**OPERATING SYSTEM LAB EXERCISE – COMS**

**5. C Program for First In First Serve Algorithm**

**#include<stdio.h>**

**#include<conio.h>**

**#include<dos.h>**

**void main()**

**{**

**int n,b[10],w[10],i,j,h;**

**int stime[10];**

**float avg=0;**

**clrscr();**

**printf("\n\tJOB SCHEDULING ALGORITHM[FCFS]");**

**printf("\n\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");**

**printf("\nEnter how many jobs:");**

**scanf("%d",&n);**

**printf("\nEnter burst time for corresponding job....\n");**

**for(i=0;i<n;i++)**

**{**

**printf("\nProcess %d:", i+1);**

**scanf("%d",&b[i]);**

**}**

**w[0]=0;**

**printf("\n process 1 waiting time is 0");**

**for(i=1;i<n;i++)**

**{**

**w[i]=b[i-1]+w[i-1];**

**printf("\n Process %d waiting time: %d",i+1,w[i]);**

**avg+=w[i];**

**}**

**printf("\n total waiting time:%f",avg);**

**printf("\n\n the average waiting time is:%f\n",avg/n);**

**printf("\nGaunt Chart\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n\t");**

**h=22;**

**for(i=0;i<n;i++)**

**{**

**printf("%d",b[i]);**

**for(j=1;j<=b[i];j++)**

**printf("%c",h);**

**}**

**printf("\n\t");**

**for(i=0;i<n;i++)**

**{**

**printf("%d",w[i]);**

**for(j=1;j<=w[i];j++)**

**printf("%c",h);**

**delay(1000);**

**}**

**getch();**

**}**

|  |
| --- |
| **SAMPLE INPUT AND OUTPUT:**  Enter howmany jobs: 5  Enter burst time for corresponding job....  Process 1:4  Process 2:2  Process 3:8  Process 4:2  Process 5:9  process 1 waiting time is 0 Process 2 waiting time: 4 Process 3 waiting time: 6 Process 4 waiting time: 14 Process 5 waiting time: 16 total waiting time:40.000000  the average waiting time is:8.000000  Gaunt Chart \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*          4▬▬▬▬2▬▬8▬▬▬▬▬▬▬▬2▬▬9▬▬▬▬▬▬▬▬▬         04▬▬▬▬6▬▬▬▬▬▬14▬▬▬▬▬▬▬▬▬▬▬▬▬▬16▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬ |

