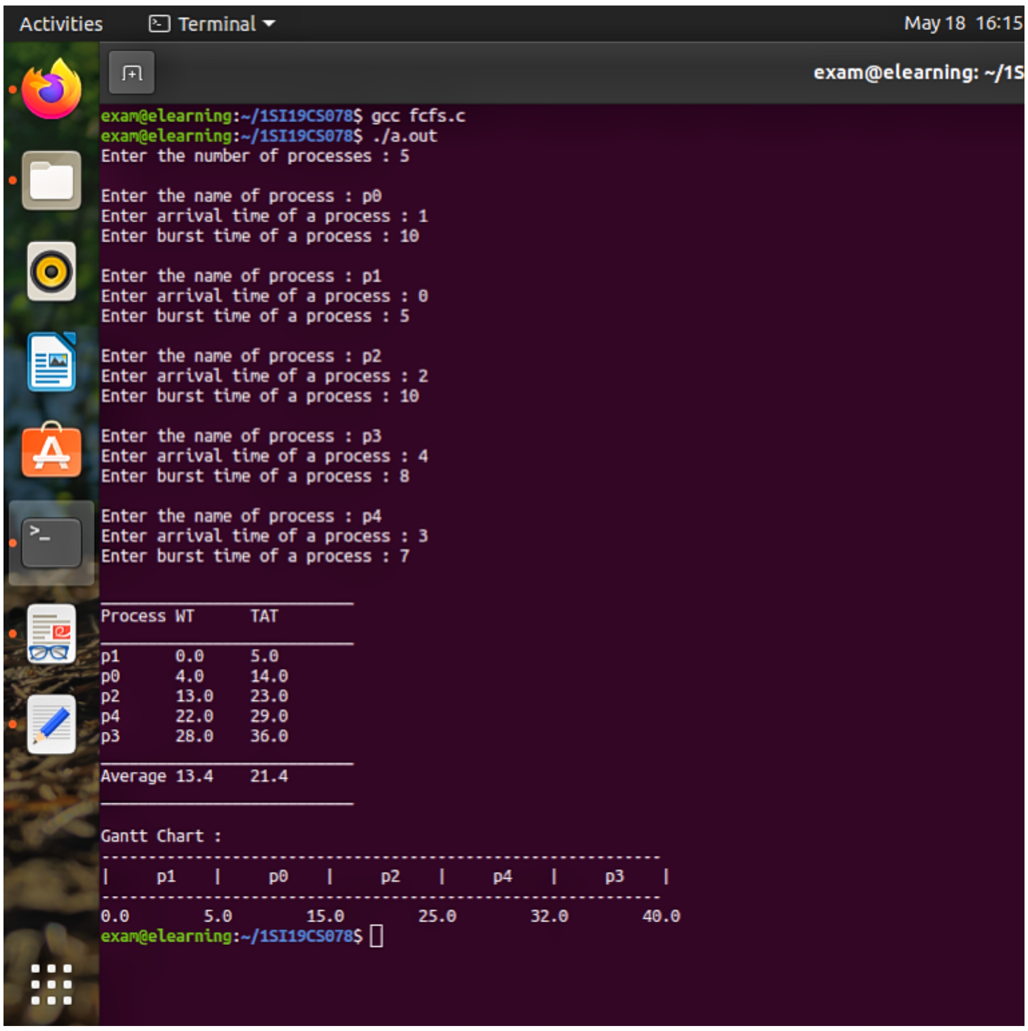
1. Given the list of processes, their burst times and arrival times, Write a C program to implement the FCFS CPU scheduling algorithm. Display the turnaround time & waiting time for each process. Also calculate the average turnaround time and average waiting time.

Program

```
#include<stdio.h>

#include<string.h>

typedef struct{

char name[10];

float a;

float b;

}FCFS;

int main(){

int n,i,j,y=0;

char z[20][25];

printf("Enter the number of processes : ");

scanf("%d",&n);

printf("\n");

FCFS s[n],temp;

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

printf("Enter the name of process : ");

scanf("%s",s[i].name);

printf("Enter arrival time of a process : ");

scanf("%f",&s[i].a);

printf("Enter burst time of a process : ");

scanf("%f",&s[i].b);

printf("\n");}

for(i=0;i<n-1;i++){

for(j=i;j<n-i-1;j++){

if(s[j].a>s[j+1].a){

temp=s[j];

s[j]=s[j+1];

s[j+1]=temp;}}}

float k=0,w[n],t[n],g[n+1];
```

```
int p=1;
g[0]=k;
float aw=0,tw=0;
for(i=0;i<n;i++){
w[i]=k-s[i].a;
if(w[i]<0){
strcpy(z[y],"ID");
y=y+1;
w[i]=0;
k=s[i].a;
g[p]=k;
p=p+1;}
strcpy(z[y],s[i].name);
y=y+1;
k=k+s[i].b;
g[p]=k;
p=p+1;
t[i]=k-s[i].a;
aw=aw+w[i];
tw=tw+t[i];}
printf("____________________________\n");
printf("Process\tWT\tTAT\n");
printf("____________________________\n");
for(i=0;i<n;i++){
printf("%s\t%.1f\t%.1f\n",s[i].name,w[i],t[i]);}
printf("____________________________\n");
printf("Average\t%.1f\t%.1f\n",aw/n,tw/n);
printf("____________________________\n\n");
printf("Gantt Chart :\n");
for(i=0;i<y;i++){
printf("-----------");}
printf("\n");
printf("|    %s",z[0]);
for(i=1;i<y;i++){
printf("    |    %s",z[i]);}
printf("    |\n");
for(i=0;i<y;i++){
printf("------------");}
printf("\n");
for(i=0;i<p;i++){
printf("%.1f        ",g[i]);}
printf("\n");
return 0;}
```

Output



```
exam@elearning:~/1SI19CS078$ gcc fcfs.c
exam@elearning:~/1SI19CS078$ ./a.out
Enter the number of processes : 5

Enter the name of process : p0
Enter arrival time of a process : 1
Enter burst time of a process : 10

Enter the name of process : p1
Enter arrival time of a process : 0
Enter burst time of a process : 5

Enter the name of process : p2
Enter arrival time of a process : 2
Enter burst time of a process : 10

Enter the name of process : p3
Enter arrival time of a process : 4
Enter burst time of a process : 8

Enter the name of process : p4
Enter arrival time of a process : 3
Enter burst time of a process : 7

__________________________
Process WT      TAT
__________________________
p1      0.0     5.0
p0      4.0     14.0
p2      13.0    23.0
p4      22.0    29.0
p3      28.0    36.0
__________________________
Average 13.4    21.4
__________________________

Gantt Chart :
------------------------------------------------------------
|    p1    |    p0    |    p2    |    p4    |    p3    |
------------------------------------------------------------
0.0        5.0        15.0       25.0       32.0       40.0
exam@elearning:~/1SI19CS078$ 
```

2. Given the list of processes, their burst times, priority and arrival times, Write a C program to implement the preemptive priority CPU scheduling algorithm. Display the turnaround time & waiting time for each process. Also calculate the average turnaround time and average waiting time.

