PAGE NO :
DATE : 16/8/23

Write a C program to simulate the following contigious memory allocation technique
a) worst - fit .
b) Best - fit
c) First fit

a) . worst - fit

```
#l include <Stdio. h>
  Void worstfit (int blocksize[], int blocks, int processize[]
     int processes)
{   int allocation [processes];
     int occupied [blocks];
      for (int i=0 ; i<processes ; i++){
                allocation[i]= -1;
      }
   for (int i=0 ; i< blocks ; i++) {
          occupied [i]=0;
      }
   for (int i=0 ; i< processes; i++){
        int   index placed = -1;
      for (int j=0; j< blocks ; j++) {
        if (block size [j] >= process size [i] &&
                !occupied [j])
         {
           if (index placed == -1)
              index placed = j;
            else if (block size [index placed] < block
                                    size [j])
                    index placed = j;
      }
   }
```

PAGE NO :
DATE :

```
if (indexplaced != -1){
    allocation [i] = indexplaced;
    Occupied [indexplaced] = 1;
    blocksize [indexplaced] -= processSize [i];
}}
printf ("\n process No. \t process Size\t Block no.
                                        \n");
 for (int i=0 ; i< processes ; i++){
    printf ("%d \t\t \t %d \t\t\t", i+1,
          processSize (i]);
  if (allocation [i]! = -1)
        Printf ("%d\n", allocation [i] +1);
  else
     Printf(" Not  allocated \n");
   }}
int main () {
     int i, blocks, processes;
    printf ("Enter the no. of blocks: ");
    Scanf ("%d", & blocks);
     int blocksize [blocks];
      Printf(" \n Enter size of each block: ");
       for (int i=0 ; i< blocks ; i++)
       Scanf ( "%d", & processes);
          int processsize [processes];
      Printf ("\n Enter size of each process: ");
          for (i=0; i<processes ; i++)
         Scanf ("%d", & process Size [i]);
       worst fit (blocksize, blocks, processsize, processes);
           return 0;
}
```

b)

Output:

```
Enter no. of block : 3
Enter size of each block : 5   2   7
Enter no. of process : 2
Enter size of each process : 1   4.
process no.          process size          Block no.
   1                      1                     3
   2                      4                     1
```

```
# include <stdio.h>
# define MAX 10
Void Best fit ( int blocksize [], int blocks, int
        process size[], int processes, int m){
        int allocation [processes];
        int occupied [blocks];
        for (int i=0 ; i<processes ; i++){
            allocation [i] = -1;
    }
    for (int i=0 ; i<blocks ; i++){
            occupied [i]=0;
    }
    for (int i=0 ; i< processes ; i++){
        int indexplaced = -1;
        for (int j=0 ; j<blocks ; j++){
            if [ blocksize [j] >= process size[i] &&
                !occupied [j]] {
                if (indexplaced == -1)
                    index placed = j;
                else if (blocksize (j)< block size [indexplaced])
                        indexplaced = j,
            }
        }
```

PAGE NO :
DATE :

```
if (indexplaced != -1)
{
   allocation [i] = index placed;
    Occupied [index placed] = 1;
   }
  }
printf ("\n process No \t processSize \t Blockno.\n");
    for (int i=0; i<processes; i++){
      printf("%d \t\t\t %d \t\t", i+1, processSize
                                              [i]);
    if (allocation [i] != -1)
        printf ("%d \n" allocation [i]+1);
      else
        printf ("not allocated \n");
    }
  }
int main()
  int p, m, j;
    printf (" Enter the no of process and block:");
    Scanf ("%d %d", &p, &m);
     int processSize [p], blockSize [m];
      printf ("Enter the process sizes:");
       for (j=0; j<p; j++)
            Scanf ("%d", & processSize[j]);
    printf (" Enter the blockSizes : ");
       for (j=0; j<m; j++)
          Scanf ("%d", & blocksize[j]);
     int blocks = Size of (blocksize) / Size of (block
                                             size[i]);
     int process = Size of (processSize) / Size of (process
                                             size[i]);

     BestFit (blocksize, block, processSize, processes);
          return 0;
   }
```