**6. C Program for Shortest Job First Algorithm**

**#include<stdio.h>**

**#include<conio.h>**

**#include<dos.h>**

**void main()**

**{**

**int n,b[10],w[10],i,j,h,t,tt;**

**int stime[10],a[10];**

**float avg=0;**

**clrscr();**

**printf("\n\tJOB SCHEDULING ALGORITHM[SJF]");**

**printf("\n\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");**

**printf("\nEnter howmany jobs:");**

**scanf("%d",&n);**

**printf("\nEnter burst time for corresponding job....\n");**

**for(i=1;i<=n;i++)**

**{**

**printf("\nProcess %d:",i);**

**scanf("%d",&b[i]); a[i]=i;**

**}**

**for(i=1;i<=n;i++)**

**for(j=i;j<=n;j++)**

**if(b[i]>b[j])**

**{**

**t=b[i]; tt=a[i];**

**b[i]=b[j]; a[i]=a[j];**

**b[j]=t; a[j]=tt;**

**}**

**w[1]=0;**

**printf("\nprocess %d waiting time is 0",a[1]);**

**for(i=2;i<=n;i++)**

**{**

**w[i]=b[i-1]+w[i-1];**

**printf("\nProcess %d waiting time: %d",a[i],w[i]);**

**avg+=w[i];**

**}**

**printf("\ntotal waiting time:%f",avg);**

**printf("\n\nthe average waiting time is:%f\n",avg/n);**

**printf("\nGaunt Chart\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n\t");**

**h=22;**

**for(i=1;i<=n;i++)**

**{**

**printf("%d",b[i]);**

**for(j=1;j<=b[i];j++)**

**printf("%c",h);**

**}**

**printf("\n\t");**

**for(i=1;i<=n;i++)**

**{**

**printf("%d",w[i]);**

**for(j=1;j<=w[i];j++)**

**printf("%c",h);**

**delay(1000);**

**}**

**getch();**

**}**

**#include<stdio.h>**

**#include<conio.h>**

**int z[10],b[10],n,m[50],r,q,e=0,avg=0,i,j;**

**float f;**

**main()**

**{**

**clrscr();**

**printf("\n\tJOB SCHEDULING ALGORITHM[RR]");**

**printf("\n\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");**

**printf("\nEnter how many jobs:");**

**scanf("%d",&n);**

**printf("\nEnter burst time for corresponding job...\n");**

**for(i=1;i<=n;i++)**

**{**

**printf("\nProcess %d: ",i);**

**scanf("%d",&b[i]); z[i]=b[i];**

**}**

**printf("\nENTER THE TIME SLICE VALUE:");**

**scanf("%d",&q);**

**rr();**

**average();**

**getch();**

**return 0;**

**}**

**rr()**

**{**

**int max=0;**

**max=b[1];**

**for(j=1;j<=n;j++)**

**if(max<=b[j])**

**max=b[j];**

**if((max%q)==0)**

**r=(max/q);**

**else**

**r=(max/q)+1;**

**for(i=1;i<=r;i++)**

**{**

**printf("\nround %d",i);**

**for(j=1;j<=n;j++)**

**{**

**if(b[j]>0)**

**{**

**b[j]=b[j]-q;**

**if(b[j]<=0)**

**{**

**b[j]=0;**

**printf("\nprocess %d is completed",j);**

**}**

**else**

**printf("\nprocess %d remaining time is %d",j,b[j]);**

**}**

**}**

**delay(1000);**

**}**

**return 0;**

**}**

**average()**

**{**

**for(i=1;i<=n;i++)**

**{**

**e=0;**

**for(j=1;j<=r;j++)**

**{**

**if(z[i]!=0)**

**{**

**if(z[i]>=q)**

**{**

**m[i+e]=q; z[i]-=q;**

**}**

**else**

**{**

**m[i+e]=z[i]; z[i]=0;**

**}**

**}**

**else**

**m[i+e]=0;**

**e=e+n;**

**}**

**}**

**for(i=2;i<=n;i++)**

**for(j=1;j<=i-1;j++)**

**avg=avg+m[j];**

**for(i=n+1;i<=r\*n;i++)**

**{**

**if(m[i]!=0)**

**{**

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

**avg=m[j]+avg;**

**}**

**}**

**f=avg/n;**

**printf("\nTOTAL WATING:%d",avg);**

**printf("\n\nAVERAGE WAITING TIME:%f\n",f);**

**for(i=1;i<=r\*n;i++)**

**{ if(m[i]!=0)**

**if(i%n==0){**

**printf("P%d",(i%n)+(n)); }**

**else**

**printf("P%d",(i%n));**

**for(j=1;j<=m[i];j++)**

**printf("%c",22);**

**}**

**printf("\n");**

**getch();**

**return 0;**

**}**

**6. C Program for Priority Scheduling Algorithm**

**//priority scheduling**

**#include<stdio.h>**

**#include<conio.h>**

**#include<dos.h>**

**void main()**

**{**

**int n,b[10],w[10],i,j,h,t,tt;**

**int stime[10],a[10],p[10];**

**float avg=0;**

**clrscr();**

**printf("\n\tPRIORITY SCHEDULING ALGORITHM");**

**printf("\n\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");**

**printf("\nEnter howmany jobs:");**

**scanf("%d",&n);**

**printf("\nEnter burst time & priority for corresponding job....