Program

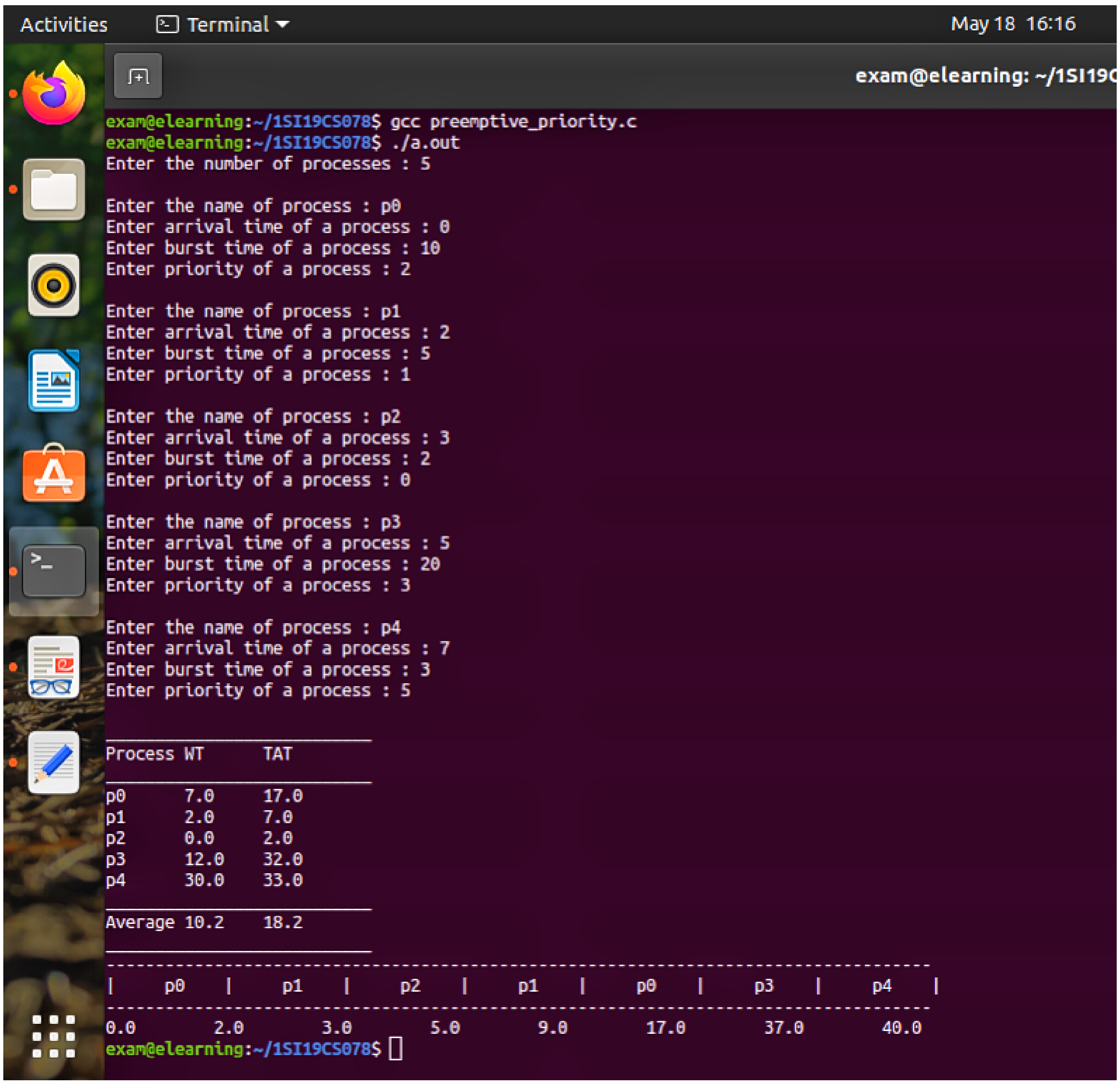
```
#include<stdio.h>

#include<string.h>

typedef struct{

char name[10];

float a,b, r;

int af,bf,p;

}PP;

int main(){

int n,i,j;

printf("Enter the number of processes : ");

scanf("%d",&n);

printf("\n");

PP s[n];

char temp[100];

char str[20][25];

int y=0;

float k[100];

int z=1;

k[0]=0;

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

printf("Enter the name of process : ");

scanf("%s",s[i].name);

printf("Enter arrival time of a process : ");

scanf("%f",&s[i].a);

printf("Enter burst time of a process : ");

scanf("%f",&s[i].b);

printf("Enter priority of a process : ");

scanf("%d",&s[i].p);
```

```
s[i].r=s[i].b;
s[i].af=0;
s[i].bf=0;
printf("\n");}
int g=0;
int m,flag;
while(1){
for(i=0;i<n;i++){
if(s[i].a<=g){
s[i].af=1;}}
int pr=9999;
for(i=0;i<n;i++){
if(s[i].af==1){
if(s[i].p<pr && s[i].bf==0){
pr=s[i].p;
m=i;}}}
if(g!=0){
if(strcmp(temp,s[m].name)!=0){
strcpy(str[y],temp);
y+=1;
strcpy(temp,s[m].name);
k[z]=g;
z+=1;}}
else{
strcpy(temp,s[m].name);}
if(s[m].bf==0){
s[m].b-=1;}
if(s[m].b==0){
s[m].bf=1;}
g+=1;
flag=0;
for(i=0;i<n;i++){
```

```
if(s[i].bf==0){
flag=1;
break;}}
if(flag==0){
break;}}
k[z]=g;
z+=1;
strcpy(str[y],temp);
y+=1;
float wt[n],tat[n];
float aw=0,tw=0;
for(i=0;i<n;i++){
for(j=0;j<y;j++){
if(strcmp(s[i].name,str[j])==0){
m=j;}}
tat[i]=k[m+1]-s[i].a;
wt[i]=tat[i]-s[i].r;
aw+=wt[i];
tw+=tat[i];}
printf("____________________________\n");
printf("Process\tWT\tTAT\n");
printf("____________________________\n");
for(i=0;i<n;i++){
printf("%s\t%.1f\t%.1f\n",s[i].name,wt[i],tat[i]);}
printf("____________________________\n");
printf("Average\t%.1f\t%.1f\n",aw/n,tw/n);
printf("____________________________\n");
for(i=0;i<y;i++){
printf("------------");}
printf("\n");
printf("|     %s",str[0]);
for(i=1;i<y;i++){
```

```
printf("    |     %s",str[i]);}
printf("    |\n");
for(i=0;i<y;i++){
printf("------------");}
printf("\n");
for(i=0;i<z;i++){
printf("%.1f        ",k[i]);}
printf("\n");
return 0;}
```

<u>Output</u>



```
exam@elearning:~/1SI19CS078$ gcc preemptive_priority.c
exam@elearning:~/1SI19CS078$ ./a.out
Enter the number of processes : 5

Enter the name of process : p0
Enter arrival time of a process : 0
Enter burst time of a process : 10
Enter priority of a process : 2

Enter the name of process : p1
Enter arrival time of a process : 2
Enter burst time of a process : 5
Enter priority of a process : 1

Enter the name of process : p2
Enter arrival time of a process : 3
Enter burst time of a process : 2
Enter priority of a process : 0

Enter the name of process : p3
Enter arrival time of a process : 5
Enter burst time of a process : 20
Enter priority of a process : 3

Enter the name of process : p4
Enter arrival time of a process : 7
Enter burst time of a process : 3
Enter priority of a process : 5

___________________________
Process WT      TAT
___________________________
p0      7.0     17.0
p1      2.0     7.0
p2      0.0     2.0
p3      12.0    32.0
p4      30.0    33.0
___________________________
Average 10.2    18.2
___________________________
------------------------------------------------------------------------------------
|     p0     |     p1     |     p2     |     p1     |     p0     |     p3     |     p4     |
------------------------------------------------------------------------------------
0.0         2.0         3.0         5.0         9.0         17.0        37.0        40.0
exam@elearning:~/1SI19CS078$ 
```

