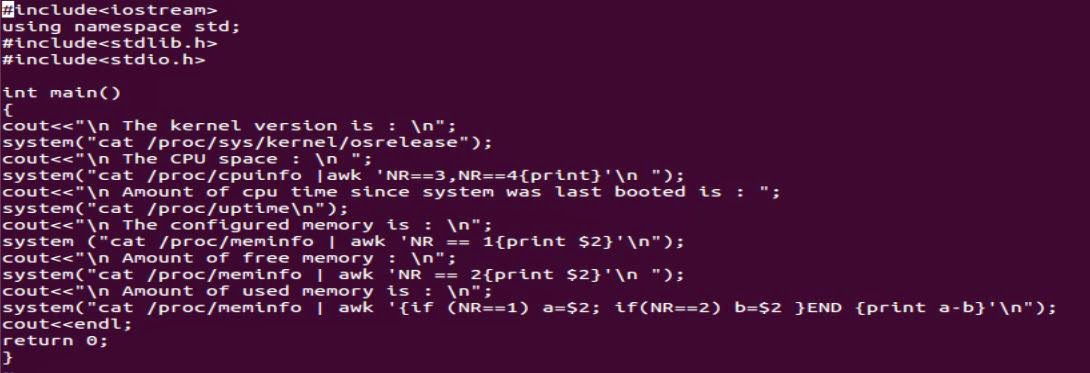
## **PROGRAM1:**

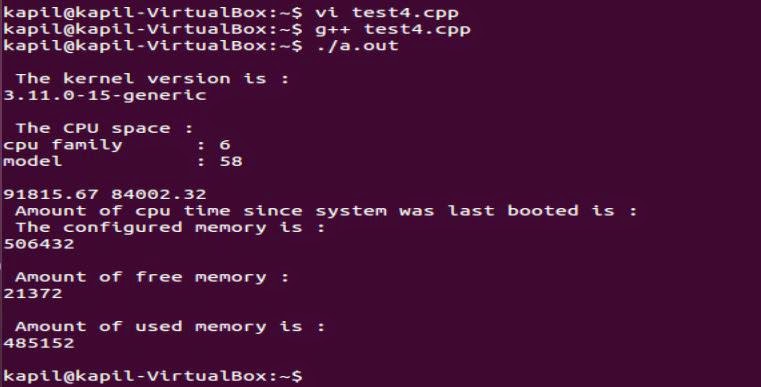
## A PROGRAM to report behaviour of Linux kernel including kernel version, CPU type and model. (CPU information)

## OR

## A PROGRAM to report behaviour of Linux kernel including information on configured memory, amount of free and used memory. (memory information)

[](http://2.bp.blogspot.com/-GjKHH1wjIpo/VGDQXm3XHbI/AAAAAAAAAdk/dLQtt90JuSY/s1600/cpu_new1.JPG)

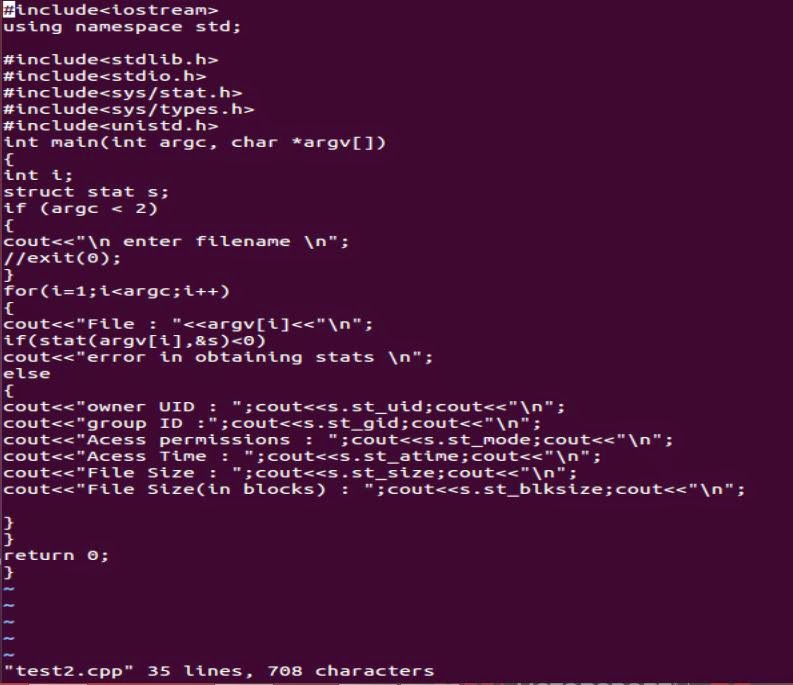
### Output:

[](http://1.bp.blogspot.com/-ING4oDJ7emA/VGDQXOgX4hI/AAAAAAAAAdg/T2jaFu8hFEs/s1600/cpu_new2.JPG)

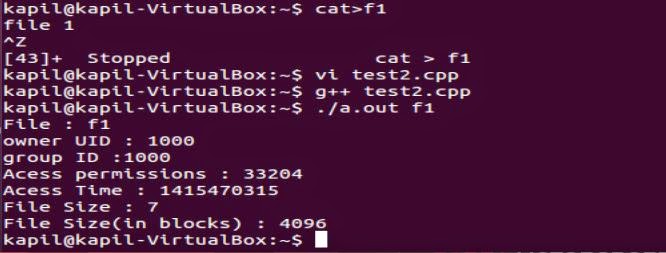
**PROGRAM 2:**

### 4.A PROGRAM to print file details including owner access permissions, file access time, where file name is given as argument.

## A PROGRAM to print file details including owner access permissions, file access time, where file name is given as argument.

[](http://2.bp.blogspot.com/-s_08h1HiZjI/VGDRCKUO2WI/AAAAAAAAAd0/X9neegRwcaE/s1600/ques4.JPG)

### Output:

[](http://2.bp.blogspot.com/--kk3s-uTAn8/VGDRCBhUU2I/AAAAAAAAAdw/ns4A1mZVt9I/s1600/ques4_1.JPG)

**PROGRAM 3:**

###### Program 3: WAP to copy a source file into the target file and display the target file using system calls.

**Contributor: Gursimran [SGGSCC (DU)]**

**#include <stdio.h>**

**#include <sys/types.h>**

**#include <sys/stat.h>**

**#include <fcntl.h>**

**#include <unistd.h>**

**int main()**

**{**

**char buff[50];**

**int n;**

**int fd1=open("source.txt",O\_RDONLY);**

**int fd2=open("target.txt",O\_WRONLY||O\_CREAT);**

**if(fd1 < 0)**

**printf("\nSource File not Opened Successfully.\n");**

**if(fd2 < 0)**

**printf("\nTarget File not Opened Successfully.\n");**

**while((n=read(fd1,buff,50))>0)**

**{**

**write(fd2,buff,n);**

**}**

**write(fd2,"\0",1);**

**if(n < 0)**

**printf("\nFile could not read the content.\n");**

**close(fd1);**

**close(fd2);**

**fd2=open("target.txt",O\_RDONLY);**

**printf("\nFile 1 has been Copied to File 2 and its Contents are\n");**

**while((n=read(fd2,buff,49))>0)**

**{**

**buff[49]='\0';**

**printf(buff);**

**}**

**return 0;**

**}**

**PROGRAM 4:**

###### Program 4: WAP (using fork() and/or exec() commands) where parent and child execute: a. same program, same code b. same program, different code c. different programs d. before terminating, the parent waits for the child to finish its task

