Lab Report: 07

Theory: The operating system must keep list of each memory location noting which are free and which are busy. Then as new jobs come into the system, the free partitions must be allocated.

These partitions may be allocated by 4 ways:

1. First-Fit Memory Allocation
2. Best-Fit Memory Allocation
3. Worst-Fit Memory Allocation

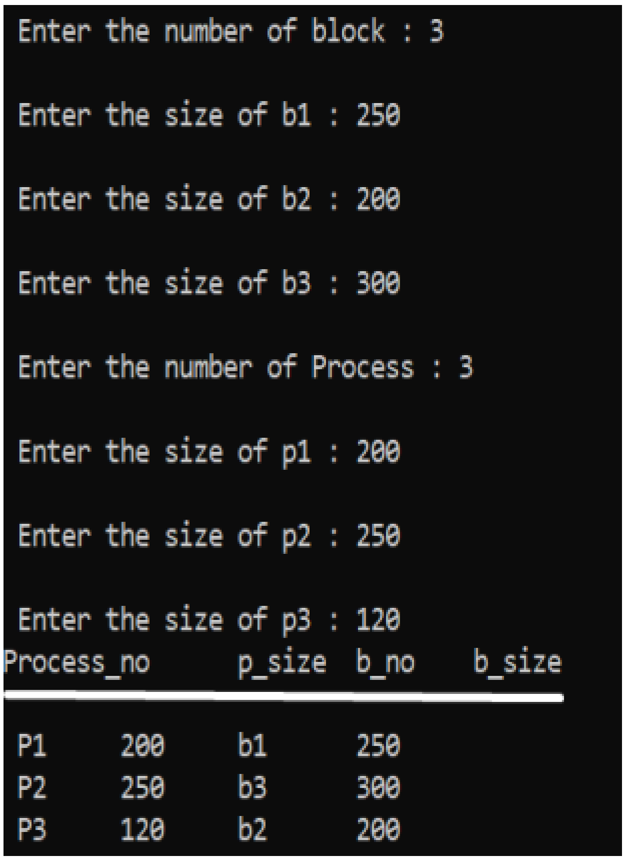
First-Fit Memory Allocation:
This method keeps the free/busy list of jobs organized by memory location, low-ordered to high-ordered memory. In this method, first job claims the first available memory with space more than or equal to it's size. The operating system doesn't search for appropriate partition but just allocate the job to the nearest memory partition available with sufficient size.

Code:

```
#include<stdio.h>
int main()
{
    int p_no,b_no,b_size[10],p_size[10],b_check[10],track[10],i,j;
    printf("\n Enter the number of block : ");
    scanf("%d",&b_no);
    for(i=0;i<b_no;i++)
    {
        printf("\n Enter the size of b%d : ",i+1);
        scanf("%d",&b_size[i]);
    }
    printf("\n Enter the number of Process : ");
    scanf("%d",&p_no);
    for(i=0;i<p_no;i++)
```

```
{
    printf("\n Enter the size of p%d : ",i+1);
    scanf("%d",&p_size[i]);
}
for(i=0;i<b_no;i++)
{
    b_check[i]=-1;
}
for(i=0;i<p_no;i++)
{
    track[i]=-1;
}
for(i=0;i<p_no;i++)
{
   for(j=0;j<b_no;j++)
   {
      if(b_check[j]==-1 && p_size[i]<=b_size[j])
      {
         b_check[j]=i;
         track[i]=j;
         break;
      }
   }
}
printf("Process_no\tp_size\tb_no\tb_size");
printf("\n\n");
for(i=0;i<p_no;i++)
{
        printf(" P%d\t%d",i+1,p_size[i]);
        if(track[i]==-1)
        {
           printf("\t Not allocated...");
        }
        else
        printf("\tb%d\t%d",track[i]+1,b_size[track[i]]);
        printf("\n");
    }
}
```

Input and Output:



```
Enter the number of block : 3

Enter the size of b1 : 250

Enter the size of b2 : 200

Enter the size of b3 : 300

Enter the number of Process : 3

Enter the size of p1 : 200

Enter the size of p2 : 250

Enter the size of p3 : 120
Process_no      p_size  b_no    b_size

 P1     200     b1      250
 P2     250     b3      300
 P3     120     b2      200
```

Best-Fit Memory Allocation:

This method keeps the free/busy list in order by size – smallest to largest. In this method, the operating system first searches the whole of the memory according to the size of the given job and allocates it to the closest-fitting free partition in the memory, making it able to use memory efficiently. Here the jobs are in the order from smallest job to largest job.