PAGE NO :
DATE :

```
Enter the number of process and blocks : 2  3
Enter the process sizes : 1    4
Enter the block size : 5    2  7

Process No          process size          Block no.
1                   1                     2
2                   4                     1
```

c)
```
# include <stdio.h>
# include <conio.h>
# define max 25

Void main() {
    int frag[max], b[max], f[max], i, j, nb, nf, temp,
        static int bf[max], ff[max];

printf ("\n\tMemory management scheme - First fit");
  printf ("\nEnter the no of blocks:");
   Scanf ("%d", &nb);
   printf ("Enter the no of files:");
    Scanf ("%d", &nf);

  Printf ("\nEnter the size of the block :-\n");
      for (i=1 ; i <= nb ; i++){
            printf ("Block %d: ", i);
           Scanf ("%d", &b[i]);
           }
       printf ("Enter the size of the files:-\n");
      for (i=1 ; i<nf ; i++) {
            Printf ("File %d: ", i);
              Scanf ("%d", &f[i]);
              }
```

```
printf ("\nFile_no : \t File-Size :\t Block_no:\t
        Block -Size : ");
for (i=1 ; i<=nf ; i++) {
    int  allocated =0;
     foo (j=1 ; j<=nb ; j++){
        if (bf[j]!=1) {
        temp = b[j] - f[i]
         if (temp>=0) {
           ff[i]=j ;
             bf[j]=1;
           frag [i]= b[j] - f[i];
           allocated =1;
         printf ("\n%d\t\t %d \t \t %d \t %d",
              i, f[i], ff[i], b[ff[i]]);
         break;
      }}
    if (!allocated) {
        printf ("\n%d \t \t %d \t \t N
             Allocated \t \t", i, f[i]);
      } }
      getch();
  }
```

Output

Memory Management Scheme - First Fit.

```
Enter the no of block : 3
Enter the no of files : 2
Enter the Size of the blocks:
  Block 1 : 5
  Block 2 : 2
  Block 3 : 7
Enter the Size of the files:
    File 1 : 1
    File 2 : 4
```

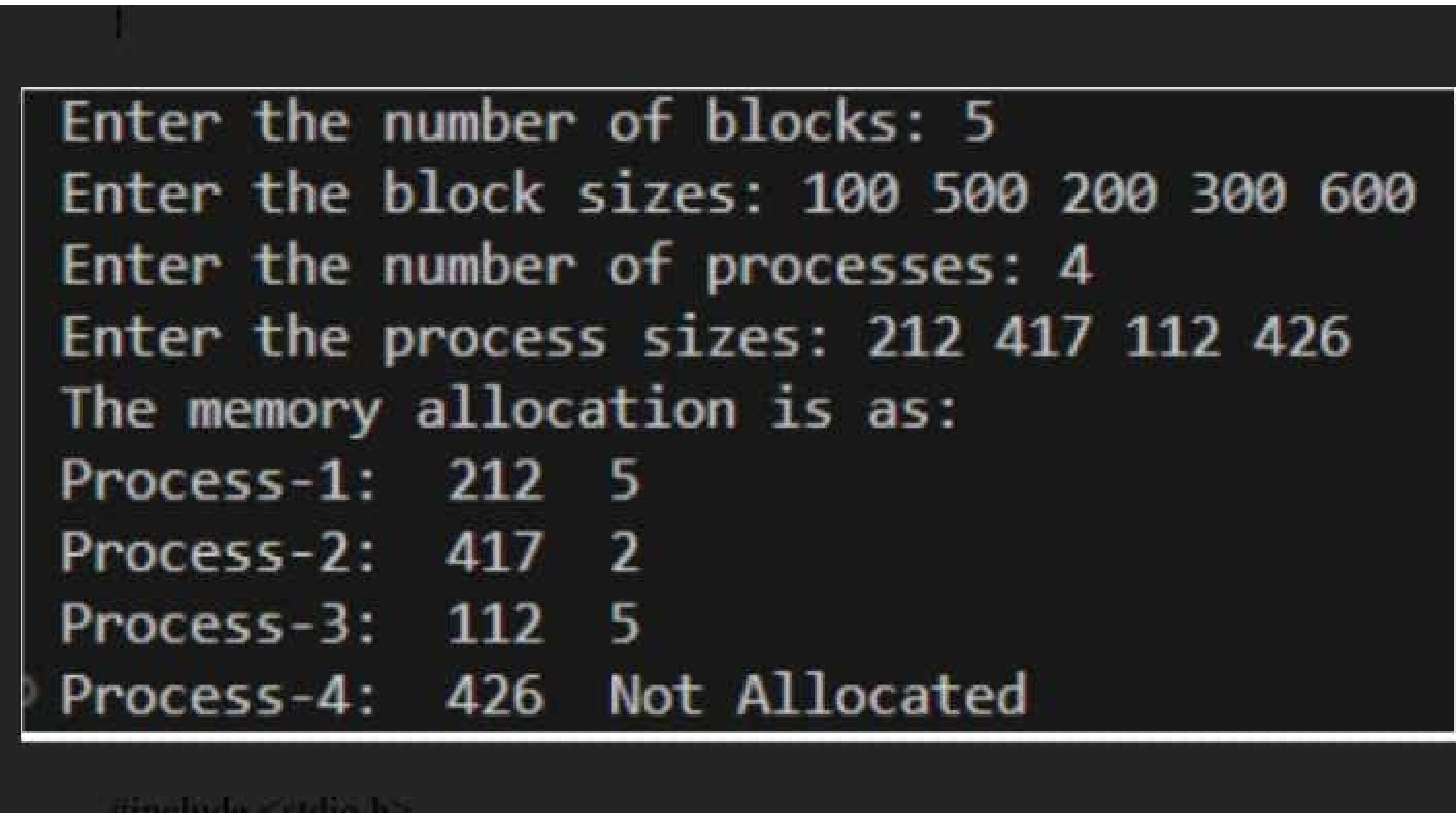
PAGE NO :
DATE :