**a. same program, same code**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <unistd.h>**

**int main()**

**{**

**pid\_t pid,p;**

**p=fork();**

**pid=getpid();**

**if(p < 0)**

**{**

**fprintf(stderr,"Fork Failed");**

**return 1;**

**}**

**printf("Output of Fork id: %d \n",p);**

**printf("process id is:%d \n",pid);**

**return 0;**

**}**

**b. same program, different code**

**#include <sys/types.h>**

**#include <stdio.h>**

**#include <unistd.h>**

**int main()**

**{**

**pid\_t pid;**

**/\* fork a child process \*/**

**pid = fork();**

**if (pid < 0)**

**{**

**/\* error occurred \*/**

**fprintf(stderr, "Fork Failed");**

**return 1;**

**}**

**else if (pid == 0)**

**{**

**/\* child process \*/**

**execlp("/bin/ls","ls",NULL);**

**}**

**else**

**{**

**printf("Parent Process.");**

**}**

**return 0;**

**}**

**c. different programs**

**#include <iostream>**

**#include <stdio.h>**

**#include <unistd.h>**

**#include <sys/types.h>**

**using namespace std;**

**int main()**

**{**

**pid\_t pid;**

**pid=fork();**

**if(pid<0)**

**{**

**cout<<"No process entered";**

**}**

**else if(pid==0)**

**{**

**execlp("/bin/ls","ls",NULL);**

**cout<<"child process";**

**}**

**else**

**{**

**cout<<"parent process";**

**}**

**return 0;**

**}**

**d. before terminating, the parent waits for the child to finish its task**

**#include <sys/types.h>**

**#include <stdio.h>**

**#include <unistd.h>**

**int main()**

**{**

**pid\_t pid;**

**/\* fork a child process \*/**

**pid = fork();**

**if (pid < 0)**

**{**

**/\* error occurred \*/**

**fprintf(stderr, "Fork Failed");**

**return 1;**

**}**

**else if (pid == 0)**

**{**

**/\* child process \*/**

**execlp("/bin/ls","ls",NULL);**

**}**

**else**

**{**

**/\* parent process \*/**

**/\* parent will wait for the child to complete \*/**

**wait(NULL);**

**printf("Child Complete");**

**}**

**return 0;**

**}**

**PROGRAM 5:**

###### Program 5: WAP to demonstrate producer-consumer problem using shared memory.

**Contributor: Gursimran [SGGSCC (DU)]**

**#include <stdio.h>**

**#include <stdlib.h>**

**int in=0,out=0;**

**int sm[100];**

**void producer(int prod)**

**{**

**if((in+1)%100!=out)**

**{**

**sm[in]=prod;**

**in=(in+1)%100;**

**}**

**else**

**printf("Buffer is Full.");**

**}**

**int consumer()**

**{**

**int cons;**

**if(in!=out)**

**{**

**cons=sm[out];**

**out=(out+1)%100;**

**return cons;**

**}**

**else**

**{**

**printf("Buffer Empty.");**

**return -9999;**

**}**

**}**

**int main()**

**{**

**int con,cons,prod,ch;**

**do{**

**printf("\nEnter if you are a Producer or Consumer.\n");**

**printf("Choose 1. for producer and 2. for Consumer=");**

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

**if(ch==1)**

**{**

**printf("Enter the Input");**

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

**producer(prod);**

**}**

**else**

**{**

**if(ch==2)**

**{**

**cons=consumer();**

**printf("consume output:%d",cons);**

**}**

**else**

**printf("Choose Value between 1 and 2 only.");**

**}**

**printf("If you want to Execute Again\nPress 1 for Yes.");**

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

**}while(con==1);**

**return 0;**

**}**

###### Program 6: WAP to demonstrate InterProcess Communication (IPC) between parent and child using pipe system call.

**Contributor: Gursimran [SGGSCC (DU)]**

**#include <stdio.h>**

**#include <sys/types.h>**

**#include <string.h>**

**#include <unistd.h>**

**#define buffersize 25**

**#define readend 0**

**#define writeend 1**

**int main()**

**{**

**char writemsg[buffersize]="greetings";**

**char readmsg[buffersize];**

**int fd[2];**

**pid\_t pid;**

**if(pipe(fd)==-1)**

**{**

**fprintf(stderr,"pipe failed");**

**return 1;**

**}**

**pid=fork();**

**if(pid < 0)**

**{**

**fprintf(stderr,"fork failed");**

**return 1;**

**}**

**if(pid>0)**

**{**

**printf("parent is writing %s \n",writemsg);**

**close(fd[readend]);**

**write(fd[writeend],writemsg,strlen(writemsg)+1);**

**close(fd[writeend]);**

**}**

**else**

**{**

**close(fd[writeend]);**

**read(fd[readend],readmsg,buffersize);**

**printf("child has read %s",readmsg);**

**close(fd[readend]);**

**}**

**return 0;**

**}**

###### Program 7: Write programs to understand working of Pthread library.

**Contributor: Gursimran [SGGSCC (DU)]**

**#include <pthread.h>**

**#include <stdio.h>**

**int sum; /\* this data is shared by the thread(s) \*/**

**void \*runner(void \*param); /\* threads call this function \*/**

**int main(int argc, char \*argv[])**

**{**

**pthread\_t tid; /\* the thread identifier \*/**

**pthread\_attr\_t attr; /\* set of thread attributes \*/**

**if (argc != 2)**

**{**

**fprintf(stderr,"usage: a.out <integer value>\n");**

**return -1;**

**}**

**if (atoi(argv[1]) < 0)**

**{**

**fprintf(stderr,"%d must be >= 0\n",atoi(argv[1]));**

**return -1;**

**}**

**/\* get the default attributes \*/**

**pthread\_attr\_init(&attr);**

**/\* create the thread \*/**

**pthread\_create(&tid,&attr,runner,argv[1]);**

**/\* wait for the thread to exit \*/**

**pthread\_join(tid,NULL);**

**printf("sum = %d\n",sum);**

**}**

**/\* The thread will begin control in this function \*/**

**void \*runner(void \*param)**

**{**

**int i, upper = atoi(param);**

**sum = 0;**

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

**sum += i;**

**pthread\_exit(0);**

**}**

**PROGRAM 8:**

### 6.Write program to implement FCFS scheduling algorithm

## Write program to implement FCFS scheduling algorithm

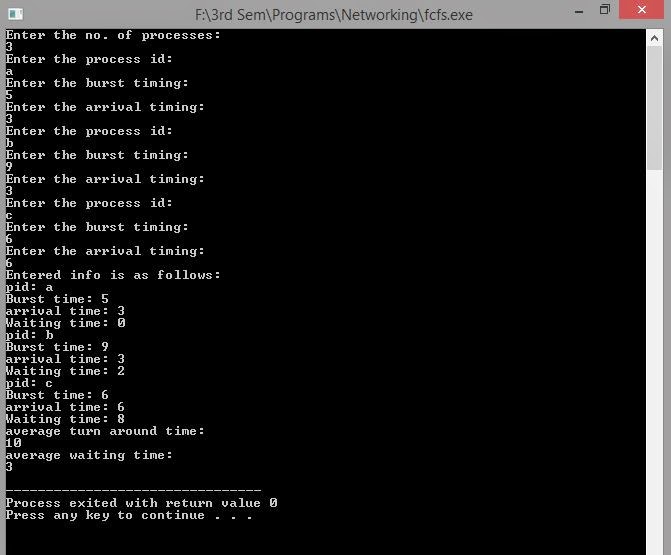
### For Dev C++

---------------------------------------------------------------------------------------------------------

