6.Develop a C program to simulate the following contiguous memory allocation Techniques: a) Worst fit b) Best fit c) First fit.

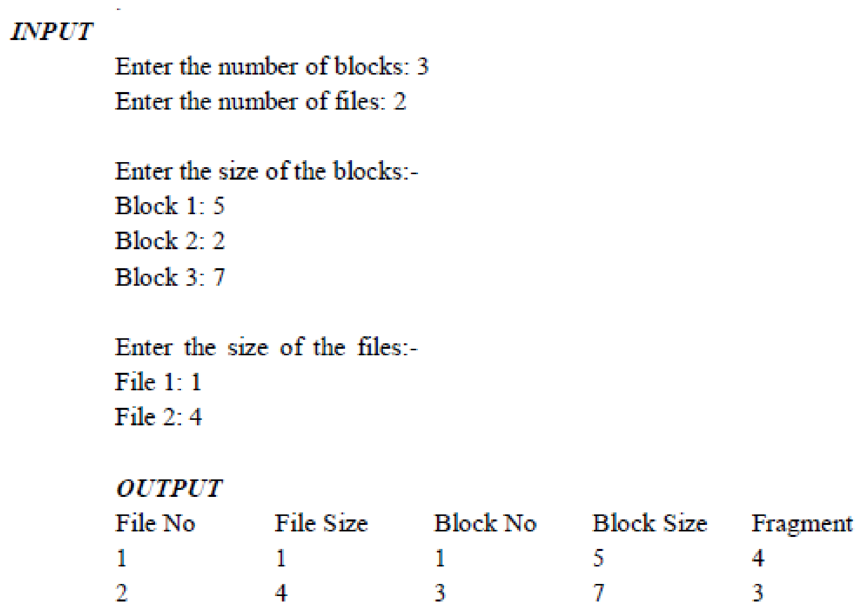
a) WORST-FIT

PROGRAM

```
#include<stdio.h>
#define max 25

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

   printf("\n\tMemory Management Scheme - Worst Fit\n");
   printf("Enter the number of blocks:");
   scanf("%d", &nb);
   printf("Enter the number of files:");
   scanf("%d", &nf);
   printf("\nEnter the size of the blocks:-\n");
   for (i = 0; i < nb; i++) {
      printf("Block %d:", i + 1);
      scanf("%d", &b[i]);
   }
   printf("Enter the size of the files :-\n");
   for (i = 0; i < nf; i++) {
      printf("File %d:", i + 1);
      scanf("%d", &f[i]);
   }

   for (i = 0; i < nf; i++) {
      int index = -1; // Use -1 to indicate no block has been found yet
      int maxFrag = -1; // Initialize maxFrag to -1 to find the worst fit
      for (j = 0; j < nb; j++) {
         if (bf[j] != 1) {
            temp = b[j] - f[i];
            if (temp >= 0 && temp > maxFrag) { // Check if it's a worse fit
               maxFrag = temp;
               index = j;
            }
         }
      }
      if (index != -1) { // If a block was found
         ff[i] = index;
         frag[i] = maxFrag;
         bf[index] = 1; // Mark this block as filled
      } else {
         // If no suitable block is found, you could set ff[i] and frag[i] to indicate failure
      }
   }
```

```
    printf("\nFile_no:\tFile_size:\tBlock_no:\tBlock_size:\tFragment");
    for (i = 0; i < nf; i++) {
        if (ff[i] != -1) // Check if file was allocated
            printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d", i + 1, f[i], ff[i] + 1, b[ff[i]], frag[i]);
        else
            printf("\n%d\t\t%d\t\tNot Allocated", i + 1, f[i]);
    }

    return 0;
}
```

OUTPUT:



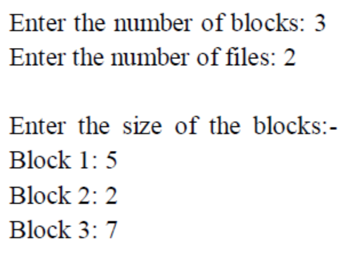
```
INPUT
            Enter the number of blocks: 3
            Enter the number of files: 2

            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

            OUTPUT
            File No     File Size    Block No    Block Size    Fragment
            1           1            1           5             4
            2           4            3           7             3
```

b) BEST-FIT

PROGRAM

```
    #include<stdio.h>
    #define max 25

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

        printf("\n\tMemory Management Scheme - Best Fit\n");
        printf("Enter the number of blocks:");
        scanf("%d", &nb);
        printf("Enter the number of files:");
        scanf("%d", &nf);
    printf("\nEnter the size of the blocks:-\n");
    for (i = 0; i < nb; i++) {
        printf("Block %d:", i + 1);
        scanf("%d", &b[i]);
    }
    printf("Enter the size of the files :-\n");
    for (i = 0; i < nf; i++) {
        printf("File %d:", i + 1);
        scanf("%d", &f[i]);
    }

    for (i = 0; i < nf; i++) {
        int index = -1; // Use -1 to indicate no suitable block has been found yet
        int minFrag = 1e9; // Initialize minFrag to a large number to find the best fit
        for (j = 0; j < nb; j++) {
            if (bf[j] != 1) {
                temp = b[j] - f[i];
                if (temp >= 0 && temp < minFrag) { // Check if it's a better fit
                    minFrag = temp;
                    index = j;
                }
            }
        }
        if (index != -1) { // If a suitable block is found
            ff[i] = index;
            frag[i] = minFrag;
            bf[index] = 1; // Mark this block as filled
        } else {
            // If no suitable block is found, you might want to indicate this differently
            // For example, setting ff[i] to -1 (or another sentinel value) to signify
allocation failure
        }
    }

    printf("\nFile_no:\tFile_size:\tBlock_no:\tBlock_size:\tFragment");
    for (i = 0; i < nf; i++) {
        if (ff[i] != -1) // Check if file was allocated
            printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d", i + 1, f[i], ff[i] + 1, b[ff[i]], frag[i]);
        else
            printf("\n%d\t\t%d\t\tNot Allocated", i + 1, f[i]);
    }

    return 0;
}
```

OUTPUT:



```
Enter the number of blocks: 3
Enter the number of files: 2

Enter the size of the blocks:-
Block 1: 5
Block 2: 2
Block 3: 7
```

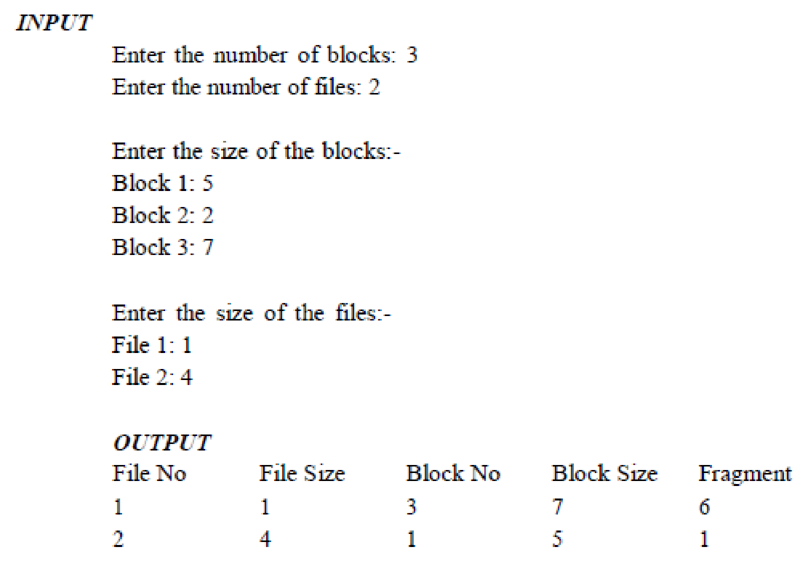
c)   FIRST-FIT

PROGRAM

```
#include<stdio.h>
#define max 25

int 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 number of blocks:");
   scanf("%d", &nb);
   printf("Enter the number of files:");
   scanf("%d", &nf);
   printf("\nEnter the size of the blocks:-\n");
   for (i = 0; i < nb; i++) {
      printf("Block %d:", i + 1);
      scanf("%d", &b[i]);
   }
   printf("Enter the size of the files :-\n");
   for (i = 0; i < nf; i++) {
      printf("File %d:", i + 1);
      scanf("%d", &f[i]);
   }

   for (i = 0; i < nf; i++) {
      for (j = 0; j < nb; j++) {
         if (bf[j] == 0) { // if block[j] is not allocated
            temp = b[j] - f[i];
            if (temp >= 0) { // if file fits in block
               ff[i] = j; // allocate block j to file i
```

```
                bf[j] = 1; // mark block as allocated
                frag[i] = temp; // fragmentation for this allocation
                break; // exit loop after first fit found
            }
        }
    }
}

printf("\nFile_no:\tFile_size:\tBlock_no:\tBlock_size:\tFragment");
for (i = 0; i < nf; i++) {
    if (bf[ff[i]] == 1) { // If the file got a block
        printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d", i + 1, f[i], ff[i] + 1, b[ff[i]], frag[i]);
    } else { // If the file didn't get a block
        printf("\n%d\t\t%d\t\tNot Allocated", i + 1, f[i]);
    }
}

return 0;
}
```

<u>OUTPUT:</u>



INPUT

```
Enter the number of blocks: 3
Enter the number 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
```

OUTPUT

```
File No     File Size     Block No     Block Size     Fragment
1           1             3            7              6
2           4             1            5              1
```

<u>Viav:</u>

1. How will you implement Next Fit Strategy in your code?
2. Write a program for process simulation and memory release?

Extra:

b[max]	Stores sizes of memory blocks
f[max]	Stores sizes of files/processes
nb	Number of memory blocks
nf	Number of files
bf[max]	Block status (0 = free, 1 = allocated)
ff[max]	Stores allocated block index for each file
frag[max]	Stores internal fragmentation
temp	Temporary variable to calculate block–file difference