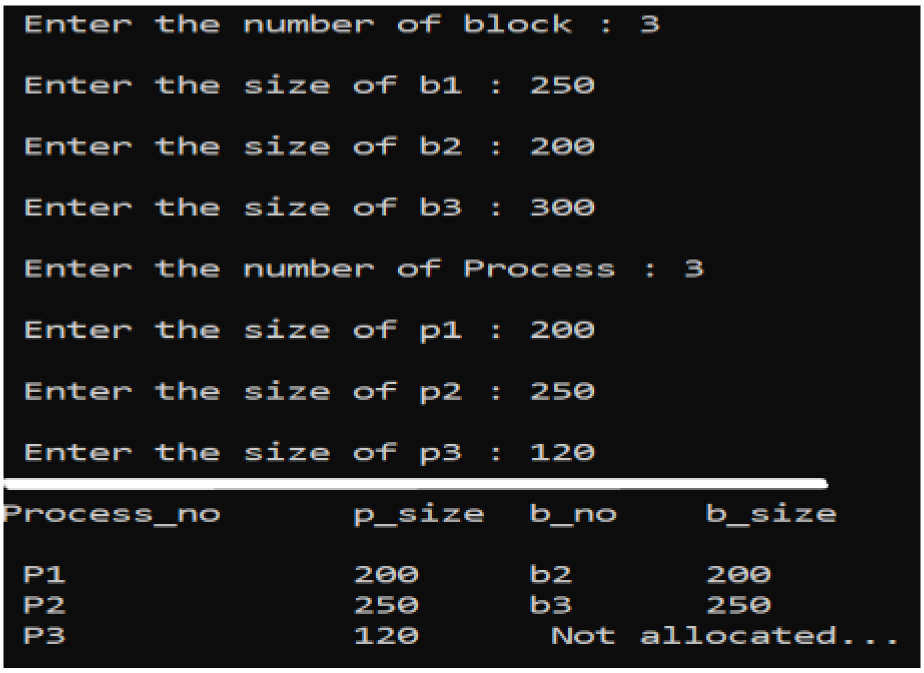
Code:

```
#include<stdio.h>
int main()
{
    int p_no,b_no,b_size[10],p_size[10],b_check[10],track[10],i,j,temp;
    printf("\n Enter the number of block : ");
    scanf("%d",&b_no);
    for(i=0;i<b_no;i++)
    {
        printf("\n Enter the size of b%d : ",i+1);
        scanf("%d",&b_size[i]);
    }
    printf("\n Enter the number of Process : ");
    scanf("%d",&p_no);
    for(i=0;i<p_no;i++)
    {
        printf("\n Enter the size of p%d : ",i+1);
        scanf("%d",&p_size[i]);
    }
    for(i=0;i<b_no;i++)
    {
        b_check[i]=-1;
    }
```

```
for(i=0;i<p_no;i++)
{
    track[i]=-1;
}
for(i=0; i<p_no; i++){
   for(j=0; j<b_no; j++){
      if(b_size[j] > b_size[j+1])
      {
         temp = b_size[j];
         b_size[j]=b_size[j+1];
         b_size[j+1]=temp;
      }
   }
}
for(i=0;i<p_no;i++)
{
   for(j=0;j<b_no;j++)
   {
      if(b_check[j]==-1 && p_size[i]<=b_size[j])
      {
         b_check[j]=i;
         track[i]=j;
         break;
      }
   }
}
printf("Process_no\tp_size\tb_no\tb_size");
printf("\n\n");
```

```
    for(i=0;i<p_no;i++)
    {
        printf(" P%d\t%d",i+1,p_size[i]);
        if(track[i]==-1)
        {
            printf("\t Not allocated...");
        }
        else
        printf("\tb%d\t%d",track[i]+1,b_size[track[i]]);
        printf("\n");
    }
}
```

Input and Output:



```
 Enter the number of block : 3

 Enter the size of b1 : 250

 Enter the size of b2 : 200

 Enter the size of b3 : 300

 Enter the number of Process : 3

 Enter the size of p1 : 200

 Enter the size of p2 : 250

 Enter the size of p3 : 120

Process_no      p_size  b_no    b_size

 P1             200     b2      200
 P2             250     b3      250
 P3             120      Not allocated...
```

<u>Worst-Fit Memory Allocation</u>:
In this allocation technique, the process traverses the whole memory and always search for the largest hole/partition, and then the process is placed in that hole/partition. It is a slow process because it has to traverse the entire memory to search the largest hole.