#### #include<iostream> #include<stdio.h> using namespace std; class fcfs {               public:         int burst\_tm,arv\_tm,wait\_tm;         char pid; }; int  main() {      int n,sum=0,carry;      cout<<"Enter the no. of processes:\n";      cin>>n;      fcfs temp;      fcfs \*f=new fcfs[n];      for( int i=0;i<n;i++)      {           cout<<"Enter the process id:\n";           cin>>f[i].pid;           cout<<"Enter the burst timing:\n";           cin>>f[i].burst\_tm;           cout<<"Enter the arrival timing:\n";           cin>>f[i].arv\_tm;           //system("cls");      }          int k=0,flag=1;      while(k!=0)      {                 if(f[k].arv\_tm==0)                 {                                   flag=0;                                   break;                 }                 k++;      }      if(!flag)      {              cout<<"enter all values of arival time as no arrival time is 0\n";              return 1;      }          for(k=0;k<n;k++)      {                      temp=f[k];                      for(int i=k+1;i<n;i++)                      {                              if(temp.arv\_tm>f[i].arv\_tm)                              {                                                         temp=f[i];                                                         f[i]=f[k];                                                         f[k]=temp;                              }                      }      }          cout<<"Entered info is as follows:\n";          int avg\_wait\_tm=0,avg\_around\_tm=0;      for(k=0;k<n;k++)      {                                                                  f[k].wait\_tm=sum-f[k].arv\_tm;                      if(f[k].wait\_tm<0)                                       f[k].wait\_tm=0;                                                          sum+=f[k].burst\_tm;                      avg\_wait\_tm+=f[k].wait\_tm;                      avg\_around\_tm+=f[k].wait\_tm+f[k].burst\_tm;      }          avg\_around\_tm/=n;      avg\_wait\_tm/=n;      for(k=0;k<n;k++)      {                      cout<<"pid: "<<f[k].pid<<endl<<"Burst time: "<<f[k].burst\_tm<<endl<<"arrival time: "<<f[k].arv\_tm<<endl<<"Waiting time: "<<f[k].wait\_tm<<endl;                          }      cout<<"average turn around time:\n"<<avg\_around\_tm<<endl<<"average waiting time:\n"<<avg\_wait\_tm<<endl;      return 0; }

------------------------------------------------------------------------------------------------------------

### Output.............

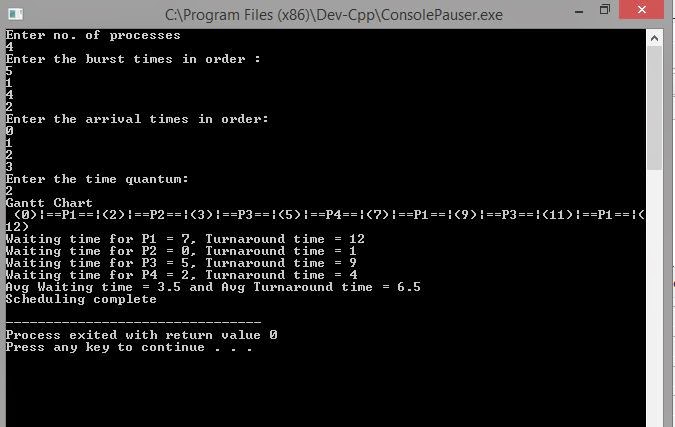
[](http://3.bp.blogspot.com/-YlDfijQJcPc/VBquLTgD4KI/AAAAAAAAANo/onqRm_lXwVM/s1600/fcfs.JPG)

### 7. Write program to implement Round Robin scheduling algorithm

## Using Dev C++

**------------------------------------------------------------------------------------------------------------------**  
 **#include<iostream>**  
**using namespace std;**  
**//ducslectures.blogspot.in**  
**class sched**  
**{**  
**public:**  
**int n,bt[10],at[10],tat[10],wt[10],rt[10],finish[10],twt,ttat,total;**  
**void readData();**  
**void Init();**  
**void dispTime();**  
**void computeRR();**  
**};**  
  
**void sched::readData()**  
**{**  
**cout<<"Enter no. of processes\n";**  
**cin>>n;**  
**cout<<"Enter the burst times in order :\n";**  
**for(int i=0;i<n;i++)**  
**cin>>bt[i];**  
**cout<<"Enter the arrival times in order:\n";**  
**for(int i=0;i<n;i++)**  
**cin>>at[i];**  
**}**  
  
**void sched::Init()**  
**{**  
**total=0;**  
**twt=0;**  
**ttat=0;**  
**for(int i=0; i<n; i++)**  
 **{**  
**rt[i]=bt[i];**  
**finish[i]=0;**  
**wt[i]=0;**  
**tat[i]=0;**  
**total+=bt[i];**  
**}**  
**}**  
 **void sched::dispTime()**  
**{**  
**for(int i=0;i<n;i++)**  
**{**  
  
**twt+=wt[i];**  
**tat[i]=wt[i]+bt[i];**  
**ttat+=tat[i];**  
**cout<<"Waiting time for P"<<(i+1)<<" = "<<wt[i]<<", Turnaround time = "<<tat[i]<<endl;**  
  
**}**  
  
**cout<<"Avg Waiting time = "<<(double)twt/n<<" and Avg Turnaround time = "<<(double)ttat/n<<endl;**  
**cout<<"Scheduling complete\n";**  
**}**  
 **void sched::computeRR(){**  
  
**readData();**  
**Init();**  
**int time,j,q,i,dec=0;**  
**cout<<"Enter the time quantum:\n";**  
**cin>>q;**  
**cout<<"Gantt Chart\n ";**  
**for(time=0;time<total;)**  
**{**  
**for(i=0;i<n;i++)**  
**{**  
**if(at[i]<=time && finish[i]==0)**  
**{**  
  
**cout<<"("<<time<<")|==P"<<(i+1)<<"==|";**  
**if(rt[i]<q)  {**  
**dec=rt[i];**  
**}**  
**else {dec=q;}**  
  
**rt[i]=rt[i]-dec;**  
**if(rt[i]==0)**  
 **finish[i]=1;**  
**for(j=0;j<n;j++)**  
**if(j!=i && finish[j]==0 && at[j]<=time)**  
**wt[j]+=dec;**  
**time=time+dec;**  
  
**}**  
  
**}**  
  
**}                                                         //ducslectures.blogspot.in**  
  
**cout<<"("<<total<<")"<<endl;**  
**dispTime();**  
**}**  
**int main()**  
**{**  
**sched s;**  
**int ch;**  
**s.computeRR();**  
**}**  
  
  
**---------------------------------------------------------------------------------------------------------------**

### Output...........

[](http://4.bp.blogspot.com/-drpZcwPObXs/VCG9cZWJ64I/AAAAAAAAAPQ/EASbcKbZW9o/s1600/rr.JPG)

### 8. Write program to implement SJF scheduling algorithm.

## Write program to implement SJF scheduling algorithm (IF Arrival time is 0)

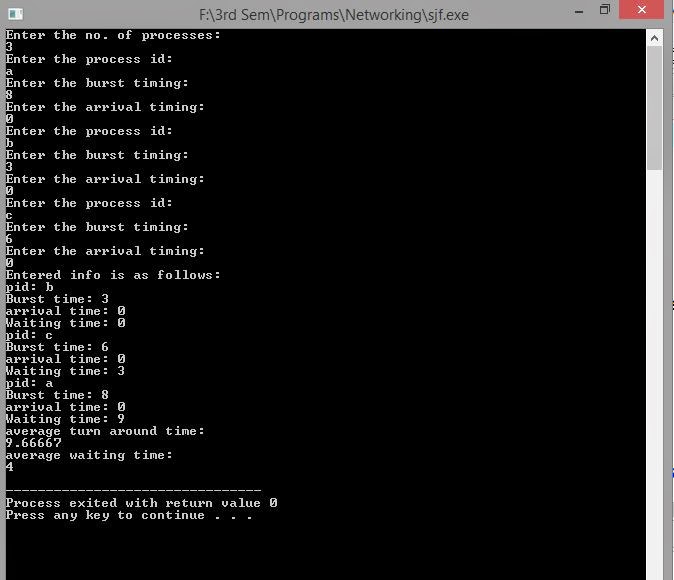
### For Dev C++

--------------------------------------------------------------------------------------------------------