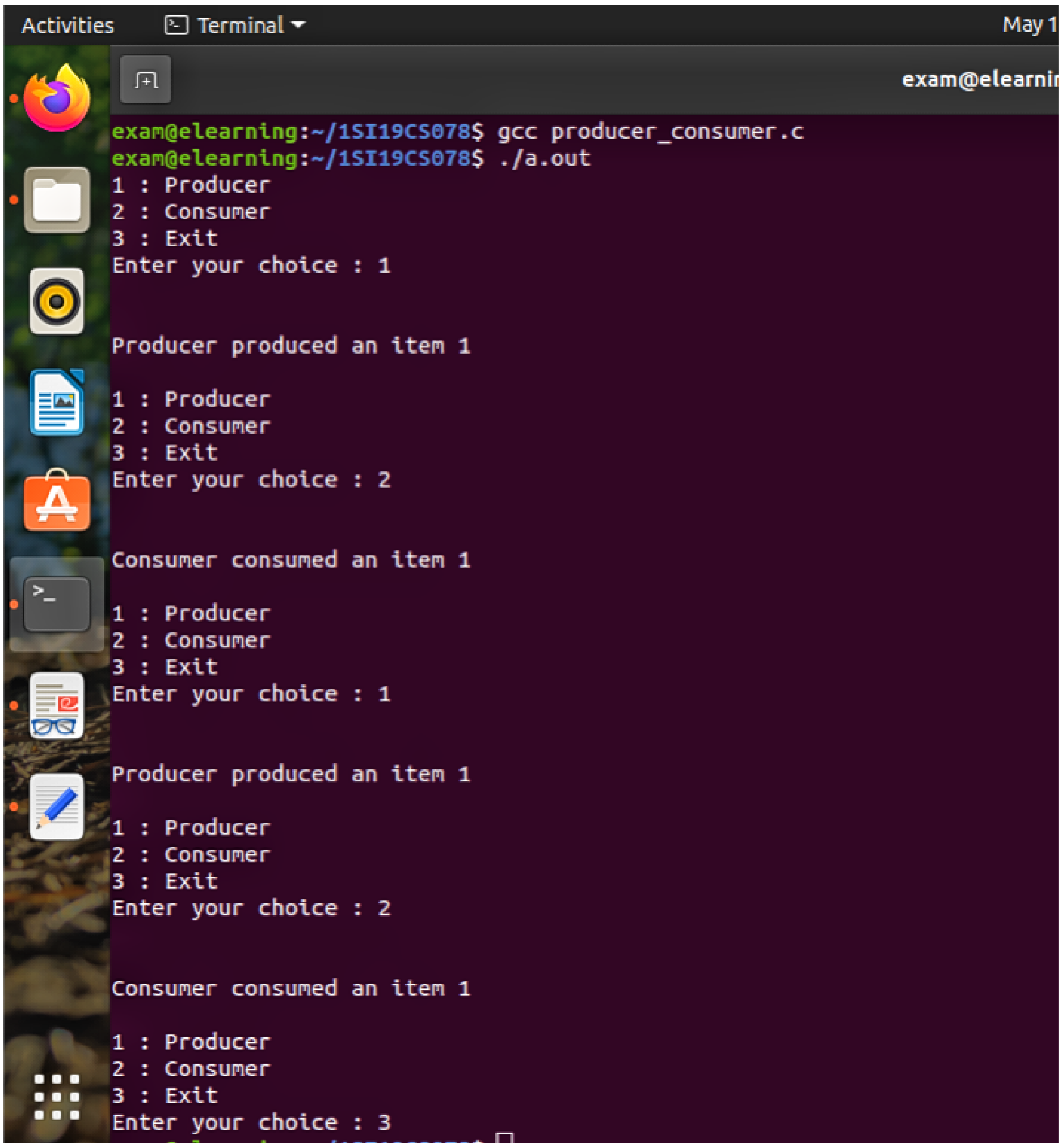
3. Write a C program to implement producer-consumer problem using semaphores.

Program

```
#include<stdio.h>
#include<stdlib.h>
int mutex=1,full=0,empty=2,x=0;
void producer(){
mutex-=1;
full+=1;
empty-=1;
x++;
mutex+=1;
printf("\n\nProducer produced an item %d\n\n",x);}
void consumer(){
mutex-=1;
full-=1;
empty+=1;
printf("\n\nConsumer consumed an item %d\n\n",x);
x--;
mutex+=1;}
int main(){
int choice;
while(1){
printf("1 : Producer\n2 : Consumer\n3 : Exit\nEnter your choice : ");
scanf("%d",&choice);
switch(choice){
case 1:if(mutex==1 && empty!=0){
producer();}
else{
printf("\n\nBuffer is full\n\n");}
break;
case 2:if(mutex==1 && full!=0){
```

```
consumer();}
else{
printf("\n\nBuffer is empty\n\n");}
break;
case 3:exit(0);}
}
return 0;
}
```

Output



```
exam@elearning:~/1SI19CS078$ gcc producer_consumer.c
exam@elearning:~/1SI19CS078$ ./a.out
1 : Producer
2 : Consumer
3 : Exit
Enter your choice : 1


Producer produced an item 1

1 : Producer
2 : Consumer
3 : Exit
Enter your choice : 2


Consumer consumed an item 1

1 : Producer
2 : Consumer
3 : Exit
Enter your choice : 1


Producer produced an item 1

1 : Producer
2 : Consumer
3 : Exit
Enter your choice : 2


Consumer consumed an item 1

1 : Producer
2 : Consumer
3 : Exit
Enter your choice : 3
```

