5. Simulate the Multiprogramming with a variable number of tasks (MVT)

Program:

#include<stdio.h>

main()

{

int mm,np,ps[100],rm[100],am=0,flag=0,i,j;

printf(" enter the memory size :");

scanf("%d",&mm);

printf(" enter no of processes:");

scanf("%d",&np);

for(i=0;(i<np)&&(am<mm);i++)

{

printf(" enter the size of p[%d]:",i+1);

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

am=am+ps[i];

if(am>=mm)

{

flag=1;

break;

}

rm[i]=mm-am;

}

j=i;

printf("process\tsize\trm\n");

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

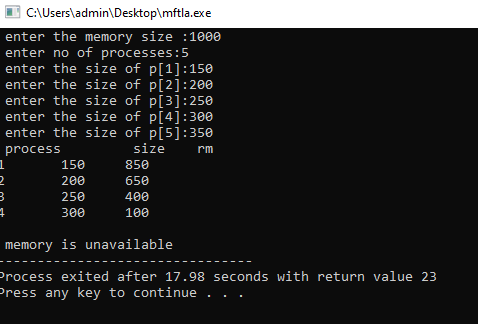
printf("%d\t%d\t%d\n",i+1,ps[i],rm[i]);

if(flag==1)

printf("\n memory is unavailable");

}

Output:



4. Simulate the Multiprogramming with a fixed number of tasks (MFT)

Program:

#include<stdio.h>

#include<math.h>

main()

{

int np,nb,mm,bs,i,j,ps[100],nba[100],ifm[100],sb=0,flag=0;

float x;

printf("enter the memory size");

scanf("%d",&mm);

printf("enter the number of blocks");

scanf("%d",&nb);

printf("enter the number of processes");

scanf("%d",&np);

bs=mm/nb;

for(i=1;(i<=np)&&(sb<nb);i++)

{

printf("enter the size of p[%d]:",i);

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

if (ps[i]<=bs)

nba[i]=1;

else

{

x=ps[i]/(float)bs;

nba[i]=(ceil)(x);

}

ifm[i]=nba[i]\*bs-ps[i];

sb=sb+nba[i];

if(sb>nb)

{

i=i-1;

flag=1;

}

}

j=1;

printf("process \t size \t nba \t ifm \n");

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

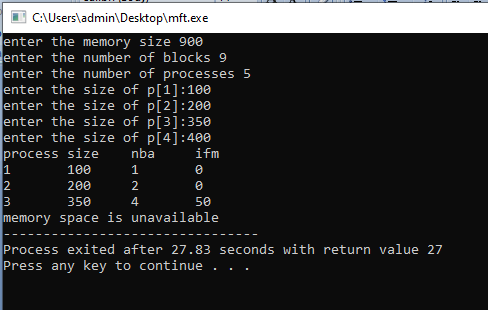
printf("%d\t%d\t%d\t%d\n",i,ps[i],nba[i],ifm[i]);

if (flag==1)

printf("memory space is unavailable");

}

Output:



9. Simulate the following File allocation strategies (a) Sequenced

Program:

#include<stdio.h>

main()

{

int n,i,j,size[20],sb[20],eb[20],t[20],x,c[20][20];

printf(" enter no of files:");

scanf("%d",&n);

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

{

printf(" enter no of blocks occupied by file %d",i+1);

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

printf(" enter the starting block of file %d",i+1);

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

t[i]=sb[i];

for(j=0;j<size[i];j++)

c[i][j]=sb[i]++;

}

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

{

eb[i]=t[i]+size[i]-1;

}

printf("filename\tlength\tstartblock\tendblock\n");

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

printf("%d\t%d\t%d\t%d\n",i+1,size[i],t[i],eb[i]);

printf(" \n enter filename:");

scanf("%d",&x);

printf("\n filename is %d ",x);

printf("\n length is %d",size[x-1]);

printf("\n blocks occupied:");

for(i=0;i<size[x-1];i++)

printf("%4d",c[x-1][i]);

}

Output:

