Ex. No.: 10a) **231901039**

Date: 5/4/2025

BEST FIT

Aim:

To implement Best Fit memory allocation technique using C.

Algorithm:

1. Input memory blocks and processes with sizes
2. Initialize all memory blocks as free.
3. Start by picking each process and find the minimum block size that can be assigned to current process
4. If found then assign it to the current process.
5. If not found then leave that process and keep checking the further processes.

Program Code:

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

int main() {
    int blockSize[MAX], processSize[MAX], blockAllocated[MAX];
    int blockCount, processCount;
    int i, j, bestIdx;
    printf("Enter the number of memory blocks: ");
    scanf("%d", &blockCount);
    printf("Enter the size of each memory block:\n");
    for (i = 0; i < blockCount; i++) {
        printf("Block %d: ", i + 1);
        scanf("%d", &blockSize[i]);
        blockAllocated[i] = 0;//Initially all blocks are free
    }
    printf("\nEnter the number of processes: ");
    scanf("%d", &processCount);
    printf("Enter the size of each process:\n");
    for (i = 0; i < processCount; i++) {
        printf("Process %d: ", i + 1);
        scanf("%d", &processSize[i]);
    }
    int allocation[MAX];
    for (i = 0; i < processCount; i++) {
        bestIdx = -1;
        for (j = 0; j < blockCount; j++) {
            if (!blockAllocated[j] && blockSize[j] >=
processSize[i]) {
```

```
                if (bestIdx == -1 || blockSize[j] <
blockSize[bestIdx]) {
                    bestIdx = j;
                }
            }
        }
        if (bestIdx != -1) {
            allocation[i] = bestIdx;
            blockAllocated[bestIdx] = 1;
        } else {
            allocation[i] = -1;
        }
    }
    printf("\nProcess No\tProcess Size\tBlock No\n");
    for (i = 0; i < processCount; i++) {
        printf("%d\t\t%d\t\t", i + 1, processSize[i]);
        if (allocation[i] != -1)
            printf("%d\n", allocation[i] + 1);
        else
            printf("Not Allocated\n");
    }

    return 0;
}
```

Sample Output:

Process No.	Process	Size	Block no.
1	212		4
2	417		2
3	112		3
4	426		5

Result:

Best Fit memory allocation technique was implemented using C.

Ex. No.: 10b) **231901054**

Date: 5/4/2025

FIRST FIT

Aim:

To write a C program for implementation memory allocation methods for fixed partition using first fit.

Algorithm:

1. Define the max as 25.

2: Declare the variable frag[max],b[max],f[max],i,j,nb,nf,temp, highest=0, bf[max],ff[max]. 3:

Get the number of blocks,files,size of the blocks using for loop.

4: In for loop check bf[j]!=1, if so temp=b[j]-f[i]

5: Check highest

Program Code:

```
#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("Enter the number of blocks: ");
   scanf("%d", &nb);
   printf("Enter the number of files: ");
   scanf("%d", &nf);

   printf("\nEnter the size of each block:\n");
   for (i = 0; i < nb; i++) {
       printf("Block %d: ", i + 1);
       scanf("%d", &b[i]);
   }
   printf("\nEnter the size of each file:\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] != 1 && b[j] >= f[i]) {
```

```
                ff[i] = j;
                frag[i] = b[j] - f[i];
                bf[j] = 1;
                break;
            }
        }
    }
    printf("\nFile No\tFile Size\tBlock No\tBlock
Size\tFragment\n");
    for (i = 0; i < nf; i++) {
        printf("%d\t%d\t\t%d\t\t%d\t\t%d\n",
               i + 1, f[i], ff[i] + 1, b[ff[i]], frag[i]);
    }

    return 0;
}
```

Sample Output:

```
Enter the number of blocks:4
Enter the number of files:3

Enter the size of the blocks:-
Block 1:5
Block 2:8
Block 3:4
Block 4:10
Enter the size of the files:-
File 1:1
File 2:4
File 3:7

File_no:        File_size :     Block_no:       Block_size:     Fragment
1               1               1               5               4
2               4               2               8               4
3               7               4               10              3_
```

Result:

First Fit memory allocation technique was implemented using C