4. Write a C program to implement Bankers algorithm for the purpose of deadlock avoidance.

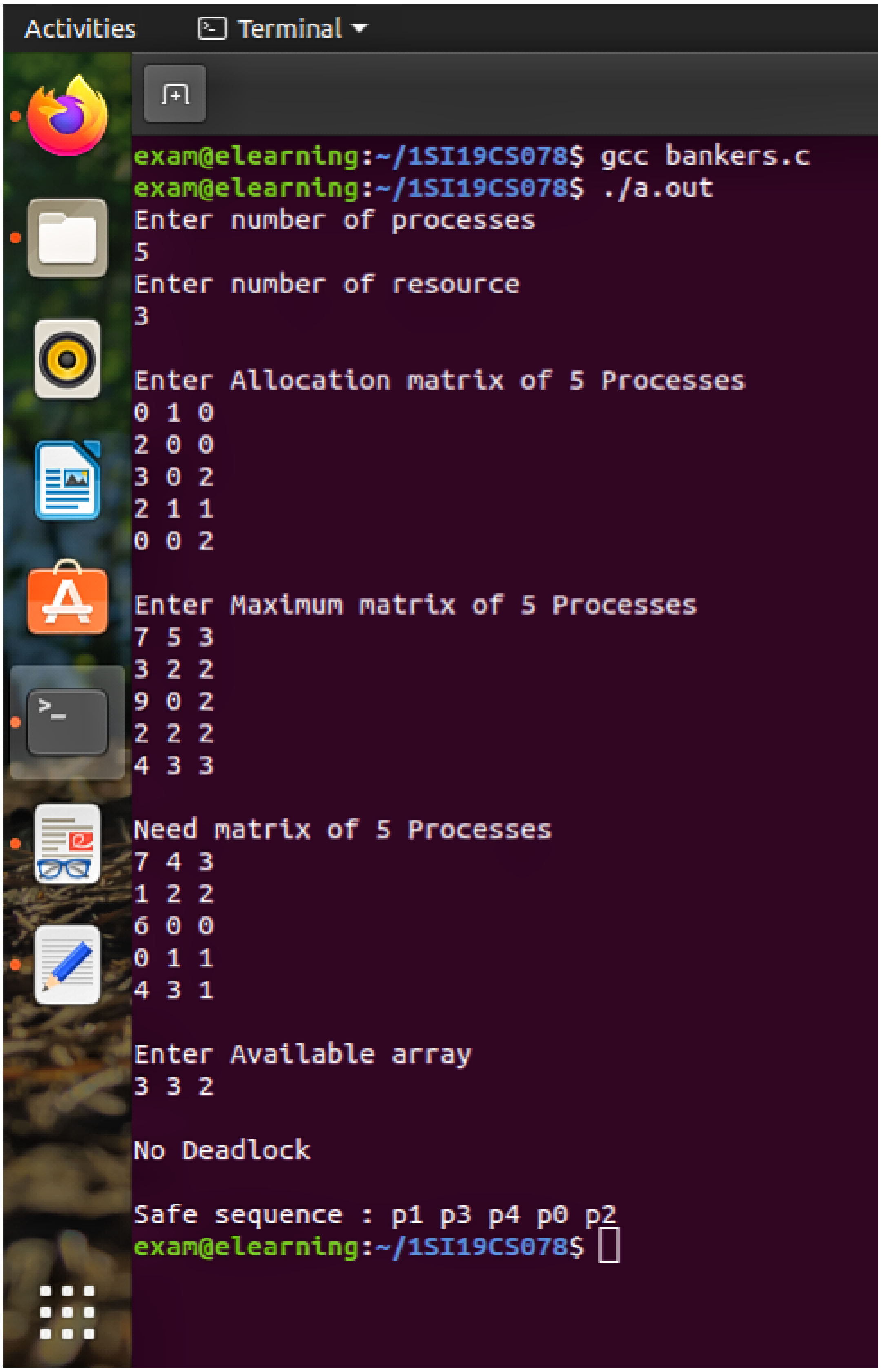
Program

```
#include<stdio.h>
#include<stdlib.h>
int main(){
int p,r;
printf("Enter number of processes\n");
scanf("%d",&p);
printf("Enter number of resource\n");
scanf("%d",&r);
int all[p][r],m[p][r],n[p][r],av[r],f[p],s[p],z=0;
int i,j,flag;
printf("\nEnter Allocation matrix of %d Processes\n",p);
for(i=0;i<p;i++){
for(j=0;j<r;j++){
scanf("%d",&all[i][j]);}}
printf("\nEnter Maximum matrix of %d Processes\n",p);
for(i=0;i<p;i++){
for(j=0;j<r;j++){
scanf("%d",&m[i][j]);
n[i][j]=m[i][j]-all[i][j];}}
printf("\nNeed matrix of %d Processes\n",p);
for(i=0;i<p;i++){
for(j=0;j<r;j++){
printf("%d ",n[i][j]);}
f[i]=1;
printf("\n");}
printf("\nEnter Available array\n");
for(j=0;j<r;j++){
scanf("%d",&av[j]);}
```

```
int count=0;
while(1){
flag=0;
for(i=0;i<p;i++){
int pflag=0;
if(f[i]==1){
for(j=0;j<r;j++){
if(n[i][j]>av[j]){
pflag=1;
break;}}
if(pflag==0){
for(j=0;j<r;j++){
av[j]+=all[i][j];
}
flag=1;
f[i]=0;
s[z]=i;
z+=1;
count+=1;}}}
if(flag==0)
break;
}
if(count==p){
printf("\nNo Deadlock \n\nSafe sequence : ");
for(j=0;j<p;j++){
printf("p%d ",s[j]);}
printf("\n");
}
else{
printf("\nDeadlock Detected,safe sequence does not exist\n");
}
return 0;
```

}

Output



```
exam@elearning:~/1SI19CS078$ gcc bankers.c
exam@elearning:~/1SI19CS078$ ./a.out
Enter number of processes
5
Enter number of resource
3

Enter Allocation matrix of 5 Processes
0 1 0
2 0 0
3 0 2
2 1 1
0 0 2

Enter Maximum matrix of 5 Processes
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3

Need matrix of 5 Processes
7 4 3
1 2 2
6 0 0
0 1 1
4 3 1

Enter Available array
3 3 2

No Deadlock

Safe sequence : p1 p3 p4 p0 p2
exam@elearning:~/1SI19CS078$
```

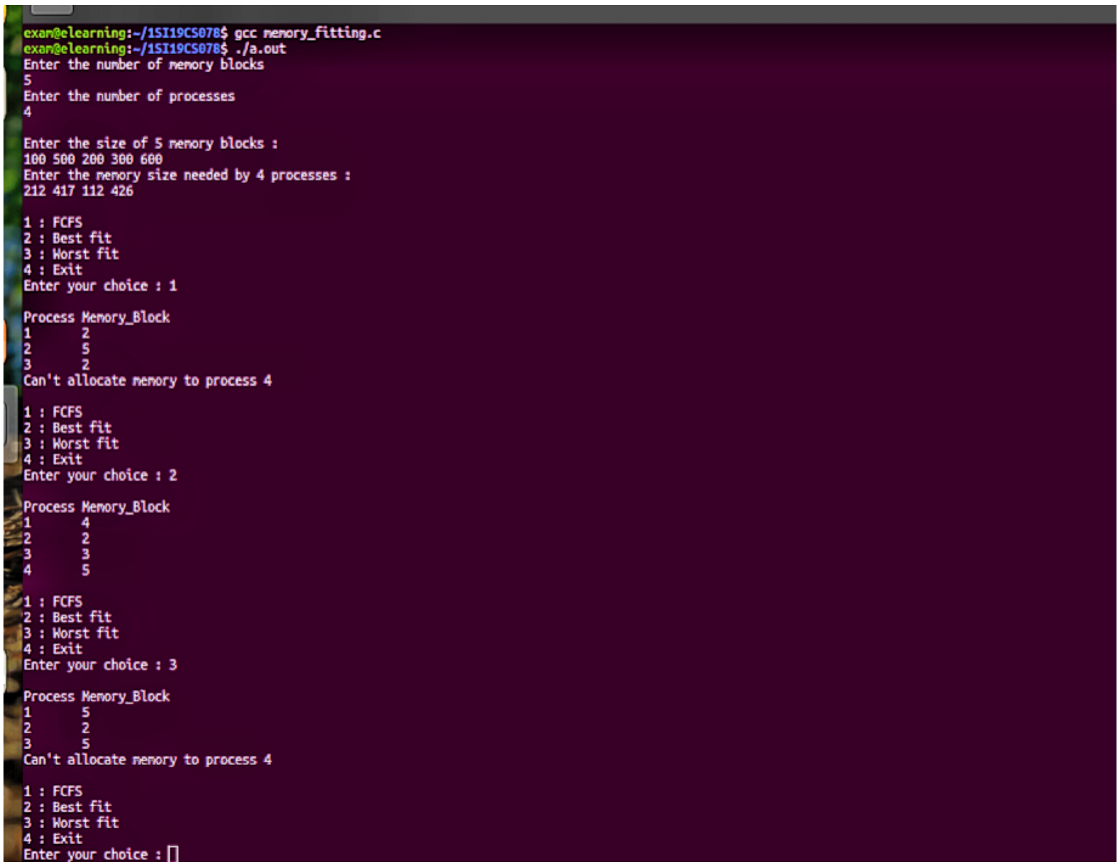
5. Write a C program to implement the following contiguous memory allocation techniques : a) Worst-fit b) Best-fit c) First-fit