\n");**

**for(i=1;i<=n;i++)**

**{**

**printf("\nProcess %d:",i);**

**scanf("%d %d",&b[i],&p[i]); a[i]=i;**

**}**

**for(i=1;i<=n;i++)**

**for(j=i;j<=n;j++)**

**if(p[i]<p[j])**

**{**

**t=b[i]; tt=a[i];**

**b[i]=b[j]; a[i]=a[j];**

**b[j]=t; a[j]=tt;**

**}**

**w[1]=0;**

**printf("\nprocess %d waiting time is 0",a[1]);**

**for(i=2;i<=n;i++)**

**{**

**w[i]=b[i-1]+w[i-1];**

**printf("\nProcess %d waiting time: %d",a[i],w[i]);**

**avg+=w[i];**

**}**

**printf("\ntotal waiting time:%f",avg);**

**printf("\n\nthe average waiting time is:%f\n",avg/n);**

**printf("\nGaunt Chart\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n\t");**

**h=22;**

**for(i=1;i<=n;i++)**

**{**

**printf("%d",b[i]);**

**for(j=1;j<=b[i];j++)**

**printf("%c",h);**

**}**

**printf("\n\t");**

**for(i=1;i<=n;i++)**

**{**

**printf("%d",w[i]);**

**for(j=1;j<=w[i];j++)**

**printf("%c",h);**

**delay(1000);**

**}**

**getch();**

**}**

**7. C Program For Round Robin Scheduling Method**

**#include<stdio.h>**

**#include<conio.h>**

**int z[10],b[10],n,m[50],r,q,e=0,avg=0,i,j;**

**float f;**

**main()**

**{**

**clrscr();**

**printf("\n\tJOB SCHEDULING ALGORITHM[RR]");**

**printf("\n\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");**

**printf("\nEnter how many jobs:");**

**scanf("%d",&n);**

**printf("\nEnter burst time for corresponding job...\n");**

**for(i=1;i<=n;i++)**

**{**

**printf("\nProcess %d: ",i);**

**scanf("%d",&b[i]); z[i]=b[i];**

**}**

**printf("\nENTER THE TIME SLICE VALUE:");**

**scanf("%d",&q);**

**rr();**

**average();**

**getch();**

**return 0;**

**}**

**rr()**

**{**

**int max=0;**

**max=b[1];**

**for(j=1;j<=n;j++)**

**if(max<=b[j])**

**max=b[j];**

**if((max%q)==0)**

**r=(max/q);**

**else**

**r=(max/q)+1;**

**for(i=1;i<=r;i++)**

**{**

**printf("\nround %d",i);**

**for(j=1;j<=n;j++)**

**{**

**if(b[j]>0)**

**{**

**b[j]=b[j]-q;**

**if(b[j]<=0)**

**{**

**b[j]=0;**

**printf("\nprocess %d is completed",j);**

**}**

**else**

**printf("\nprocess %d remaining time is %d",j,b[j]);**

**}**

**}**

**delay(1000);**

**}**

**return 0;**

**}**

**average()**

**{**

**for(i=1;i<=n;i++)**

**{**

**e=0;**

**for(j=1;j<=r;j++)**

**{**

**if(z[i]!=0)**

**{**

**if(z[i]>=q)**

**{**

**m[i+e]=q; z[i]-=q;**

**}**

**else**

**{**

**m[i+e]=z[i]; z[i]=0;**

**}**

**}**

**else**

**m[i+e]=0;**

**e=e+n;**

**}**

**}**

**for(i=2;i<=n;i++)**

**for(j=1;j<=i-1;j++)**

**avg=avg+m[j];**

**for(i=n+1;i<=r\*n;i++)**

**{**

**if(m[i]!=0)**

**{**

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

**avg=m[j]+avg;**

**}**

**}**

**f=avg/n;**

**printf("\nTOTAL WATING:%d",avg);**

**printf("\n\nAVERAGE WAITING TIME:%f\n",f);**

**for(i=1;i<=r\*n;i++)**

**{ if(m[i]!=0)**

**if(i%n==0){**

**printf("P%d",(i%n)+(n)); }**

**else**

**printf("P%d",(i%n));**

**for(j=1;j<=m[i];j++)**

**printf("%c",22);**

**}**

**printf("\n");**

**getch();**

**return 0;**

**}**

**SAMPLE INPUT AND OUTPUT:**

**Enter how many jobs:4**

**Enter burst time for corresponding job...**

**Process 1: 7**

**Process 2: 5**

**Process 3: 4**

**Process 4: 2**

**ENTER THE TIME SLICE VALUE:2**

**round 1**

**process 1 remaining time is 5**

**process 2 remaining time is 3**

**process 3 remaining time is 2**

**process 4 is completed**

**round 2**

**process 1 remaining time is 3**

**process 2 remaining time is 1**

**process 3 is completed**

**round 3**

**process 1 remaining time is 1**

**process 2 is completed**

**round 4**

**process 1 is completed**

**TOTAL WATING:39**

**AVERAGE WAITING TIME:9.000000**

**P1▬▬P2▬▬P3▬▬P4▬▬P1▬▬P2▬▬P3▬▬P1▬▬P2▬P1▬**

**8. C Program For Implementation Of System Calls**

**#include<stdio.h>**

**#include<conio.h>**

**#include<dir.h>**

**#include<dos.h>**

**void main()**

**{**