#### #include<iostream> #include<stdio.h> using namespace std; class sjf {               public:         int burst\_tm,arv\_tm,wait\_tm,temp;         char pid; }; int main() {      int n,sum=0,carry;      cout<<"Enter the no. of processes:\n";      cin>>n;      sjf temp;      sjf \*f=new sjf[n];      for( int i=0;i<n;i++)      {           cout<<"Enter the process id:\n";           cin>>f[i].pid;           cout<<"Enter the burst timing:\n";           cin>>f[i].burst\_tm;           cout<<"Enter the arrival timing:\n";           cin>>f[i].arv\_tm;      }          int k=0,flag=1;      while(k!=0)      {                 if(f[k].arv\_tm==0)                 {                                   flag=0;                                   break;                 }                 k++;      }      if(!flag)      {              cout<<"enter all values of arival time as no arrival time is 0\n";              return 1;      }          for(k=0;k<n;k++)      {                      temp=f[k];                      for(int i=k+1;i<n;i++)                      {                              if(temp.burst\_tm>f[i].burst\_tm)                              {                                                         temp=f[i];                                                         f[i]=f[k];                                                         f[k]=temp;                              }                      }      }            cout<<"Entered info is as follows:\n";          float avg\_wait\_tm=0,avg\_around\_tm=0;          for(k=0;k<n;k++)      {                                                                  f[k].wait\_tm=sum-f[k].arv\_tm;                      if(f[k].wait\_tm<0)                                       f[k].wait\_tm=0;                      if(f[k].arv\_tm>(f[k-1].burst\_tm+f[k-1].arv\_tm))                      {                                                                     carry=f[k].arv\_tm-sum;                                                                     sum+=(f[k].burst\_tm+carry);                      }                      else                      sum+=f[k].burst\_tm;                      avg\_wait\_tm+=f[k].wait\_tm;                      avg\_around\_tm+=f[k].wait\_tm+f[k].burst\_tm;      }           avg\_around\_tm/=n;      avg\_wait\_tm/=n;          for(k=0;k<n;k++)      {                      cout<<"pid: "<<f[k].pid<<endl<<"Burst time: "<<f[k].burst\_tm<<endl<<"arrival time: "<<f[k].arv\_tm<<endl<<"Waiting time: "<<f[k].wait\_tm<<endl;                          }          cout<<"average turn around time:\n"<<avg\_around\_tm<<endl<<"average waiting time:\n"<<avg\_wait\_tm<<endl;              return 0;   }

--------------------------------------------------------------------------------------------------------

### Output................

[](http://4.bp.blogspot.com/-W00NfrAUrVw/VBrFM3etVvI/AAAAAAAAAN0/Z_vW05C_vB8/s1600/fcfs1.JPG)

### 8.1Write program to implement SJF scheduling algorithm (Non-Preemptive)

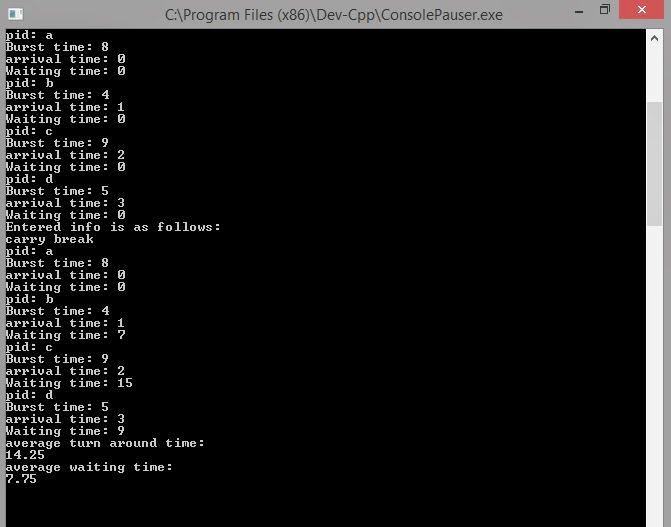
## Using Dev c++

---------------------------------------------------------------------------------------------------------------------

#### #include<iostream> #include<stdio.h> using namespace std; class sjf {               public:         int burst\_tm,arv\_tm,wait\_tm,temp;         char pid;               sjf()         {              temp=0;              //status=0;         } }; int main() {      int n,sum=0,carry=0;      float avg\_wait\_tm=0,avg\_around\_tm=0;      cout<<"Enter the no. of processes:\n";      cin>>n;          sjf temp;      sjf \*f=new sjf[n];      for( int i=0;i<n;i++)      {           cout<<"Enter the process id:\n";           cin>>f[i].pid;           cout<<"Enter the burst timing:\n";           cin>>f[i].burst\_tm;           cout<<"Enter the arrival timing:\n";           cin>>f[i].arv\_tm;              }          int k1=0,flag=1;      while(k1<n)      {                 if(f[k1].arv\_tm==0)                 {                                   flag=0;                                   break;                 }                 k1++;      }      if(flag==1)      {              cout<<"enter all values of arival time as no arrival time is 0\n";              return 1;      }              for(int k=0;k<n;k++)      {                      temp=f[k];                      for(int i=k+1;i<n;i++)                      {                              if(temp.arv\_tm>f[i].arv\_tm)                              {                                                         temp=f[i];                                                         f[i]=f[k];                                                         f[k]=temp;                              }                              //cout<<"f[0]: "<<f[0].pid<<endl;                      }      }          for(int k=0;k<n;k++)      {                      cout<<"pid: "<<f[k].pid<<endl<<"Burst time: "<<f[k].burst\_tm<<endl<<"arrival time: "<<f[k].arv\_tm<<endl<<"Waiting time: "<<f[k].wait\_tm<<endl;                                              }          cout<<"Entered info is as follows:\n";      carry=f[0].arv\_tm;      int i=0;      for(int k=1;k<n;k++)      {                  if(f[k].arv\_tm==carry)                  {                                        if(f[i].burst\_tm>f[k].burst\_tm)                                                                       i=k;                  }                  else                  {                   cout<<"carry break\n";                   break;                  }      }          int j,tem=0,k,b;      do      {          j=0;                  if((f[i].arv\_tm<=sum)&&(f[i].temp!=1))                  {                                     f[i].wait\_tm=sum-f[i].arv\_tm;                                     f[i].temp=1;                                     sum+=f[i].burst\_tm;                  }                  else                  {                      if(tem==0)                      {                                sum=f[i].arv\_tm;                                tem=1;                      }                      f[i].wait\_tm=sum-f[i].arv\_tm;                      f[i].temp=1;                      sum+=f[i].burst\_tm;                  }                  flag=1;                  for(int k=0;k<n;k++)                  {                                      if(f[k].arv\_tm<=sum)                                      {                                                         //flag=0;                                                         if(f[k].temp==1)                                                                         j++;                                                         else if((f[k].burst\_tm<f[i].burst\_tm)||(f[i].temp==1))                                                         {                                                                   flag=0;                                                                   i=k;                                                         }                                      }                  }                                                      if((flag==1)&&(j!=n-1))                    {                                         for(b=i;b<n;b++)                                         if(!f[b].temp)                                         {                                                       carry=f[b].arv\_tm;                                                       i=b;                                                       break;                                         }                                         for(int t=b+1;t<n;t++)                                         {                                                 if(f[t].arv\_tm==carry)                                                 {                                                                       if(f[t].burst\_tm<f[i].burst\_tm)                                                                                                      i=t;                                                 }                                                 else                                                 break;                                         }                  }      }while(j!=n);                                                                                                                                                        for(k=0;k<n;k++)      {                      cout<<"pid: "<<f[k].pid<<endl<<"Burst time: "<<f[k].burst\_tm<<endl<<"arrival time: "<<f[k].arv\_tm<<endl<<"Waiting time: "<<f[k].wait\_tm<<endl;                      avg\_wait\_tm+=f[k].wait\_tm;                      avg\_around\_tm+=f[k].wait\_tm+f[k].burst\_tm;                                              }      avg\_around\_tm/=n;      avg\_wait\_tm/=n;      cout<<"average turn around time:\n"<<avg\_around\_tm<<endl<<"average waiting time:\n"<<avg\_wait\_tm<<endl;          //cout<<"average turn around time:\n"<<avg\_around\_tm<<endl<<"average waiting time:\n"<<avg\_wait\_tm<<endl;      getchar();      getchar();      getchar();          return 0;   }