Program

```
#include<stdio.h> #include<stdlib.h>
int process[100],memory[100],p,m;
void fcfs(int mem[]){
int flag;
printf("\nProcess\tMemory_Block\n");
for(int i=0;i<p;i++){
flag=0;
for(int j=0;j<m;j++){
if(process[i]<=mem[j]){
printf("%d\t%d\n",i+1,j+1);
mem[j]-=process[i];
flag=1;
break;}}
if(flag==0){
printf("Can't allocate memory to process %d\n",i+1);}}}
void bestfit(int mem[]){
int flag,d,min,index;
printf("\nProcess\tMemory_Block\n");
for(int i=0;i<p;i++){
flag=0; min=999999;
for(int j=0;j<m;j++){
if(process[i]<=mem[j]){
d=mem[j]-process[i];
if(d<min){
min=d; index=j;
flag=1;}}}
if(flag==0){
printf("Can't allocate memory to process %d\n",i+1);}
else{
printf("%d\t%d\n",i+1,index+1);
mem[index]-=process[i];}}}
void worstfit(int mem[])
{ int flag,d,max,index;
printf("\nProcess\tMemory_Block\n");
for(int i=0;i<p;i++){
flag=0; max=-99999;
for(int j=0;j<m;j++){
if(process[i]<=mem[j]){
d=mem[j]-process[i];
if(d>max){
max=d; index=j;
flag=1;}}}
if(flag==0){
printf("Can't allocate memory to process %d\n",i+1);}
else{
printf("%d\t%d\n",i+1,index+1);
mem[index]-=process[i];}}}
```

```c
int main(){
int i,j,temp[100];
printf("Enter the number of memory blocks\n");
scanf("%d",&m);
printf("Enter the number of processes\n");
scanf("%d",&p);
printf("\nEnter the size of %d memory blocks :\n",m);
for(i=0;i<m;i++){
scanf("%d",&memory[i]);}
printf("Enter the memory size needed by %d processes :\n",p);
for(i=0;i<p;i++){
scanf("%d",&process[i]);}
int choice;
while(1){
printf("\n1 : FCFS\n2 : Best fit\n3 : Worst fit\n4 : Exit\nEnter your choice : ");
scanf("%d",&choice);
for(i=0;i<m;i++)
temp[i]=memory[i];
switch(choice){
case 1:fcfs(temp); break;
case 2:bestfit(temp); break;
case 3:worstfit(temp); break;
case 4:exit(0);}}
return 0;}
```

<u>Output</u>



```
exam@elearning:~/1SI19CS078$ gcc memory_fitting.c
exam@elearning:~/1SI19CS078$ ./a.out
Enter the number of memory blocks
5
Enter the number of processes
4

Enter the size of 5 memory blocks :
100 500 200 300 600
Enter the memory size needed by 4 processes :
212 417 112 426

1 : FCFS
2 : Best fit
3 : Worst fit
4 : Exit
Enter your choice : 1

Process Memory_Block
1       2
2       5
3       2
Can't allocate memory to process 4

1 : FCFS
2 : Best fit
3 : Worst fit
4 : Exit
Enter your choice : 2

Process Memory_Block
1       4
2       2
3       3
4       5

1 : FCFS
2 : Best fit
3 : Worst fit
4 : Exit
Enter your choice : 3

Process Memory_Block
1       5
2       2
3       5
Can't allocate memory to process 4

1 : FCFS
2 : Best fit
3 : Worst fit
4 : Exit
Enter your choice : 
```