**int ch;**

**clrscr();**

**do**

**{**

**printf("\n\t\t\tMAIN MENU\n\t\t-------------------------\n");**

**printf("1.To Display List of Files\n");**

**printf("2.To Create New Directory\n");**

**printf("3.To Change the Working Directory\n");**

**printf("4.Exit\n");**

**printf("Enter the Number:");**

**scanf("%d",&ch);**

**switch(ch)**

**{**

**case 1:**

**list\_file();**

**break;**

**case 2:**

**directory();**

**break;**

**case 3:**

**change\_dir();**

**break;**

**case 4:**

**exit(0);**

**} }while(ch<=4);**

**}**

**int list\_file()**

**{**

**int l;**

**char e[]="\*.\*";**

**clrscr();**

**printf("\t\tLIST FILE DETAIL\n\t--------------------------------------\n");**

**printf("1.List All Files\n2.List of Extention Files\n3.List of Name Wise\n");**

**scanf("%d",&l);**

**switch(l)**

**{**

**case 1:**

**printf("List of All(\*.\*) Files\n");**

**subfun(e);**

**break;**

**case 2:**

**printf("Enter the Extention:");**

**scanf("%s",&e);**

**subfun(e);**

**break;**

**case 3:**

**printf("Enter the Name wise(eg:moha\*.\*):");**

**scanf("%s",&e);**

**subfun(e);**

**break;**

**}**

**return 0;**

**}**

**int directory()**

**{**

**struct ffblk ffblk;**

**unsigned attrib;**

**int d;**

**char name[10],buffer[MAXPATH];**

**printf("Enter the Directory name:");**

**scanf("%s",&name);**

**getcwd(buffer,MAXPATH);**

**printf("Current directory:%s\n",buffer);**

**if(\_dos\_getfileattr(name,&attrib)==0)**

**{**

**printf("%s has already available",name);**

**return 0;**

**}**

**else**

**{**

**mkdir(name);**

**printf("%s Directory Successfully Created\n",name);**

**}**

**return 0;**

**}**

**int change\_dir()**

**{**

**char buffer[MAXPATH];**

**int d,d1;**

**printf("\nCurrent Directory:%s\n",getcwd(buffer,MAXPATH));**

**printf("\t\tChange Directory\n\t\t-----------------\n");**

**printf("\n1.Step by Step Backward\n2.Goto Root Directory\n3.Forward Directory \nEnter the number:");**

**scanf("%d",&d);**

**switch(d)**

**{**

**case 1:**

**chdir("..");**

**break;**

**case 2:**

**chdir("\\");**

**break;**

**case 3:**

**printf("Please enter the Filename:");**

**scanf("%s",buffer);**

**chdir(buffer);**

**break;**

**}**

**printf("\nCurrent Directory:%s",getcwd(buffer,MAXPATH));**

**return 0;**

**}**

**int subfun(s)**

**char s[10];**

**{**

**struct ffblk ffblk;**

**int d,p=0,i=0;**

**d=findfirst(s,&ffblk,0);**

**while(!d)**

**{**

**printf("%s\n",ffblk.ff\_name);**

**d=findnext(&ffblk);**

**i++;**

**p=p+1;**

**if(p>=22)**

**{**

**printf("Press any key to continue:\n");**

**getch();**

**p=0;**

**}**

**}**

**printf("\nTotal File:%d",i);**

**return 0;**

**}**

**9. C Program for Producer – Consumer Problem concept**

**#include<stdio.h>**

**#include<conio.h>**

**void main()**

**{**

**int ch,n,c1=0,c2=0,produce[23],consume[23];**

**clrscr();**

**printf("\n\n\n\n\n\t\n\n\t\t\tEnter Stack Size : ",n);**

**scanf("%d",&n);**

**while(1)**

**{**

**clrscr();**

**printf("\t\tProducer Stack (Stack Size : %d**

**)\n\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~",n);**

**display(c1,produce);**

**printf("\n\n\t\tConsumer Stack (Stack Size : %d**

**)\n\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~",n);**

**display(c2,consume);**

**printf("\n\t\tCHOICES\n\t\t~~~~~~~\n\t1.Producer\n\t2.Consumer\n\t3.**

**Exit\nEnter your choice : ");**

**scanf("%d",&ch);**

**switch(ch)**

**{**

**case 1:**

**if(c1==n)**

**printf("Produer stack is FULL.So Producer goes to SLEEP\n");**

**else**

**{**

**c1++;**

**printf("\t\tEnter PRODUCE item is :");**

**scanf("%d",&produce[c1]);**

**}**

**break;**

**case 2:**

**if(c2==n)**

**printf("Consumer Stack is FULL.So it goes to SLEEP!..........