#### ----------------------------------------------------------------------------------------------------------------

### Output.....

[](http://1.bp.blogspot.com/-X_4R7CUJK7w/VB5pt69XJFI/AAAAAAAAAOg/F8ItzXDOXIo/s1600/sjf.JPG)

### 8.2Write program to implement SJF scheduling algorithm (Preemptive)

## Using Dev C++

---------------------------------------------------------------------------------------------------------------------

#### #include<iostream>

#### using namespace std;

#### //ducslectures.blogspot.in

#### class sched{

#### public:

#### int n,bt[10],at[10],tat[10],wt[10],rt[10],finish[10],twt,ttat,total;

#### void readData();

#### void computeSRT();

#### void Init();

#### void dispTime();

#### int getNextProcess(int);

#### };

#### 

#### void sched::readData()

#### {

#### cout<<"Enter no. of processes\n";

#### cin>>n;

#### cout<<"Enter the burst times in order :\n";

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

#### cin>>bt[i];

#### cout<<"Enter the arrival times in order:\n";

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

#### cin>>at[i];

#### }

#### 

#### void sched::Init(){

#### total=0;

#### twt=0;

#### ttat=0;

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

#### {

#### rt[i]=bt[i];

#### finish[i]=0;

#### wt[i]=0;

#### tat[i]=0;

#### total+=bt[i];

#### }

#### }

#### 

#### void sched::computeSRT()

#### {

#### readData();

#### Init();

#### int time,next=0,old,i;

#### cout<<"Gantt Chart\n ";

#### for(time=0;time<total;time++)

#### {

#### old=next;

#### next=getNextProcess(time);

#### if(old!=next || time==0)

#### {

#### cout<<"("<<time<<")|==P"<<next+1<<"==|";

#### }

#### rt[next]=rt[next]-1;

#### if(rt[next]==0)

#### {

#### finish[next]=1;

#### }

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

#### if(i!=next && finish[i]==0 && at[i]<=time)

#### wt[i]++;

#### }

#### cout<<"("<<total<<")"<<endl;

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

#### if(!finish[i]) {cout<<"Scheduling failed, cannot continue\n"; return;}

#### dispTime();

#### }

#### int sched::getNextProcess(int time)

#### {

#### int i,low;

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

#### if(finish[i]==0){low=i; break; }

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

#### if(finish[i]!=1)

#### if(rt[i]<rt[low] && at[i]<=time)

#### low=i;

#### return low;

#### 

#### }

#### void sched::dispTime()

#### {

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

#### {

#### twt+=wt[i];

#### tat[i]=wt[i]+bt[i];

#### ttat+=tat[i];

#### cout<<"Waiting time for P"<<(i+1)<<" = "<<wt[i]<<", Turnaround time = "<<tat[i]<<endl;

#### }

#### cout<<"Avg Waiting time = "<<(double)twt/n<<" and Avg Turnaround time = "<<(double)ttat/n<<endl;

#### cout<<"Scheduling complete\n";

#### }

#### int main()

#### {

#### sched s;

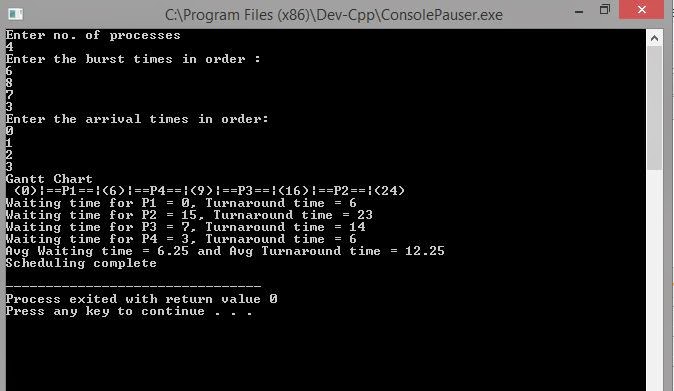
#### s.computeSRT();

#### //ducslectures.blogspot.in

#### }

---------------------------------------------------------------------------------------------------------------------

### Output........

[](http://3.bp.blogspot.com/-hCeDlLVknZw/VCG8JJyR2KI/AAAAAAAAAPE/sO5VLToG4mo/s1600/sjf+p.JPG)

###### Program 8: Write programs to implement the following scheduling algorithms: a. FCFS b. Shortest Job First c. Shortest Remaining Time First d. Non-preemptive priority based e. Preemptive priority based f. Round Robin

