Ex.No:6 **CONTIGUOUS MEMORY ALLOCATION**
Date :29.03.2023

AIM:

To write a 'C' program to perform contiguous memory allocation.

i. First fit
ii. Best fit
iii. Worst fit

ALGORITHM:

- Declare the process array and memory array globally.

i. **First fit** : Allocate the first hole that is large enough.
ii. **Best fit** : Allocate the small hole that is large enough.
iii. **Worst fit** : Allocate the largest hole that is large enough.

- Using the switch case perform the following algorithm.

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
int noofprocess,noofhole;
int holeold[10],holesize[10],processsize[10],holenumber[10];
void oldtonew();
//First fit way of allocating the memory to the process
void firstfit(){
int i,j;
printf("\n\t\tFirst Fit\n");
printf("\n\t\t=======================================================");
printf("\n\t\tProcess Psize Allocatedhole   Remaining hole size\n");
printf("\t\t=======================================================")
for(i=1;i<=noofprocess;i++)
  {

int c=0;
for(j=1;j<=noofhole;j++)
       {
if(processsize[i]<=holesize[j])
           {
printf("\n\t\tP%d\t%d\tH%d",i,processsize[i],j);
c=1;
holesize[j]-=processsize[i];
printf("            \t%d KB",holesize[j]);
break;
}
}
if(c==0){
printf("\n\t\tP%d\t%d\tNot allocated",i,processsize[i]);
}
}
}
//Best fit way of allocating memory to the process
```

```
void bestfit(){
int i,j;
int temp;
for(i=1;i<=noofhole;i++)
   {
for(j=i+1;j<=noofhole;j++)
      {
if(holesize[i]>holesize[j])
          {
//holes are sorted
temp=holesize[i];
holesize[i]=holesize[j];
holesize[j]=temp;


//hole number is sorted according to the size
temp=holenumber[i];
holenumber[i]=holenumber[j];
holenumber[j]=temp;
}
}
}
printf("\n\t\tBest Fit\n");
printf("\t\t=======================================================");
printf("\n\t\tProcess Psize Allocatedhole   Remaining hole size\n");
printf("\t\t=======================================================");

for(i=1;i<=noofprocess;i++){
int c=0;
for(j=1;j<=noofhole;j++){
if(processsize[i]<=holesize[j]){
printf("\n\t\tP%d\t%d\tH%d",i,processsize[i],holenumber[j]);
c=1;
holesize[j]-=processsize[i];
printf("            \t%d KB",holesize[j]);
break;
}
}
if(c==0)
printf("\n\t\tP%d\t%d\tNot allocated",i,processsize[i]);
}
}
//Worst fit way of allocating memory to the process
void worstfit(){
int i,j;
int temp;
for(i=1;i<=noofhole;i++)



   {
for(j=i+1;j<=noofhole;j++)
      {
if(holesize[i]<holesize[j])
          {
//holes are sorted
temp=holesize[i];
```

```
holesize[i]=holesize[j];
holesize[j]=temp;
//hole number is sorted according to the size
temp=holenumber[i];
holenumber[i]=holenumber[j];
holenumber[j]=temp;
}
}
}

printf("\n\t\tWorst Fit\n");
printf("\t\t=======================================================");
printf("\n\t\tProcess Psize Allocatedhole   Remaining hole size\n");
printf("\t\t=======================================================");

for(i=1;i<=noofprocess;i++){
int c=0;
for(j=1;j<=noofhole;j++){
if(processsize[i]<=holesize[j]){
printf("\n\t\tP%d\t%d\tH%d",i,processsize[i],holenumber[j]);
c=1;
holesize[j]-=processsize[i];
printf("            \t%d KB",holesize[j]);
break;
}
}


if(c==0)
printf("\n\t\tP%d\t%d\tNot allocated",i,processsize[i]);
}
}
void oldtonew()
{
int i;
    for(i=1;i<=noofhole;i++)
    {
        holesize[i]=holeold[i];
    }
}
int main(){
int i,j;
int ch;

printf("\n\t\t-------------------------------------------------------\n");
printf("\t\t\t\tMemory Allocation\n");
printf("\n\t\t-------------------------------------------------------\n");
printf("Enter the number of the memory holes: ");
scanf("%d",&noofhole);
printf("Enter the eack block hole size: \n");
for(i=1;i<=noofhole;i++){
printf("Hole %d: ",i);
scanf("%d",&holesize[i]);
holenumber[i]=i;
holeold[i]=holesize[i];
}
printf("\n\t\t-------------------------------------------------------\n");
```

```
printf("Enter the number of process: ");
scanf("%d",&noofprocess);
printf("Enter the size of each process:\n");


for(i=1;i<=noofprocess;i++){
printf("Process %d: ",i);
scanf("%d",&processsize[i]);
}
int n=1;
while(n==1)
   {
printf("\n\t\t-----------------------------------------------------\n");

printf("\nMenu for allocation:\n");
printf("\n1.First Fit \n2.Best Fit \n3.Worst Fit");
printf("\nEnter your choice: ");
scanf("%d",&ch);
printf("\n\t\t-----------------------------------------------------\n");

switch(ch)
   {
case 1:
firstfit();
break;
case 2:
          oldtonew();
bestfit();
break;
case 3:
          oldtonew();
          worstfit();
          break;
default:
printf("Wrong choice!!");
}
}

getch();
return 0;
}
```

OUTPUT:

```
Menu for allocation:

1.First Fit
2.Best Fit
3.Worst Fit
Enter your choice: 3

                Worst Fit
                =========================================================
                Process Psize Allocatedhole   Remaining hole size
                =========================================================
                P1      418     H5              182 KB
                P2      202     H2              298 KB
                P3      506     Not allocated
                P4      112     H5              70 KB
                P5      95      H2              203 KB
```

```
Menu for allocation:

1.First Fit
2.Best Fit
3.Worst Fit
Enter your choice: 2

                Best Fit
                =========================================================
                Process Psize Allocatedhole   Remaining hole size
                =========================================================
                P1      418     H1              82 KB
                P2      202     H3              98 KB
                P3      506     H5              94 KB
                P4      112     H4              88 KB
                P5      95      H2              5 KB
```

```
Menu for allocation:

1.First Fit
2.Best Fit
3.Worst Fit
Enter your choice: 1

                First Fit

                =========================================================
                Process Psize Allocatedhole   Remaining hole size
                =========================================================
                P1      418     H1              82 KB
                P2      202     H3              98 KB
                P3      506     H5              94 KB
                P4      112     H4              88 KB
                P5      95      H2              5 KB
```

```
                                Memory Allocation
Enter the number of the memory holes: 5
Enter the eack block hole size:
Hole 1: 500
Hole 2: 100
Hole 3: 300
Hole 4: 200
Hole 5: 600
Enter the number of process: 5
Enter the size of each process:
Process 1: 418
Process 2: 202
Process 3: 506
Process 4: 112
Process 5: 95
```

Observation	
Record	
Total	
Initial	

RESULT:

The programs for contiguous memory allocation are executed successfully and the outputs are verified.