File_no	File_size	Block_no	Block size
1	1	1	5
2	4	3	7

C

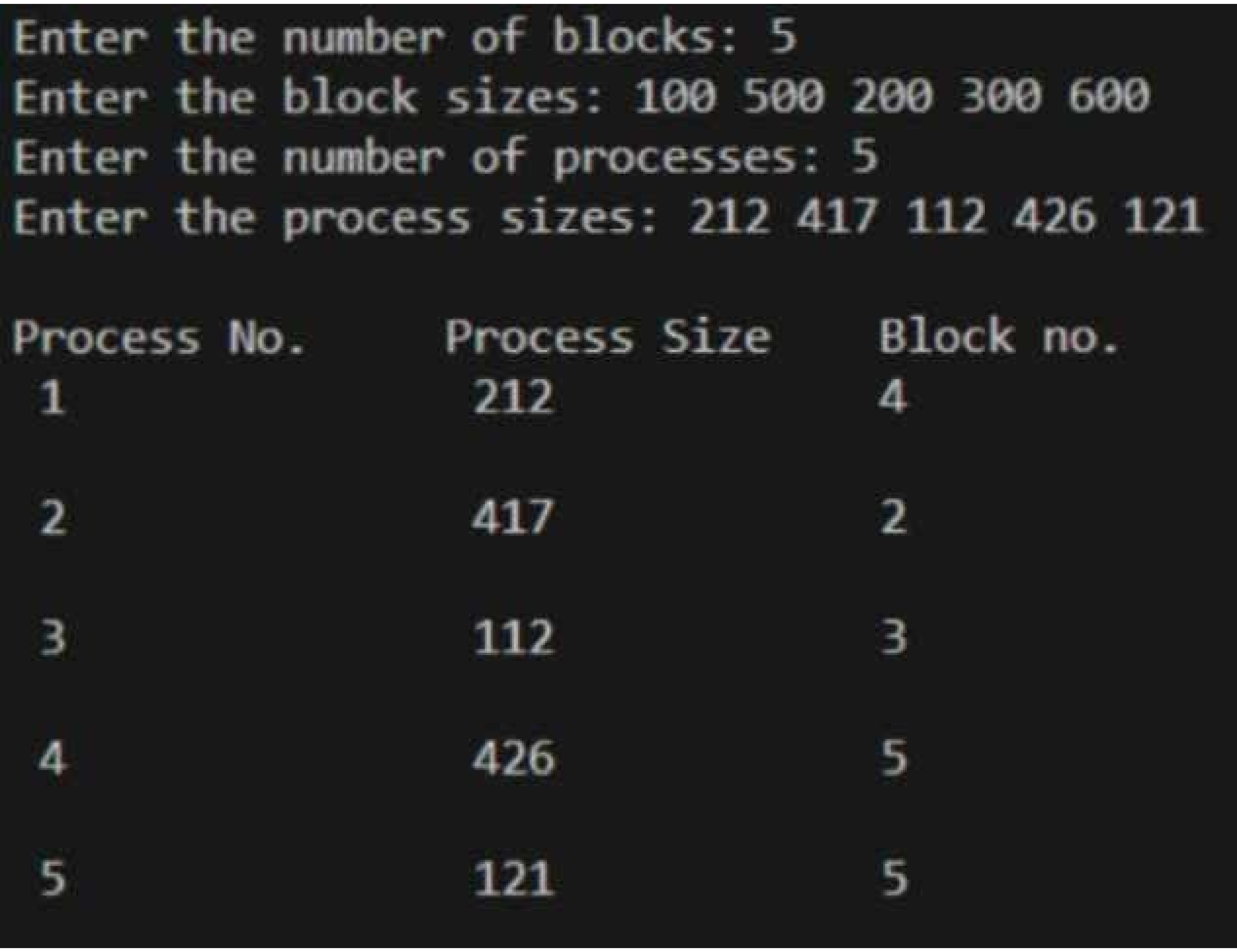
10/10

N
23/8/23

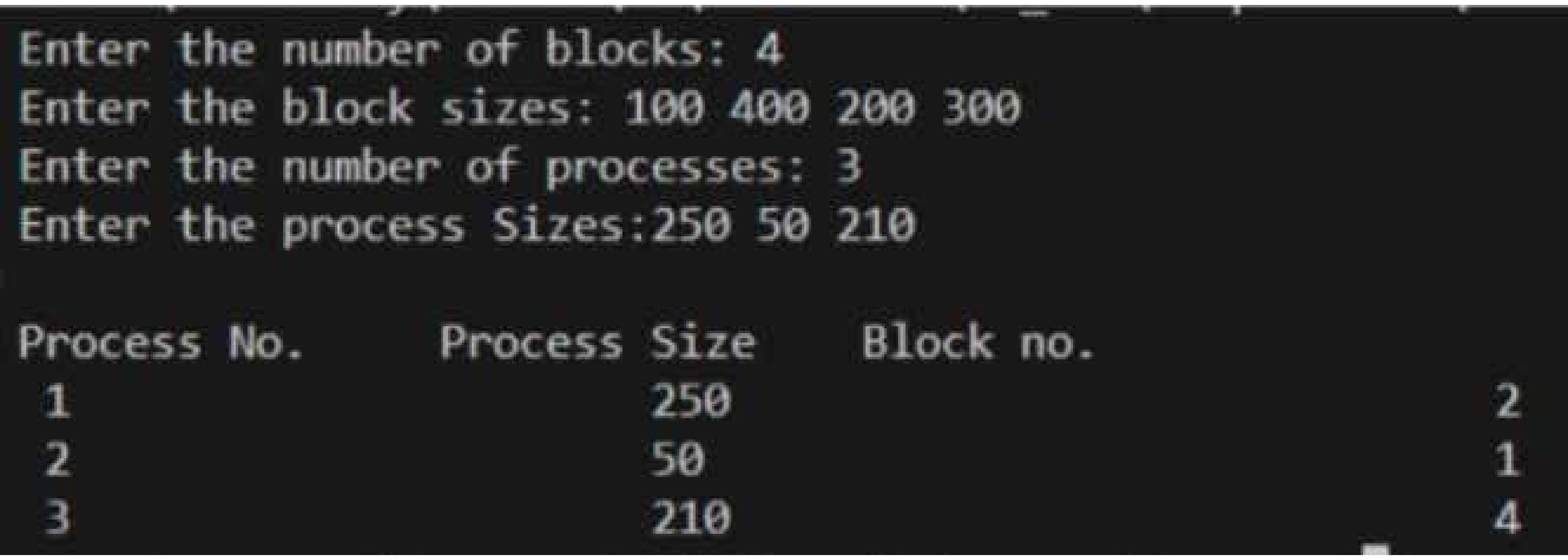


```
}
```

```
Enter the number of blocks: 5
Enter the block sizes: 100 500 200 300 600
Enter the number of processes: 4
Enter the process sizes: 212 417 112 426
The memory allocation is as:
Process-1:  212  5
Process-2:  417  2
Process-3:  112  5
Process-4:  426  Not Allocated
```



```
Enter the number of blocks: 5
Enter the block sizes: 100 500 200 300 600
Enter the number of processes: 5
Enter the process sizes: 212 417 112 426 121

Process No.     Process Size    Block no.
 1               212             4

 2               417             2

 3               112             3

 4               426             5

 5               121             5
```



```
Enter the number of blocks: 4
Enter the block sizes: 100 400 200 300
Enter the number of processes: 3
Enter the process Sizes:250 50 210

Process No.      Process Size     Block no.
 1                    250                                                 2
 2                    50                                                  1
 3                    210                                                 4
```