**Contributor: Gursimran [SGGSCC (DU)]**

**a. FCFS**

**#include <iostream>**

**using namespace std;**

**class ps**

**{**

**public:**

**int art;**

**int bt;**

**int wt;**

**int ta;**

**ps()**

**{**

**art=bt=wt=ta=0;**

**}**

**};**

**int main()**

**{**

**float wta=0,taa=0;int n;**

**ps tmp;**

**cout<<"\nEnter the Number of Processes=";**

**cin>>n;**

**ps \*p=new ps[n];**

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

**{**

**cout<<"\nEnter the arrival Time for Process "<< i <<"=";**

**cin>>p[i-1].art;**

**cout<<"\nEnter the Burst Time for Process "<< i <<"=";**

**cin>>p[i-1].bt;**

**}**

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

**{**

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

**if(p[i].art<=p[j].art)**

**{**

**p[j].wt=p[i].art+p[i].wt+p[i].bt-p[j].art;**

**if(p[j].wt < 0)**

**p[j].wt=0;**

**}**

**else**

**{**

**tmp=p[i];**

**p[i]=p[j];**

**p[j]=tmp;**

**}**

**p[i].ta=p[i].bt+p[i].wt;**

**}**

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

**{**

**wta+=p[i].wt;**

**taa+=p[i].ta;**

**cout<<"\nThe Waiting Time for Process "<< i+1<<"="<< p[i].wt;**

**cout<<"\nThe Turnaround Time for Process "<< i+1<<"="<< p[i].ta;**

**}**

**cout<<"\nAverage Waiting Time="<< wta/(float)n;**

**cout<<"\nAverage Turnaround Time="<< taa/(float)n;**

**return 0;**

**}**

**b. Shortest Job First**

**#include <bits/stdc++.h>**

**using namespace std;**

**struct Pro**

**{**

**int pid;**

**int bt;**

**};**

**void bubbleSort(Pro arr[], int n)**

**{**

**int i, j;**

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

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

**if (arr[j].bt > arr[j+1].bt)**

**swap(arr[j], arr[j+1]);**

**}**

**bool comparison(Pro a, Pro b)**

**{**

**return (a.bt < b.bt);**

**}**

**void findWaitingTime(Pro p[], int n, int wt[])**

**{**

**wt[0] = 0;**

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

**wt[i] = p[i-1].bt + wt[i-1] ;**

**}**

**void findTurnAroundTime(Pro p[], int n, int wt[], int tat[])**

**{**

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

**tat[i] = p[i].bt + wt[i];**

**}**

**void findavgTime(Pro p[], int n)**

**{**

**int wt[n], tat[n], twt = 0, ttat = 0;**

**findWaitingTime(p, n, wt);**

**findTurnAroundTime(p, n, wt, tat);**

**cout << "\nProcesses "<< " Burst time " << " Waiting time " << " Turn around time\n";**

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

**{**

**twt = twt + wt[i];**

**ttat = ttat + tat[i];**

**cout << " " << p[i].pid << "\t\t" << p[i].bt << "\t " << wt[i] << "\t\t " << tat[i] << endl;**

**}**

**cout << "Average waiting time = " << (float)twt / (float)n;**

**cout << "\nAverage turn around time = " << (float)ttat / (float)n;**

**}**

**int main()**

**{**

**int n;**

**cout<<"\nSJF\nEnter the Number of Processes=";**

**cin>>n;**

**Pro \*proc= new Pro[n];**

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

**{**

**cout<<"\nEnter the Burst Time for Process "<< i+1<<"=";**

**cin>>proc[i].bt;**

**proc[i].pid=i+1;**

**}**

**bubbleSort(proc,n);**

**cout << "Order in which process gets executed\n";**

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

**cout << proc[i].pid <<" ";**

**findavgTime(proc, n);**

**return 0;**

**}**

**POPULAR POSTS**

* [**How to Enable “Tab Groups” feature in Google Chrome Browser**](https://blog.tutorialsduniya.com/how-to-enable-tab-groups-feature-in-google-chrome-browser)
* [**How to move Google Chrome’s ToolBar to bottom**](https://blog.tutorialsduniya.com/how-to-move-google-chromes-toolbar-to-bottom)
* [**List of Mobiles that receives Android 10 update**](https://blog.tutorialsduniya.com/list-of-mobiles-that-receives-android-10-update)
* [**How to use Dark Mode in Google Chrome**](https://blog.tutorialsduniya.com/how-to-use-dark-mode-in-google-chrome)
* [**How to Send Pictures without Losing Quality on WhatsApp**](https://blog.tutorialsduniya.com/how-to-send-pictures-without-losing-quality-on-whatsapp)

**c. Shortest Remaining Time First**

**#include <iostream>**

**using namespace std;**

**class sched**

**{**

**public:**

**int n,bt[10],at[10],tat[10],wt[10],rt[10],finish[10],twt,ttat,total;**

**void readData();**

**void computeSRT();**

**void Init();**

**void dispTime();**

**int getNextProcess(int);**

**};**

**void sched::readData()**

**{**

**cout<<"Enter no. of processes\n";**

**cin>>n;**

**cout<<"Enter the burst times in order :\n";**

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

**cin>>bt[i];**

**cout<<"Enter the arrival times in order:\n";**

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

**cin>>at[i];**

**}**

**void sched::Init()**

**{**

**total=0;**

**twt=0;**

**ttat=0;**

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

**{**

**rt[i]=bt[i];**

**finish[i]=0;**

**wt[i]=0;**

**tat[i]=0;**

**total+=bt[i];**

**}**

**}**

**void sched::computeSRT()**

**{**

**readData();**

**Init();**

**int time,next=0,old,i;**

**cout<<"Gantt Chart\n ";**

**for(time=0; time < total; time++)**

**{**

**old=next;**

**next=getNextProcess(time);**

**if(old!=next || time==0)**

**{**

**cout<<"("<< time<<")|==P"<< next+1<<"==|";**

**}**

**rt[next]=rt[next]-1;**

**if(rt[next]==0)**

**{**

**finish[next]=1;**

**}**

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

**if(i!=next && finish[i]==0 && at[i]<=time)**

**wt[i]++;**

**}**

**cout<<"("<< total<<")"<< endl;**

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

**if(!finish[i]) {cout<<"Scheduling failed, cannot continue\n"; return;}**

**dispTime();**

**}**

**int sched::getNextProcess(int time)**

**{**

**int i,low;**

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

**if(finish[i]==0){low=i; break; }**

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

**if(finish[i]!=1)**

**if(rt[i] < rt[low] && at[i]<=time)**

**low=i;**

**return low;**

**}**

**void sched::dispTime()**

**{**

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

**{**

**twt+=wt[i];**

**tat[i]=wt[i]+bt[i];**

**ttat+=tat[i];**

**cout<<"Waiting time for P"<<(i+1)<<" = "<< wt[i]<<", Turnaround time = "<< tat[i]<< endl;**

**}**

**cout<<"Avg Waiting time = "<<(double)twt/n<<" and Avg Turnaround time = "<<(double)ttat/n<< endl;**

**cout<<"Scheduling complete\n";**

**}**

**int main()**

**{**

**sched s;**

**s.computeSRT();**

**}**

**d. Non-Preemptive priority based**

**#include <bits/stdc++.h>**

**using namespace std;**

**struct Process**

**{**

**int pid;**

**int bt;**

**int priority;**

**};**

**bool comparison(Process a, Process b)**

**{**

**return (a.priority > b.priority);**

**}**

**void findWaitingTime(Process proc[], int n, int wt[])**

**{**

**wt[0] = 0;**

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

**wt[i] = proc[i-1].bt + wt[i-1] ;**

**}**

**void findTurnAroundTime( Process proc[], int n, int wt[], int tat[])**

**{**

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

**tat[i] = proc[i].bt + wt[i];**

**}**

**void findavgTime(Process proc[], int n)**

**{**

**int wt[n], tat[n], total\_wt = 0, total\_tat = 0;**

**findWaitingTime(proc, n, wt);**

**findTurnAroundTime(proc, n, wt, tat);**

**cout << "\nProcesses "<< " Burst time " << " Waiting time " << " Turn around time\n";**

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

**{**

**total\_wt = total\_wt + wt[i];**

**total\_tat = total\_tat + tat[i];**

**cout << " " << proc[i].pid << "\t\t" << proc[i].bt << "\t " << wt[i] << "\t\t " << tat[i] << endl;**

**}**

**cout << "\nAverage waiting time = " << (float)total\_wt / (float)n;**

**cout << "\nAverage turn around time = " << (float)total\_tat / (float)n;**

**}**

**void priorityScheduling(Process proc[], int n)**

**{**

**sort(proc, proc + n, comparison);**

**cout<< "Order in which processes gets executed \n";**

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

**cout << proc[i].pid <<" " ;**

**findavgTime(proc, n);**

**}**

**int main()**

**{**

**int n;**

**cout<<"\nPriority Scheduling\nEnter the Number of Processes=";**

**cin>>n;**

**Process \*proc= new Process[n];**

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

**{**

**cout<<"\nEnter the Burst Time for Process "<< i+1<<"=";**

**cin>>proc[i].bt;**

**cout<<"\nEnter the Priority of Process "<< i+1<<"=";**

**cin>>proc[i].priority;**

**proc[i].pid=i+1;**

**}**

**priorityScheduling(proc, n);**

**return 0;**

**}**

**e. Preemptive priority based**

**#include <iostream>**

**using namespace std;**

**int main()**

**{**

**int n;**

**cout<<"enter your toal no. of process";**

**cin>>n;**

**float total,wait[n];**

**float p[n],twaiting=0,waiting=0;**

**int proc;**

**int stack[n];**

**float brust[n],arrival[n],sbrust,temp[n],top=n,prority[n];**

**int i;**

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

**{**

**p[i]=i;**

**stack[i]=i;**

**cout<<"enter arival time :";**

**cin>>arrival[i];**

**cout<<endl<<"enter brust time:";**

**cin>>brust[i];**

**cout<<endl<<"enter prority time:";**

**cin>>prority[i];**

**temp[i]=arrival[i];**

**sbrust=brust[i]+sbrust;**

**}**

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

**{**

**//section 1**

**proc=stack[0];**

**if(temp[proc]==i)**

**twaiting=0;**

**else**

**twaiting=i-(temp[proc]);**

**temp[proc]=i+1;**

**wait[proc]=wait[proc]+twaiting;**

**waiting=waiting+(twaiting);**

**brust[proc]=brust[proc]-1;**

**if(brust[proc]==0)**

**{**

**for(int x=0;x<top-1;x++)**

**stack[x]=stack[x+1];**

**top=top-1;**

**}**

**for(int z=0;z<top-1;z++)**

**{**

**if((prority[stack[0]]>prority[stack[z+1]]) && (arrival[stack[z+1]] <= i+1))**

**{**

**int t=stack[0];**

**stack[0]=stack[z+1];**

**stack[z+1]=t;**

**}**

**}**

**}**

**cout<<"Average waiting time is:"<<waiting/n;**

**float tu=(sbrust+waiting)/n;**

**cout<<endl<<"Average turnaround time is:"<<tu;**

**return 0;**

**POPULAR POSTS**

* [**How to Enable “Tab Groups” feature in Google Chrome Browser**](https://blog.tutorialsduniya.com/how-to-enable-tab-groups-feature-in-google-chrome-browser)
* [**How to combine multiple PDF files on Mobile**](https://blog.tutorialsduniya.com/how-to-combine-multiple-pdf-files-on-android-mobile)