\n\tReset the Cosumer Stack\n",c2=0);**

**else if(c1==0)**

**printf("\tProducer stack is EMPTY\n");**

**else**

**{**

**c2++;**

**consume[c2]=produce[c1];**

**printf("\t\tCONSUME one item");**

**c1--;**

**}**

**break;**

**case 3:**

**exit(0);**

**default:**

**printf("\tIt is Wrong choice,Please enter correct**

**choice!............\n");**

**}**

**getch();**

**}**

**}**

**display(int c,int stack[])**

**{**

**int i;**

**printf("\n--------------------------------------------------------------------------**

**-----\n");**

**if(c==0)**

**printf("\tStack is EMPTY\n\t\t(Now It is sleeping)");**

**else**

**for(i=1;i<=c;i++)**

**printf("\t%d",stack[i]);**

**printf("\n--------------------------------------------------------------------------**

**-----\n");**

**}**

|  |
| --- |
| **SAMPLE INPUT AND OUTPUT:**  Enter Stack Size :  4      Producer Stack (Stack Size : 4)                 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -------------------------------------------------------------------------------         Stack is EMPTY                 (Now It is sleeping) ------------------------------------------------------------------------- -----                    Consumer Stack (Stack Size : 4)                 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -------------------------------------------------------------------------------         Stack is EMPTY                 (Now It is sleeping) ------------------------------------------------------------------------- -----                  CHOICES                 ~~~~~~~         1.Producer         2.Consumer         3.Exit         Enter your choice :  2              Producer stack is EMPTY                  Producer Stack (Stack Size : 4)                 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -------------------------------------------------------------------------------         10 ------------------------------------------------------------------------- -----                    Consumer Stack (Stack Size : 4)                 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -------------------------------------------------------------------------------         Stack is EMPTY                 (Now It is sleeping) ------------------------------------------------------------------------- -----                  CHOICES                 ~~~~~~~         1.Producer         2.Consumer         3.Exit Enter your choice : 1                         Enter PRODUCE item is :30                  Producer Stack (Stack Size : 4)                 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -------------------------------------------------------------------------------         10      30 ------------------------------------------------------------------------- -----                    Consumer Stack (Stack Size : 4)                 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -------------------------------------------------------------------------------         Stack is EMPTY                 (Now It is sleeping) ------------------------------------------------------------------------- -----                  CHOICES                 ~~~~~~~         1.Producer         2.Consumer         3.Exit Enter your choice :  2                  Producer Stack (Stack Size : 4)                 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -------------------------------------------------------------------------------         10 ------------------------------------------------------------------------- -----                    Consumer Stack (Stack Size : 4)                 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -------------------------------------------------------------------------------         30 ------------------------------------------------------------------------- -----                  CHOICES                 ~~~~~~~         1.Producer         2.Consumer         3.Exit Enter your choice :3 |