6. Write a C program to implement the following page replacement algorithms:

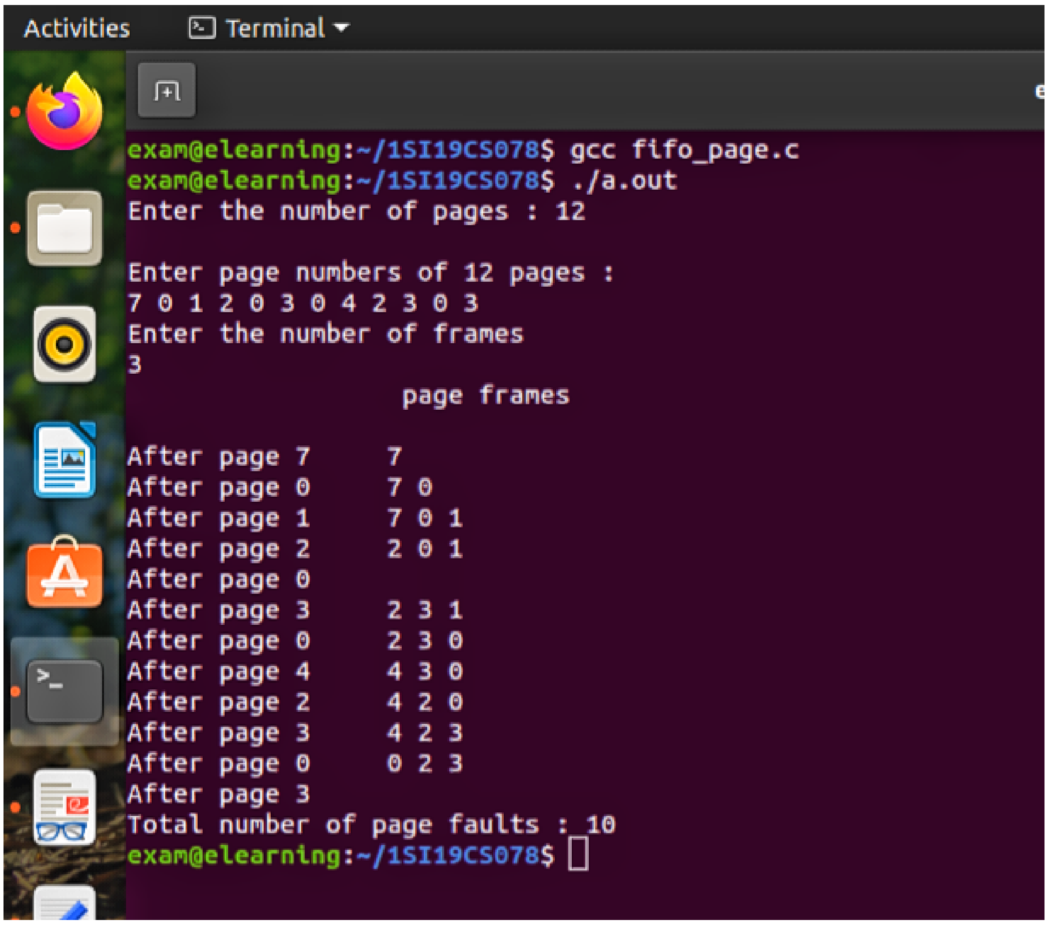
a)FIFO b) LRU c) LFU

FIFO Program

```
#include<stdio.h> #include<stdlib.h>
int main(){
int n,i,j,flag,count=0;
printf("Enter the number of pages : ");
scanf("%d",&n);
int a[n];
printf("\nEnter page numbers of %d pages : \n",n);
for(i=0;i<n;i++){
scanf("%d",&a[i]);}
printf("Enter the number of frames\n");
int f;
scanf("%d",&f);
int frame[f],z=0;
printf("                  page frames\n");
for(i=0;i<f;i++){
flag=0;
for(j=0;j<z;j++){
if(frame[j]==a[i]){
flag=1; break;}}
printf("\nAfter page %d     ",a[i]);
if(flag==0){
count+=1;
frame[z]=a[i];
z+=1;
for(j=0;j<z;j++){
printf("%d ",frame[j]);}
if(z==f) break;}}
int c=0;
for(i=f;i<n;i++){
```

```
flag=0;
for(j=0;j<f;j++){
if(frame[j]==a[i]){
flag=1;
break;}}
printf("\nAfter page %d     ",a[i]);
if(flag==0){
frame[c]=a[i];
count+=1;
c=(c+1)%f;
for(j=0;j<f;j++){
printf("%d ",frame[j]);}}}
printf("\nTotal number of page faults : %d\n",count);
return 0;}
```

Output

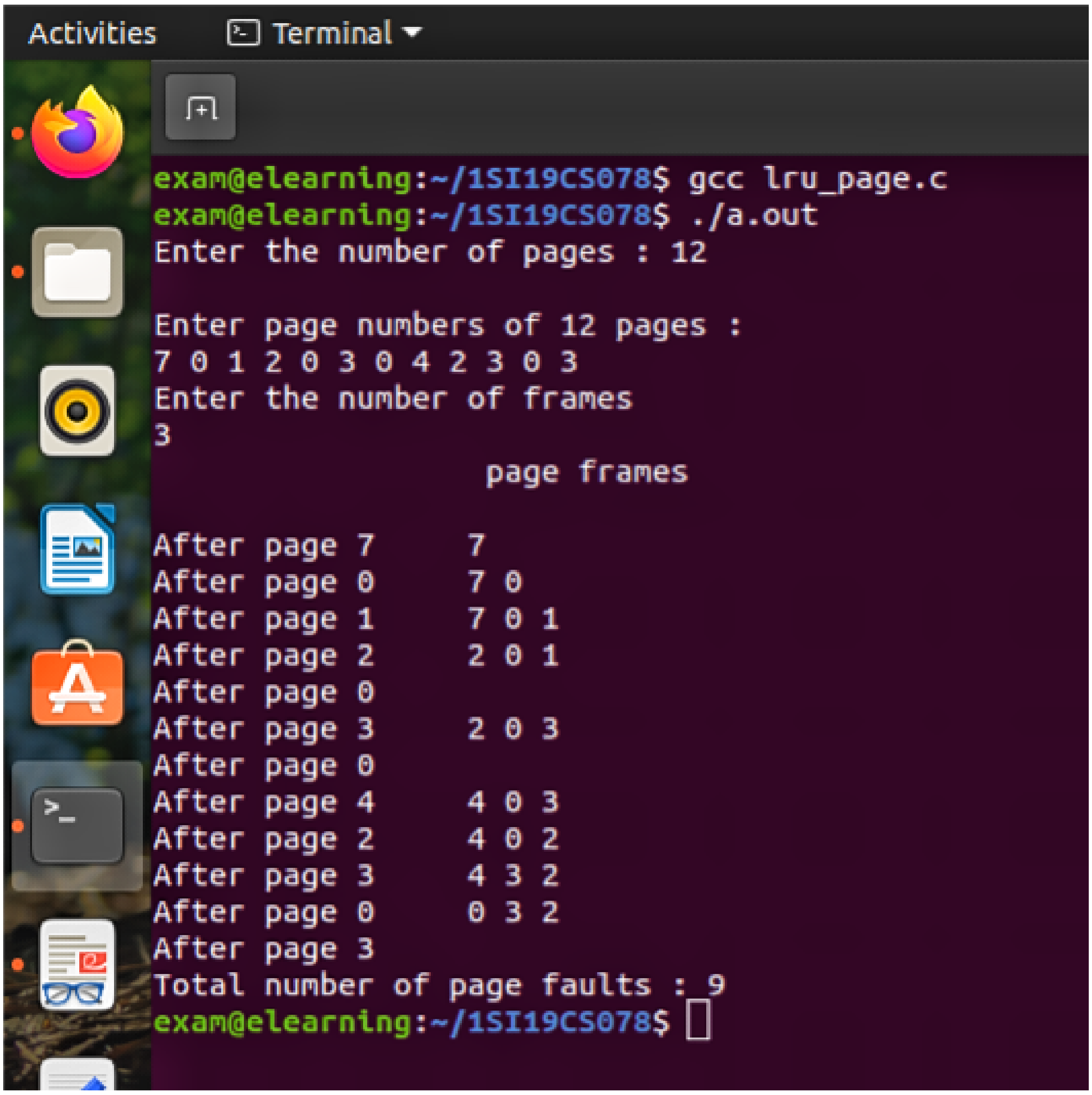


```
exam@elearning:~/1SI19CS078$ gcc fifo_page.c
exam@elearning:~/1SI19CS078$ ./a.out
Enter the number of pages : 12

Enter page numbers of 12 pages :
7 0 1 2 0 3 0 4 2 3 0 3
Enter the number of frames
3
                         page frames

After page 7     7
After page 0     7 0
After page 1     7 0 1
After page 2     2 0 1
After page 0
After page 3     2 3 1
After page 0     2 3 0
After page 4     4 3 0
After page 2     4 2 0
After page 3     4 2 3
After page 0     0 2 3
After page 3
Total number of page faults : 10
exam@elearning:~/1SI19CS078$
```

LRU Program

```
#include<stdio.h> #include<stdlib.h>
int main(){int n,i,j;
int k,l,in,flag,count=0;
printf("Enter the number of pages : ");
scanf("%d",&n);
int a[n];
printf("\nEnter page numbers of %d pages : \n",n);
for(i=0;i<n;i++){
scanf("%d",&a[i]);}
printf("Enter the number of frames\n");
int f;
scanf("%d",&f);
int frame[f];
printf("                page frames\n");
for(i=0;i<f;i++){
frame[i]=a[i]; count+=1;
printf("\nAfter page %d     ",a[i]);
for(j=0;j<i+1;j++){
printf("%d ",frame[j]);}}
for(i=f;i<n;i++){
flag=0;
for(j=0;j<f;j++){
if(frame[j]==a[i]){
flag=1; break;}}
printf("\nAfter page %d     ",a[i]);
if(flag==0){
count+=1; l=i;
for(j=0;j<f;j++){
for(k=i-1;k>=0;k--){
if(a[k]==frame[j]){
if(k<l){l=k;
in=j;}
break;}}}
frame[in]=a[i];
for(j=0;j<f;j++){
printf("%d ",frame[j]);}}}
printf("\nTotal number of page faults : %d\n",count);
return 0;}
```