**}[List of Mobiles that receives Android 10 update](https://blog.tutorialsduniya.com/list-of-mobiles-that-receives-android-10-update" \t "_blank)**

* [**How to use Dark Mode in Google Chrome**](https://blog.tutorialsduniya.com/how-to-use-dark-mode-in-google-chrome)
* [**How to Schedule Text SMS on Android Mobile**](https://blog.tutorialsduniya.com/how-to-schedule-text-sms-on-android-mobile)

**f. Round Robin**

**#include <iostream>**

**using namespace std;**

**void findWaitingTime(int processes[], int n, int bt[], int wt[], int quantum)**

**{**

**int rem\_bt[n];**

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

**rem\_bt[i] = bt[i];**

**int t = 0;**

**while (1)**

**{**

**bool done = true;**

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

**{**

**if (rem\_bt[i] > 0)**

**{**

**done = false;**

**if (rem\_bt[i] > quantum)**

**{**

**t += quantum;**

**rem\_bt[i] -= quantum;**

**}**

**else**

**{**

**t = t + rem\_bt[i];**

**wt[i] = t - bt[i];**

**rem\_bt[i] = 0;**

**}**

**}**

**}**

**if (done == true)**

**break;**

**}**

**}**

**void findTurnAroundTime(int processes[], int n, int bt[], int wt[], int tat[])**

**{**

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

**tat[i] = bt[i] + wt[i];**

**}**

**void findavgTime(int processes[], int n, int bt[], int quantum)**

**{**

**int wt[n], tat[n], total\_wt = 0, total\_tat = 0;**

**findWaitingTime(processes, n, bt, wt, quantum);**

**findTurnAroundTime(processes, n, bt, wt, tat);**

**cout << "Processes "<< " Burst time "<< " Waiting time " << " Turn around time\n";**

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

**{**

**total\_wt = total\_wt + wt[i];**

**total\_tat = total\_tat + tat[i];**

**cout << " " << i+1 << "\t\t" << bt[i] <<"\t " << wt[i] <<"\t\t " << tat[i] << endl;**

**}**

**cout << "Average waiting time = " << (float)total\_wt / (float)n;**

**cout << "\nAverage turn around time = " << (float)total\_tat / (float)n;**

**}**

**int main()**

**{**

**cout<<"\nRR\n";**

**int processes[] = { 1, 2, 3};**

**int n = sizeof processes / sizeof processes[0];**

**int bt[] = {10, 5, 8};**

**int quan= 2;**

**findavgTime(processes, n, bt, quan);**

**return 0;**

**}**

###### Program 9: WAP to implement firstfit, bestfit and worstfit allocation strategies.

**Contributor: Gursimran [SGGSCC (DU)]**

**First Fit**

**#include <iostream>**

**using namespace std;**

**struct job**

**{**

**int no;**

**int size;**

**int add;**

**};**

**struct mem**

**{**

**int no;**

**int size;**

**int f;**

**int jno;**

**};**

**int main()**

**{**

**cout<<"\n Enter No of Blocks in memory:";**

**int loc;**

**cin>>loc;**

**int x,y;**

**mem m[99],tempm;**

**for (int i=0; i < loc; i++)**

**{**

**cout<<"\n Enter Size of block"<< i+1<<":";**

**cin>>m[i].size;**

**m[i].no=i+1;**

**m[i].f=0;**

**m[i].jno=0;**

**}**

**int jobn;**

**cout<<"\n Enter No of Jobs:";**

**cin>>jobn;**

**job j[99];**

**for(x=0; x < jobn; x++)**

**{**

**cout<<"\n Enter size of Job"<< x+1<<":";**

**cin>>y;**

**j[x].no=x+1;**

**j[x].size=y;**

**j[x].add=0;**

**}**

**cout<<"\nJobNo\tJobSize";**

**for(x=0; x < jobn; x++)**

**{**

**cout<<"\n";**

**cout<< j[x].no;**

**cout<<"\t"<< j[x].size;**

**}**

**for (x=0; x < jobn; x++)**

**{**

**for(y=0; y < loc; y++)**

**{**

**if((m[y].f==0)&&(m[y].size>=j[x].size)&&(j[x].add!=1))**

**{**

**m[y].jno=j[x].no;**

**m[y].f=1;**

**j[x].add=1;**

**}**

**}**

**}**

**cout<<"\nBlockNo\tMemSize\tJobNo\tJobSize\tStatus\tInternalFreg";**

**for(x=0; x < loc; x++)**

**{**

**cout<<"\n";**

**cout<< m[x].no;**

**cout<<"\t"<< m[x].size;**

**cout<<"\t"<< m[x].jno;**

**y=m[x].jno;**

**y=y-1;**

**if(m[x].jno!=0)**

**cout<<"\t"<< j[y].size;**

**else**

**cout<<"\t"<<"0";**

**cout<<"\t"<< m[x].f;**

**if(m[x].jno!=0)**

**cout<<"\t"<< m[x].size-j[y].size;**

**else**

**cout<<"\t"<<"0";**

**}**

**return 0;**

**}**

**Best Fit**

**#include <iostream>**

**using namespace std;**

**struct job**

**{**

**int no;**

**int size;**

**int add;**

**};**

**struct mem**

**{**

**int no;**

**int size;**

**int f;**

**int jno;**

**};**

**int main()**

**{**

**cout<<"\n Enter No of Blocks in memory:";**

**int loc;**

**cin>>loc;**

**int x,y;**

**mem m[99],tempm;**

**for (int i=0; i < loc; i++)**

**{**

**cout<<"\n Enter Size of block"<< i+1<<":";**

**cin>>m[i].size;**

**m[i].no=i+1;**

**m[i].f=0;**

**m[i].jno=0;**

**}**

**int jobn;**

**cout<<"\n Enter No of Jobs:";**

**cin>>jobn;**

**job j[99];**

**for(x=0; x < jobn; x++)**

**{**

**cout<<"\n Enter size of Job"<< x+1<<":";**

**cin>>y;**

**j[x].no=x+1;**

**j[x].size=y;**

**j[x].add=0;**

**}**

**cout<<"\nJobNo\tJobSize";**

**for(x=0; x < jobn; x++)**

**{**

**cout<<"\n";**

**cout<< j[x].no;**

**cout<<"\t"<< j[x].size;**

**}**

**for (x=0; x < loc; x++)**

**{**

**for (y=0; y < loc; y++)**

**{**

**if(m[x].size < m[y].size)**

**{**

**tempm=m[x];**

**m[x]=m[y];**

**m[y]=tempm;**

**}**

**}**

**}**

**for (x=0; x < jobn; x++)**

**{**

**for(y=0;y < loc; y++)**

**{**

**if((m[y].f==0)&&(m[y].size>=j[x].size)&&(j[x].add!=1))**

**{**

**m[y].jno=j[x].no;**

**m[y].f=1;**

**j[x].add=1;**

**}**

**}**

**}**

**for (x=0; x < loc; x++)**

**{**

**for (y=0; y < loc; y++)**

**{**

**if(m[x].no < m[y].no)**

**{**

**tempm=m[x];**

**m[x]=m[y];**

**m[y]=tempm;**

**}**

**}**

**}**

**cout<<"\nBlockNo\tMemSize\tJobNo\tJobSize\tStatus\tInternalFreg";**

**for(x=0; x < loc; x++)**

**{**

**cout<<"\n";**

**cout<< m[x].no;**

**cout<<"\t"<< m[x].size;**

**cout<<"\t"<< m[x].jno;**

**y=m[x].jno;**

**y=y-1;**

**if(m[x].jno!=0)**

**cout<<"\t"<< j[y].size;**

**else**

**cout<<"\t"<<"0";**

**cout<<"\t"<< m[x].f;**

**if(m[x].jno!=0)**

**cout<<"\t"<< m[x].size-j[y].size;**

**else**

**cout<<"\t"<<"0";**

**}**

**return 0;**

**}**

**POPULAR POSTS**

* [**How to combine multiple PDF files on Mobile**](https://blog.tutorialsduniya.com/how-to-combine-multiple-pdf-files-on-android-mobile)
