**Problem 10:** Write a program to implement the Best fit memory management algorithm. Program should take input total no. of memory block, their sizes, process name and process size. Output of program should give the details about memory allocated to process with fragmentation detail.

**Answer:**

*Source code:*

#include<stdio.h>

#include<stdlib.h>

typedef struct{

char process\_name[3];

int size,allocated;

}process;

typedef struct{

int size,fragment\_size;

}mem;

void algorithm(mem mem\_block[],int n, process pr[], int m){

int i,j,best\_block=-1;

for(i=0;i<m;i++){ // iterate in process array

best\_block=-1;

for(j=0;j<n;j++){

if(mem\_block[j].fragment\_size==0 && mem\_block[j].size>=pr[i].size){

if(best\_block==-1){

best\_block=j;

}

else if(mem\_block[best\_block].size>=mem\_block[j].size){

best\_block=j;

}

}

}

pr[i].allocated=best\_block;

mem\_block[best\_block].fragment\_size=mem\_block[best\_block].size-pr[i].size;

}

}

void print\_table(process pr[],int m, mem mem\_block[]){

int i,frag;

puts(" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

puts("| Process name | Size | Alloted | Fragment |");

puts("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_|");

for(i=0;i<m;i++){

if(pr[i].allocated==-1)

frag =-1;

else

frag=mem\_block[pr[i].allocated].fragment\_size;

printf("| %s | %3d | %2d | %3d |\n",

pr[i].process\_name,pr[i].size,pr[i].allocated,frag);

}

puts("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_|");

}

void main(){

int n,m,i,j;

printf("Enter total number of memory blocks\t");

scanf("%d",&n);

mem mem\_block[n];

printf("Enter the block sizes\n");

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

scanf("%d",&mem\_block[i].size);

mem\_block[i].fragment\_size=0;

}

printf("Enter total number of processes\t");

scanf("%d",&m);

process pr[m];

printf("Enter process details--> Process Name, Process Size.\n");

for(i=0;i<m;i++){

scanf("%s %d",pr[i].process\_name,&pr[i].size);

pr[i].allocated=-1;

}

algorithm(mem\_block,n,pr,m);

print\_table(pr,m,mem\_block);

}

*Output:*