Output

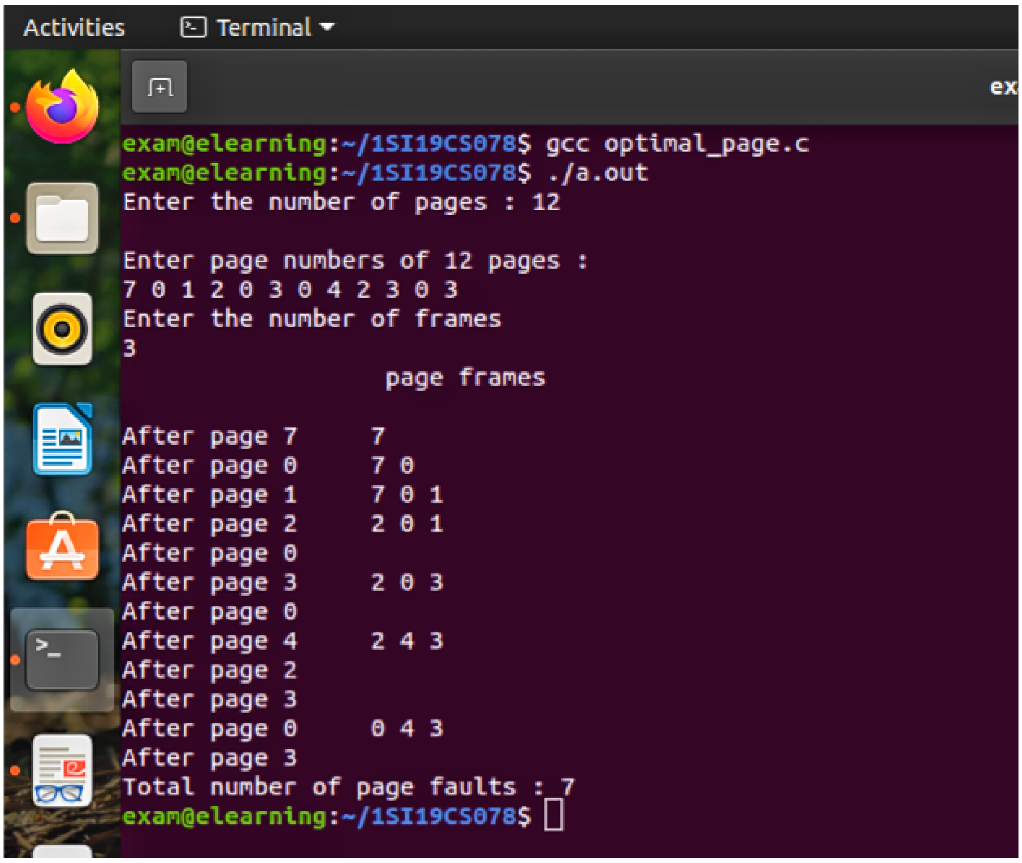


```
exam@elearning:~/1SI19CS078$ gcc lru_page.c
exam@elearning:~/1SI19CS078$ ./a.out
Enter the number of pages : 12

Enter page numbers of 12 pages :
7 0 1 2 0 3 0 4 2 3 0 3
Enter the number of frames
3
                  page frames

After page 7     7
After page 0     7 0
After page 1     7 0 1
After page 2     2 0 1
After page 0
After page 3     2 0 3
After page 0
After page 4     4 0 3
After page 2     4 0 2
After page 3     4 3 2
After page 0     0 3 2
After page 3
Total number of page faults : 9
exam@elearning:~/1SI19CS078$
```

LFU PROGRAM

```
#include<stdio.h>
#include<stdlib.h>
int main(){ int n,i,j;
int k,in,flag,count=0;
printf("Enter the number of pages : ");
scanf("%d",&n);
int a[n];
printf("\nEnter page numbers of %d pages : \n",n);
for(i=0;i<n;i++){
scanf("%d",&a[i]);}
printf("Enter the number of frames\n");
int f;
scanf("%d",&f);
int frame[f];
printf("                page frames\n");
for(i=0;i<f;i++){
frame[i]=a[i];
count+=1;
printf("\nAfter page %d     ",a[i]);
for(j=0;j<i+1;j++){
printf("%d ",frame[j]);}}
for(i=f;i<n;i++){
flag=0;
for(j=0;j<f;j++){
if(frame[j]==a[i]){
flag=1;
break;}}
printf("\nAfter page %d     ",a[i]);
if(flag==0){
count+=1;
int far=0;
for(j=0;j<f;j++){
int ff=0;
for(k=i;k<n;k++){
if(a[k]==frame[j]){ ff=1;
break;}}
if(ff=0){ in=j;
break;}
else{
if(k>far){
far=k; in=j;}}}}
frame[in]=a[i];
for(j=0;j<f;j++){
printf("%d ",frame[j]);}}}
printf("\nTotal number of page faults : %d\n",count);
return 0;}
```

Output



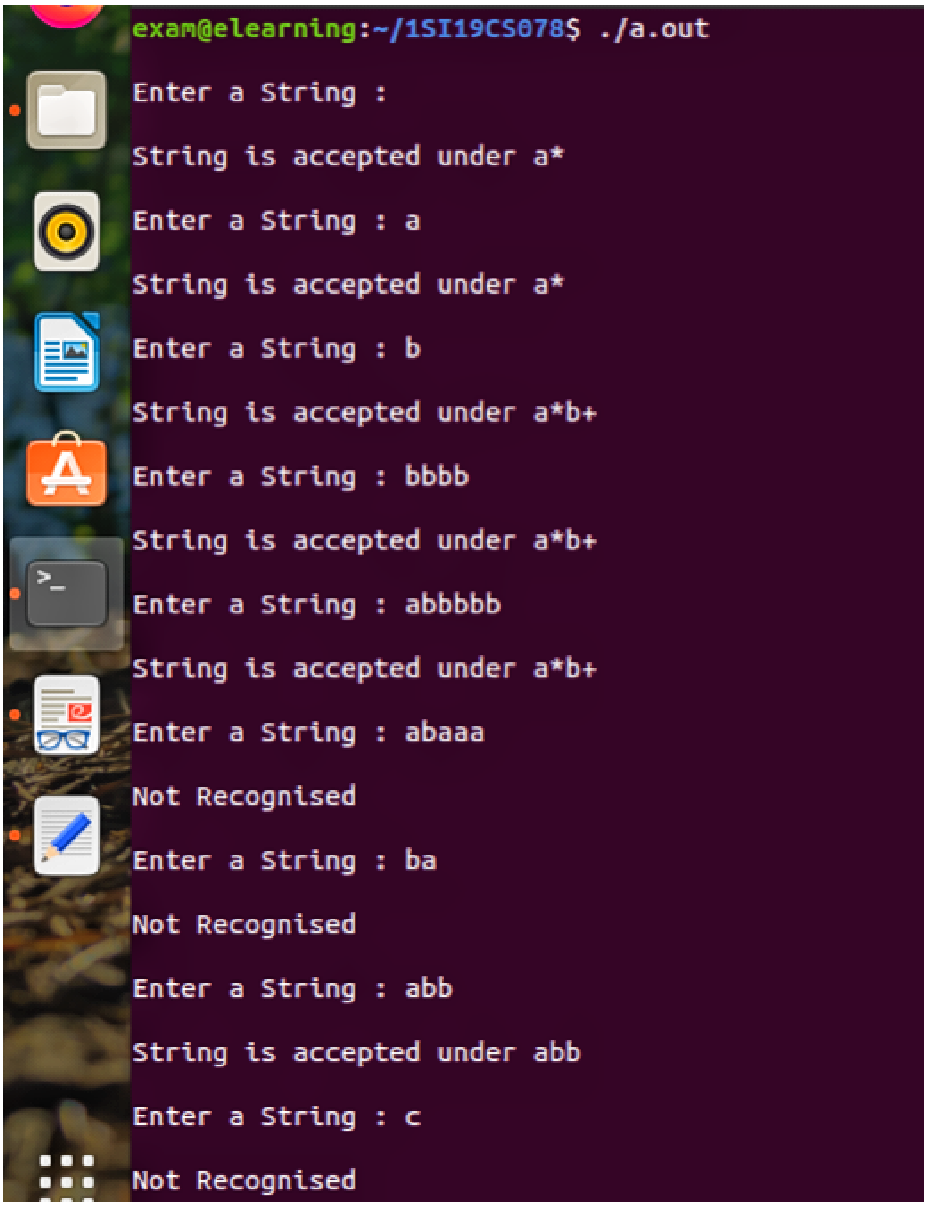
```
exam@elearning:~/1SI19CS078$ gcc optimal_page.c
exam@elearning:~/1SI19CS078$ ./a.out
Enter the number of pages : 12

Enter page numbers of 12 pages :
7 0 1 2 0 3 0 4 2 3 0 3
Enter the number of frames
3
                  page frames

After page 7     7
After page 0     7 0
After page 1     7 0 1
After page 2     2 0 1
After page 0
After page 3     2 0 3
After page 0
After page 4     2 4 3
After page 2
After page 3
After page 0     0 4 3
After page 3
Total number of page faults : 7
exam@elearning:~/1SI19CS078$
```

