Operatings System Lab (RCS 451)

Topic 1: Introduction to Linux

1.1 Steps to install Linux on Virtual Box

1. Get a Setup .exe file of Virtual Box.
2. Arrange .iso file of Linux and mount it on Virtual Box.
3. After mount there is two option present on it, first is to just try ubuntu or install it after meeting necessary condition of free memory and other setup condition for it.
4. After installation we can do the work on ubuntu similar to Operating System working Environment.

1.2 Implementation of Commands on Linux

:

Sol. Following are some of the commands which are used in Ubuntu for the general processing:-

|  |  |  |
| --- | --- | --- |
| **File Commands:-**  Ls-directory listing  Ls –al-formatted listing with hidden files  Cd dir- change directory to dir  Cd –change to home  Pwd-show current directory  Mkdir (filename)- create directory dir (if file name is dir)  Rm file- delete file  Rm –r dir- delete directory dir  Rm –f file –force remove file  Rm –rf dir- force remove directory dir  Cp file1 file2- copy file1 to file2  Mv file1 file2-rename file1 to file2  Ln –s file link- create symbolic link ‘link ‘ to file  Touch file-create or update file  Cat>file-place standard input into file  More file-output the contents of the file  Less file- output the contents of the file  Head file- output the first 10 lines of the file  Tail file-output the last 10 lines of the file  Tail –f file- output contents of file as it grows | **Network Commands**  Ping host-ping host ‘host’.  Whois domain- get whois for domain  Dig domain- get DNS for domain  Dig –x host-reverse lookup host  Wget file-download file  Wget –c file-continue stopped download  Wget –r url- recursively download files from url  **Installation-**  ./configure  Make  Make install | **System Information Commands**  Date-show current date/time  Cal- show this month’s calendar  Uptime-show uptime  w-display who is online  whoami-who are you logged in as  cat /proc/cpuinfo-cpu info  cat /proc/meminfo-memory information  man command-show manual for command  df-show disk usage  du- shows directory space usage  du –sh-human readable size in GB  free-show memory and swap usage  whereis app-show possible location of app  which app-show which app wil be run by default |

Objective: Running C programs on Linux platform

Sol: To write c programs on linux following steps are proceeded:

Steps:

1. You first need to open the editor of linux.

$- gedit filename.c

1. Write down the code, please mind since it is a gcc compiler so console is not taken into account, do not include conio.h in your file , and return type must be something , say int main or float main() and return 0 or return 0.0 respectively.

Let the code be

#include<stdio.h>

Int main()

{

Printf(“Hello ubuntu on c”);

}

Save the file by pressing ctrl+z and exit

1. After saving the file you can compile it by typing code

gcc –o hello hello.c{Enter}.

And now run by typing

./hello {Enter}

You will get the output on your screen as

[Hello ubuntu on c]

{It is just a simple program to just give view of how you can run program on linux}

Object : FCFS CPU SCHEDULING ALGORITHM

#include<stdio.h>

#include<conio.h>

main()

{

int bt[20], wt[20], tat[20], i, n;

float wtavg, tatavg;

clrscr();

printf("\nEnter the number of processes -- ");

scanf("%d", &n);

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

{

printf("\nEnter Burst Time for Process %d -- ", i);

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

}

wt[0] = wtavg = 0;

tat[0] = tatavg = bt[0];

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

{

wt[i] = wt[i-1] +bt[i-1];

tat[i] = tat[i-1] +bt[i];

wtavg = wtavg + wt[i];

tatavg = tatavg + tat[i];

}

printf("\t PROCESS \tBURST TIME \t WAITING TIME\t TURNAROUND TIME\n");

f or(i=0;i<n;i++)

printf("\n\t P%d \t\t %d \t\t %d \t\t %d", i, bt[i], wt[i], tat[i]);

printf("\nAverage Waiting Time -- %f", wtavg/n);

printf("\nAverage Turnaround Time -- %f", tatavg/n);

getch();

Object : SJF CPU SCHEDULING ALGORITHM

#include<stdio.h>

#include<conio.h>

main()

{

int p[20], bt[20], wt[20], tat[20], i, k, n, temp;

float wtavg, tatavg;

clrscr();

printf("\nEnter the number of processes -- ");

scanf("%d", &n);