* [**List of Mobiles that receives Android 10 update**](https://blog.tutorialsduniya.com/list-of-mobiles-that-receives-android-10-update)
* [**How to use Dark Mode in Google Chrome**](https://blog.tutorialsduniya.com/how-to-use-dark-mode-in-google-chrome)
* [**How to Schedule Text SMS on Android Mobile**](https://blog.tutorialsduniya.com/how-to-schedule-text-sms-on-android-mobile)
* [**How to Send Pictures without Losing Quality on WhatsApp**](https://blog.tutorialsduniya.com/how-to-send-pictures-without-losing-quality-on-whatsapp)

**Worst Fit**

**#include <iostream>**

**using namespace std;**

**int main()**

**{**

**int i,j,k,n,m[10],p[10],po[20],flag,z,y,temp,temp1;**

**cout<<"enter memory partition:\t";**

**cin>>n;**

**cout<<"\nenter memory size for\n";**

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

**{**

**cout<<"\npartition "<< i<<" :\t";**

**cin>>m[i];**

**po[i]=i;**

**}**

**cout<<"\nenter process:\t";**

**cin>>j;**

**cout<<"\nenter memory size for\n";**

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

**{**

**cout<<"\nprocess "<< i<<" :\t";**

**cin>>p[i];**

**}**

**cout<<"\nworst fit\n\t";**

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

**{**

**for(z=y;z<=n;z++)**

**{**

**if(m[y] < m[z])**

**{**

**temp=m[y];**

**m[y]=m[z];**

**m[z]=temp;**

**temp1=po[y];**

**po[y]=po[z];**

**po[z]=temp1;**

**}**

**}**

**}**

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

**{**

**flag=1;**

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

**{**

**if(p[i]<=m[k])**

**{**

**cout<<"\nprocess "<< i<<" whose memory size is "<< p[i]<<"KB allocated at memory partition:\t"<< po[k];**

**m[k]=m[k]-p[i];**

**break;**

**}**

**else**

**{**

**flag++;**

**}**

**}**

**if(flag>n)**

**{**

**cout<<"\nprocess "<< i<<" whose memory size is "<< p[i]<<"KB can't be allocated";**

**}**

**}**

**return 0;**

**}**

###### Program 10: WAP to map logical addresses to physical addresses in a paging scheme. Define the necessary data structures required for the program. Page size and physical memory size should be taken as input from the user. Also accept process id and its size from the user and allocate memory to the process. Make an interactive program to perform the following: a. Accept a process id and page no and display frame number for a valid input b. Accept a process id to deallocate and display the frame table

**Contributor: Gursimran [SGGSCC (DU)]**

**#include <stdio.h>**

**#include <math.h>**

**int bit\_calc(int);**

**int bin\_conv(int);**

**int dec\_conv(int,int);**

**void disp\_bin\_res(int,int);**

**int bin\_result[40],input[32];**

**void main()**

**{**

**int p\_s, pg\_s, ph\_s,f\_n,p\_n;**

**int pg\_tab[5][3];**

**int i;**

**int p\_bits,l\_bits,p\_b,l\_b;**

**int page\_bits;**

**printf("Please enter the Process Size in KB: ");**

**scanf("%d",&p\_s);**

**printf("Please enter Page Size in KB: ");**

**scanf("%d",&pg\_s);**

**page\_bits = bit\_calc(pg\_s\*1024);**

**printf("Please enter Physical Memory Size in KB: ");**

**scanf("%d",&ph\_s);**

**for(i=0; i < 5; i++)**

**{**

**printf("Please enter data for Page Table entry %d\n",(i+1));**

**printf("Page No: ");**

**scanf("%d",&pg\_tab[i][0]);**

**printf("Frame No: ");**

**scanf("%d",&pg\_tab[i][1]);**

**printf("Valid/Invalid Bit: ");**

**scanf("%d",&pg\_tab[i][2]);**

**}**

**f\_n=ph\_s/pg\_s;**

**printf("The Total number of Frames in the Physical Memory are: %d\n",f\_n);**

**p\_n=p\_s/pg\_s;**

**printf("The Total number of entries in the Page Table are: %d\n",p\_n);**

**p\_bits = bit\_calc(ph\_s);**

**printf("The number of bits in the Physical Address are: %d\n",(p\_bits+page\_bits));**

**printf("The distribution is %d:%d\n",p\_bits,page\_bits);**

**p\_b=p\_bits+page\_bits;**

**l\_bits = bit\_calc(p\_s);**

**printf("The number of bits in the Logical Address are: %d\n",(l\_bits+page\_bits));**

**printf("The distribution is %d:%d\n",l\_bits,page\_bits);**

**l\_b=l\_bits+page\_bits;**

**while(1)**

**{**

**printf("Please enter the logical address:\n");**

**for(i = 0; i < l\_b; i++)**

**{**

**int temp;**

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

**if(temp == 2)**

**{**

**i = -1;**

**break;**

**}**

**input[i] = temp;**

**}**

**if(i == -1)**

**break;**

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

**int page\_no = dec\_conv(0,l\_bits);**

**int pos = -1;**

**for(i=0; i < 5; i++)**

**{**

**if(page\_no == pg\_tab[i][0])**

**pos = i;**

**}**

**if(pos != -1)**

**{**

**if(pg\_tab[pos][2] == 1)**

**{**

**printf("The generated Physical Address is: ");**

**int p\_addr = pg\_tab[pos][1];**

**p\_addr = p\_addr \* pg\_s \* 1024;**

**p\_addr += dec\_conv(l\_bits,l\_b);**

**int start = bin\_conv(p\_addr);**

**disp\_bin\_res(start,p\_b);**

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

**}**

**else**

**printf("Page Fault!\n");**

**}**

**else**

**printf("Page not found!\n");**

**i = 0;**

**}**

**}**

**int dec\_conv(int start,int stop)**

**{**

**int i;**

**int res=0;**

**int n = stop - start;**

**int j = 0;**

**for(i=start; i < stop; i++)**

**{**

**double temp = pow(2,(n-j-1));**

**res=res+(input[i]\*(int)temp);**

**j++;**

**}**

**return res;**

**}**

**int bin\_conv(int n)**

**{**

**int i=0;**

**while((n/2)>=2)**

**{**

**bin\_result[i]=n%2;**

**n/=2;**

**i++;**

**}**

**i++;**

**bin\_result[i]=1;**

**return i;**

**}**

**void disp\_bin\_res(int n,int min\_l)**

**{**

**int i;**

**if(min\_l > (n+1))**

**{**

**for(i=0;i<(min\_l-n-1);i++)**

**{**

**printf("%d",0);**

**}**

**}**

**for(i=n;i>=0;i--)**

**{**

**printf("%d",bin\_result[i]);**

**}**

**}**

**int bit\_calc(int p\_s)**

**{**

**int bits = 1;**

**while((p\_s / 2) >= 2)**

**{**

**bits++;**

**p\_s = p\_s/2;**

**}**

**return bits;**

**}**