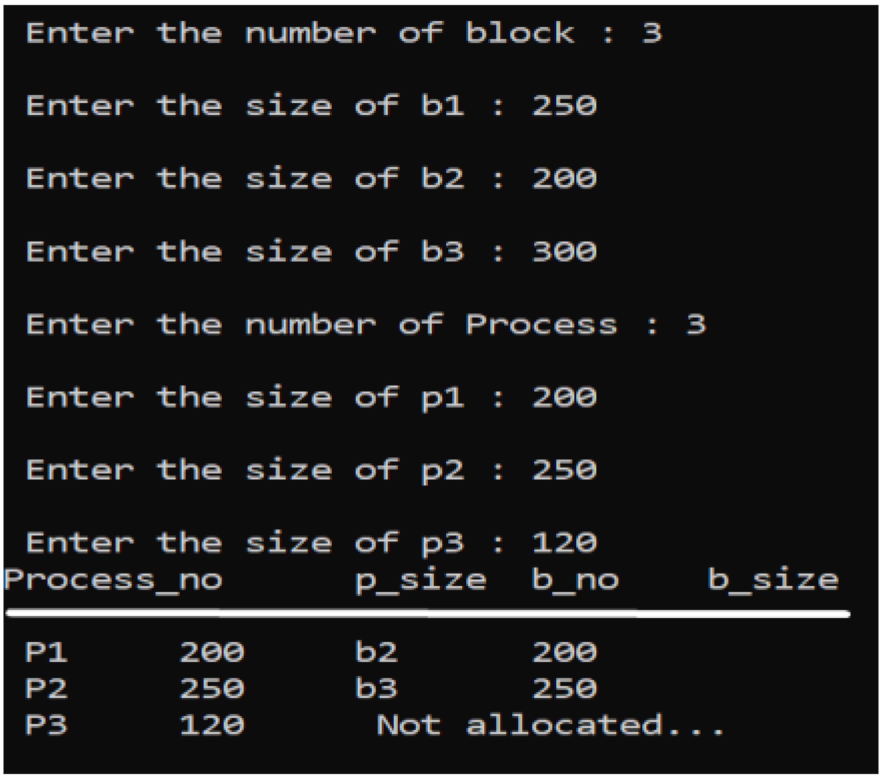
<u>Code:</u>

```
#include<stdio.h>
int main()
{
    int p_no,b_no,b_size[10],p_size[10],b_check[10],track[10],i,j,temp;
    printf("\n Enter the number of block : ");
    scanf("%d",&b_no);
    for(i=0;i<b_no;i++)
    {
         printf("\n Enter the size of b%d : ",i+1);
         scanf("%d",&b_size[i]);
    }
    printf("\n Enter the number of Process : ");
    scanf("%d",&p_no);
    for(i=0;i<p_no;i++)
    {
         printf("\n Enter the size of p%d : ",i+1);
         scanf("%d",&p_size[i]);
    }
    for(i=0;i<b_no;i++)
    {
         b_check[i]=-1;
    }
```

```
for(i=0;i<p_no;i++)
{
    track[i]=-1;
}
for(i=0; i<p_no; i++){
   for(j=0; j<b_no; j++){
      if(b_size[j] < b_size[j+1])
      {
         temp = b_size[j];
         b_size[j]=b_size[j+1];
         b_size[j+1]=temp;
      }
   }
}
for(i=0;i<p_no;i++)
{
   for(j=0;j<b_no;j++)
   {
      if(b_check[j]==-1 && p_size[i]<=b_size[j])
      {
         b_check[j]=i;
         track[i]=j;
         break;
      }
   }
}
printf("\n\n");
printf("Process_no\tp_size\tb_no\tb_size");
```

```
    printf("\n\n");
    for(i=0;i<p_no;i++)
    {
        printf(" P%d\t%d",i+1,p_size[i]);
        if(track[i]==-1)
        {
            printf("\tNot allocated...");
        }
        else
        printf("\tb%d\t%d",track[i]+1,b_size[track[i]]);
        printf("\n");
    }
}
```

Input and Output:



```
 Enter the number of block : 3

 Enter the size of b1 : 250

 Enter the size of b2 : 200

 Enter the size of b3 : 300

 Enter the number of Process : 3

 Enter the size of p1 : 200

 Enter the size of p2 : 250

 Enter the size of p3 : 120
Process_no      p_size  b_no    b_size
______________________________________
 P1      200     b2      200
 P2      250     b3      250
 P3      120      Not allocated...
```