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

{

p[i]=i;

printf("Enter Burst Time for Process %d -- ", i);

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

}

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

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

if(bt[i]>bt[k])

{

temp=bt[i];

bt[i]=bt[k];

bt[k]=temp;

temp=p[i];

p[i]=p[k];

p[k]=temp;

}

wt[0] = wtavg = 0;

tat[0] = tatavg = bt[0];

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

{

wt[i] = wt[i-1] +bt[i-1];

tat[i] = tat[i-1] +bt[i];

wtavg = wtavg + wt[i];

tatavg = tatavg + tat[i];

}

printf("\n\t PROCESS \tBURST TIME \t WAITING TIME\t TURNAROUND TIME\n");

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

printf("\n\t P%d \t\t %d \t\t %d \t\t %d", p[i], bt[i], wt[i], tat[i]);

printf("\nAverage Waiting Time -- %f", wtavg/n);

printf("\nAverage Turnaround Time -- %f", tatavg/n);

getch();

}

Object 3: ROUND ROBIN CPU SCHEDULING ALGORITHM

#include<stdio.h>

#include<conio.h>

Void main()

{

int i,j,n,bu[10],wa[10],tat[10],t,ct[10],max;

float awt=0,att=0,temp=0;

clrscr();

printf("Enter the no of processes -- ");

scanf("%d",&n);

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

{

printf("\nEnter Burst Time for process %d -- ", i+1);

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

ct[i]=bu[i];

}

printf("\nEnter the size of time slice -- ");

scanf("%d",&t);

max=bu[0];

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

if(max<bu[i])

max=bu[i];

for(j=0;j<(max/t)+1;j++)

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

if(bu[i]!=0)

if(bu[i]<=t)

{

tat[i]=temp+bu[i];

temp=temp+bu[i];

bu[i]=0;

}

else

{

bu[i]=bu[i]-t;

temp=temp+t;

}

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

{

wa[i]=tat[i]-ct[i];

att+=tat[i];

awt+=wa[i];

}

printf("\nThe Average Turnaround time is -- %f",att/n);

printf("\nThe Average Waiting time is -- %f ",awt/n);

printf("\n\tPROCESS\t BURST TIME \t WAITING TIME\tTURNAROUND TIME\n");

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

printf("\t%d \t %d \t\t %d \t\t %d \n",i+1,ct[i],wa[i],tat[i]);

getch();

Object 4: PRIORITY CPU SCHEDULING ALGORITHM

#include<stdio.h>

main()

{

int p[20],bt[20],pri[20], wt[20],tat[20],i, k, n, temp;

float wtavg, tatavg;

clrscr();

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

scanf("%d",&n);

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

{

p[i] = i;

printf("Enter the Burst Time & Priority of Process %d --- ",i);

scanf("%d %d",&bt[i], &pri[i]);

}

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

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

if(pri[i] > pri[k])

{

temp=p[i];

p[i]=p[k];

p[k]=temp;

temp=bt[i];

bt[i]=bt[k];

bt[k]=temp;

temp=pri[i];

pri[i]=pri[k];

pri[k]=temp;

}

wtavg = wt[0] = 0;

tatavg = tat[0] = bt[0];

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

{

wt[i] = wt[i-1] + bt[i-1];

tat[i] = tat[i-1] + bt[i];

wtavg = wtavg + wt[i];

tatavg = tatavg + tat[i];

}

printf("\nPROCESS\t\tPRIORITY\tBURST TIME\tWAITING TIME\tTURNAROUND TIME");

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

printf("\n%d \t\t %d \t\t %d \t\t %d \t\t %d ",p[i],pri[i],bt[i],wt[i],tat[i]);

printf("\nAverage Waiting Time is --- %f",wtavg/n);

printf("\nAverage Turnaround Time is --- %f",tatavg/n);

getch();

}

**OBJECT 5: File management in Memory**

#include<stdio.h>

int main()

{

int i,j,n,a[50],frame[10],no,k,avail,count=0;

printf("\nenter the length of the Reference string:\n");

scanf("%d",&n);

printf("\n enter the reference string:\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\n");

}

printf("Page Fault Is %d",count);

return 0;

}

Object 6: Simulate all Page Replacement LRU