7. Write a C program to recognize strings under ‘a*’, ‘a*b+’, ‘abb’.

Program

```
#include<stdio.h> #include<stdlib.h> #include<ctype.h> #include<string.h>
int main(){ printf("\nEnter a String : ");
   char str[100];  int i;
   gets(str);
   int state=0;
   for(i=0;str[i]!='\0';i++){
          switch(state){
          case 0:if(str[i]=='a') state=1;
                    else if(str[i]=='b') state=2;
                    else state=7;
                    break;
          case 1:if(str[i]=='a') state=6;
                  else if(str[i]=='b') state=2;
                    else state=5;
                    break;
          case 2:if(str[i]=='b') state=3;
                    else state=5;
                    break;
          case 3:if(str[i]=='b') state=4;
                    else state=5;
                    break;
          case 4:if(str[i]=='b') state=4;
                    else state=5;
                    break;
          case 5:printf("\nNot Recognised\n");
                              return(main());
          case 6:if(str[i]=='a') state=6;
                    else if(str[i]=='b') state=4;
                    else state=5;
                    break;} }
   if(state==0 || state==1 || state==6)
   printf("\nString is accepted under a*\n");
   else if(state==3)
   printf("\nString is accepted under abb\n");
   else if(state==4 || state==2)
   printf("\nString is accepted under a*b+\n");
   else
   printf("\nNot Recognised\n");
   return(main()); }
```

Output



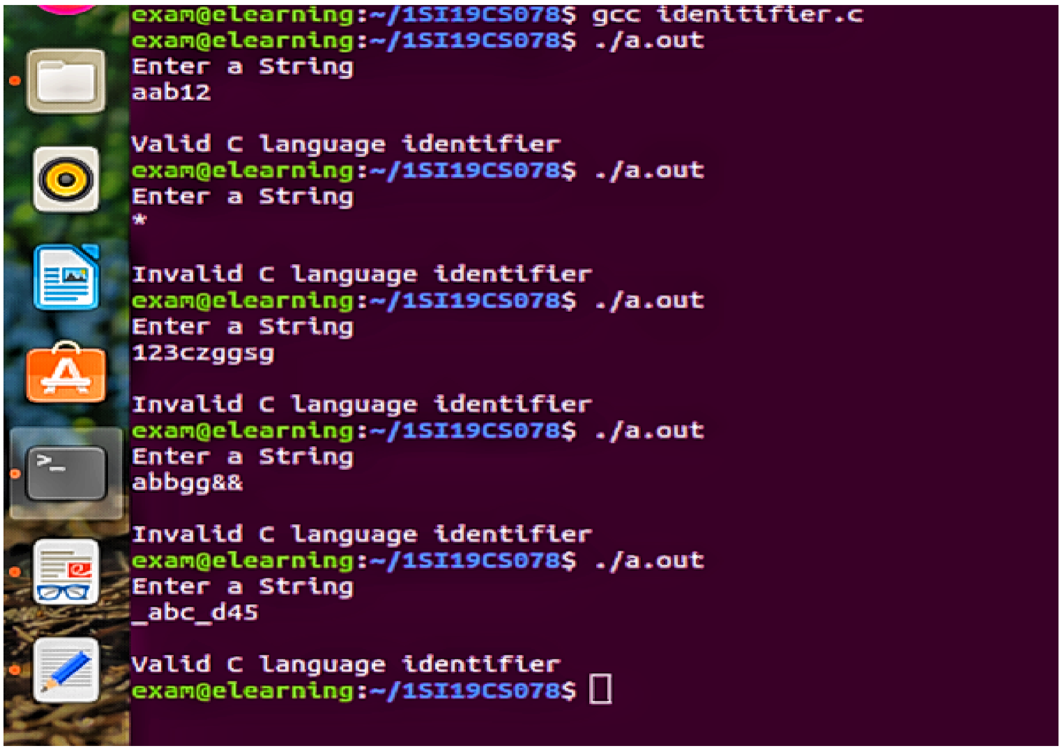
```
exam@elearning:~/1SI19CS078$ ./a.out
Enter a String :
String is accepted under a*
Enter a String : a
String is accepted under a*
Enter a String : b
String is accepted under a*b+
Enter a String : bbbb
String is accepted under a*b+
Enter a String : abbbbb
String is accepted under a*b+
Enter a String : abaaa
Not Recognised
Enter a String : ba
Not Recognised
Enter a String : abb
String is accepted under abb
Enter a String : c
Not Recognised
```

8. Write a C program to test whether a given identifier is valid or not.

Program

```
#include<stdio.h> #include<stdlib.h> #include<ctype.h>

int main()

{printf("Enter a String\n");

   char str[100];

   scanf("%[^\n]%*c",str);

   int i;

   if(str[0]!='_' && !isalpha(str[0])){

   printf("\nInvalid C language identifier\n");

   return 0;}

   for(i=1;str[i]!='\0';i++){

          if(str[i]!='_' && !isalnum(str[i])){

          printf("\nInvalid C language identifier\n");

          return 0;}}

   printf("\nValid C language identifier\n");

   return 0;}
```

Output



```
exam@elearning:~/1SI19CS078$ gcc idenitifier.c
exam@elearning:~/1SI19CS078$ ./a.out
Enter a String
aab12

Valid C language identifier
exam@elearning:~/1SI19CS078$ ./a.out
Enter a String
*

Invalid C language identifier
exam@elearning:~/1SI19CS078$ ./a.out
Enter a String
123czggsg

Invalid C language identifier
exam@elearning:~/1SI19CS078$ ./a.out
Enter a String
abbgg&&

Invalid C language identifier
exam@elearning:~/1SI19CS078$ ./a.out
Enter a String
_abc_d45

Valid C language identifier
exam@elearning:~/1SI19CS078$ 
```