|  |
| --- |
|  |

**10. C Program for Dining Philosophers Problem**

**#include<stdio.h>**

**#include<conio.h>**

**#include<dos.h>**

**#include<dir.h>**

**char fn2[20];**

**main()**

**{**

**int c;**

**clrscr();**

**do{**

**printf("\n\t\tMain Menu\n-------------------------------\n");**

**printf("1.Copy a File\n2.Move a File\n3.Exit\n");**

**scanf("%d",&c);**

**switch(c)**

**{**

**case 1:**

**copy\_file();**

**break;**

**case 2:**

**move\_file();**

**break;**

**case 3:**

**exit(0);**

**}**

**}while(c<=3);**

**getch();**

**return 0;**

**}**

**copy\_file()**

**{**

**FILE \*f1,\*f2;**

**char ch,s[10],fn1[20];**

**int a;**

**printf("\nAre u see the privious files(1/0)?");**

**scanf("%d",&a);**

**if(a==1)**

**print\_file();**

**printf("Enter the source file name:");**

**scanf("%s",&fn1);**

**printf("Enter the Destination file name:");**

**scanf("%s",&fn2);**

**f1=fopen(fn1,"r");**

**if(f1==NULL)**

**printf("Can't open the file");**

**else {**

**f2=fopen(fn2,"w");**

**while((ch=getc(f1))!=EOF)**

**putc(ch,f2);**

**printf("One File Copied");**

**fclose(f2);**

**}**

**fclose(f1);**

**return 0;**

**}**

**move\_file()**

**{**

**FILE \*f1,\*f2;**

**char ch,s[10],fn1[20];**

**int a;**

**printf("\nAre u see the privious files(1/0)?");**

**scanf("%d",&a);**

**if(a==1)**

**print\_file();**

**printf("Enter the source file name:");**

**scanf("%s",&fn1);**

**printf("Enter the Destination file name:");**

**scanf("%s",&fn2);**

**f1=fopen(fn1,"r");**

**if(f1==NULL)**

**printf("Can't open the file");**

**else {**

**f2=fopen(fn2,"w");**

**while((ch=getc(f1))!=EOF)**

**putc(ch,f2);**

**printf("One File moved");**

**fclose(f2);**

**remove(fn1);**

**}**

**fclose(f1);**

**return 0;**

**}**

**print\_file()**

**{**

**struct ffblk ffblk;**

**int d,p=0;**

**char ch;**

**d=findfirst("\*.\*",&ffblk,0);**

**while(!d)**

**{**

**printf("%s\n",ffblk.ff\_name);**

**d=findnext(&ffblk);**

**p=p+1;**

**if(p>=20)**

**{**

**printf("Press any key to continue");**

**getchar();**

**p=0;**

**}**

**}**

**return 0;**

**}**

|  |
| --- |
| **SAMPLE INPUT AND OUTPUT:**  Philos name      Right fork     Left fork -------------------------------------------------------------------         0               0               1         1               1               2         2               2               3         3               3               4         4               4               0 -------------------------------------------------------------------- Enter the Two Eating Philosophers number:1 3 Round 1 -------------------------- Philosopherre 1 is eating with rhf=1 and  lhf=2. Philosopherre 3 is eating with rhf=3 and lhf=4. Round  2 ------------------------ Philosopherre 0 is eating with rhf=0 and  lhf=1. Philosopherre 2 is eating with rhf=2 and   lhf=3. Round  4 ------------------------ Philosopherre 2 is eating with rhf=2 and  lhf=3. Philosopherre 4 is eating with rhf=4 and   lhf=0.            Philos name      Right fork     Left fork -------------------------------------------------------------------         0               0               1         1               1               2         2               2               3         3               3               4         4               4               0 -------------------------------------------------------------------- Enter the Two Eating Philosophers number:1 4 Round 1 -------------------------- Philosopherre 1 is eating with rhf=1 and  lhf=2. Philosopherre 4 is eating with rhf=4 and lhf=0. Round  2 ------------------------ Philosopherre 0 is eating with rhf=0 and  lhf=1. Philosopherre 2 is eating with rhf=2 and   lhf=3. Round  4 ------------------------ Philosopherre 2 is eating with rhf=2 and  lhf=3. Philosopherre 4 is eating with rhf=4 and   lhf=0. |

|  |
| --- |
|  |