Code

```
#include<stdio.h>
#include<stdlib.h>

void initializeZero(int arr[][3],int proc[][2],int size,int number){
   for(int i=0;i<size;i++){
      arr[i][0]=0;
      arr[i][2]=0;
   }
   for(int j=0;j<number;j++){
      proc[0][1]=0;
   }
}

void firstfit(int arr[][3], int proc[][2], int size, int number){
  for(int i=0;i<number;i++){
        for(int j=0;j<size;j++){
              if(arr[j][1]>=proc[i][0]&& arr[j][2]==0){
                    printf("Process %d allocated in memory of size %d with internal
fragmentation as %d \n",i+1, arr[j][1],arr[j][1]-proc[i][0]);
                    arr[j][0]=proc[i][0];
                    arr[j][2]=1;
                    proc[i][1]=1;
                    break;
              }
        }
        if(proc[i][1]==0){
              printf("Process P%d cannot be allocated \n",i+1);
        }
  }
  printf("The memory alocation of various processes ofthe size are as shown: \n");
  for(int k=0;k<size;k++){
        if(arr[k][0]!=0)
              printf("\t %d \t ",arr[k][0]);
  }
}

void bestfit(int arr[][3], int proc[][2], int size, int number){
  int min = 0;
  for(int i=0;i<number;i++){
        for(int j=0;j<size;j++){
              if(arr[j][1]>=proc[i][0]&& arr[j][2]==0){
                 proc[i][1]=1;
                    if(arr[min][1]>arr[j][1] || arr[min][1]<=proc[i][0]){
                          min=j;
                    }
              }
        }
        if(proc[i][1]==0){
              printf("Process P%d cannot be allocated \n",i+1);
        }
```

```
        else{
                printf("Process P%d allocated in memory of size %d with internal fragmentation as
%d \n",i+1, arr[min][1],arr[min][1]-proc[i][0]);
                        arr[min][0]=proc[i][0];
                        arr[min][2]=1;
        }
        min=0;
  }
  printf("The memory alocation of various processes ofthe size are as shown: \n");
  for(int k=0;k<size;k++){
        if(arr[k][0]!=0)
                printf("\t %d \t",arr[k][0]);
  }
}

void worstfit(int arr[][3], int proc[][2], int size, int number){
  int max = 0;
  for(int i=0;i<number;i++){
        for(int j=0;j<size;j++){
                if(arr[j][1]>=proc[i][0]&& arr[j][2]==0){
                   proc[i][1]=1;
                        if(arr[max][1]<arr[j][1]){
                                max=j;
                        }
                }
        }
        if(proc[i][1]==0 || arr[max][1]<proc[i][0]){
                printf("Process P%d cannot be allocated \n",i+1);
        }
        else{
                printf("Process %d allocated in memory of size %d with internal fragmentation as
%d \n",i+1, arr[max][1],arr[max][1]-proc[i][0]);
                        arr[max][0]=proc[i][0];
                        arr[max][2]=1;
        }
        max=0;
  }
  printf("The memory alocation of various processes ofthe size are as shown: \n");
  for(int k=0;k<size;k++){
        if(arr[k][0]!=0)
                printf("\t %d \t",arr[k][0]);
  }
}


void main(){
  int size, number;
  printf("Enter the total number of memory blocks: ");
  scanf("%d",&size);
  int arr[size][3];
  printf("Enter the size of each block: \n");
  for(int i=0;i<size;i++){
```

```
        printf("For block %d : ",i+1);
        scanf("%d",&arr[i][1]);
        arr[i][2]=0;
        arr[i][0]=0;
 }
 printf("Number of processes to be allocated with memory:  ");
 scanf("%d",&number);
 int proc[number][2];
 printf("Enter the size of each process: \n");
 for(int i=0;i<number;i++){
        printf("For process P%d : ",i+1);
        scanf("%d",&proc[i][0]);
        proc[i][1]=0;
 }
 printf("\n FIRST FIT MEMORY ALLOCATION STRATEGY: \n");
 firstfit(arr, proc, size, number);
 initializeZero(arr, proc, size, number);
 printf("\n BEST FIT MEMORY ALLOCATION STRATEGY: \n");
 bestfit(arr, proc, size, number);
 initializeZero(arr, proc, size, number);
 printf("\n WORST FIT MEMORY ALLOCATION STRATEGY: \n");
 worstfit(arr, proc, size, number);

}
```

Output

```
Enter the total number of memory blocks: 4
Enter the size of each block:
For block 1 : 3000
For block 2 : 4600
For block 3 : 5000
For block 4 : 2110
Number of processes to be allocated with memory:  4
Enter the size of each process:
For process P1 : 2100
For process P2 : 4500
For process P3 : 3200
For process P4 : 7500

 FIRST FIT MEMORY ALLOCATION STRATEGY:
Process 1 allocated in memory of size 3000 with internal fragmentation as 900
Process 2 allocated in memory of size 4600 with internal fragmentation as 100
Process 3 allocated in memory of size 5000 with internal fragmentation as 1800
Process P4 cannot be allocated
The memory alocation of various processes ofthe size are as shown:
        2100            4500            3200
 BEST FIT MEMORY ALLOCATION STRATEGY:
Process P1 allocated in memory of size 2110 with internal fragmentation as 10
Process P2 allocated in memory of size 4600 with internal fragmentation as 100
Process P3 allocated in memory of size 5000 with internal fragmentation as 1800
Process P4 cannot be allocated
The memory alocation of various processes ofthe size are as shown:
        4500            3200            2100
 WORST FIT MEMORY ALLOCATION STRATEGY:
Process 1 allocated in memory of size 5000 with internal fragmentation as 2900
Process 2 allocated in memory of size 4600 with internal fragmentation as 100
Process P3 cannot be allocated
Process P4 cannot be allocated
The memory alocation of various processes ofthe size are as shown:
        4500            2100
```