Ex. No.: 10a)
Date: 9/4/25

BEST FIT

Aim:

To implement Best Fit memory allocation technique using Python.

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:

```
def bestfit (bsize, m, psize, n):
    alloc = [-1]* n
    for i in range (n);
        best idx =-1
        for j in range (m);
            if b size [j] > = psize [i]:
                if best idx ==-1;
                    best idx=j
                elif b size [best idx] > bsize[j]:
                    best idx=j
if best idx ! = -1 :
    alloc [i] = best idx
    bsize [best idx] - = psize[i]
for i in range (n);
```

```
print (i+1 ,"      ", Psize [i] , end = "      ")
    if (alloc [i]! = = -1);
        print (alloc [i]+1)
    else:
        print ("Not Allocated")

if _name= = "_main_":
    b size = [100, 500, 200, 300, 600]
    p size = [212, 419, 312, 426]
    m = Len (b size)
    n = Len (p size)
    bestfit (b size , m , p size , n)
```

Sample Output:

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

OUTPUT:

Process No	Process size	Block No
1	212	4
2	419	2
3	312	5
4	426	Not Allocated

Result:

Hence the Bestfit for the given processes is implemented and verified.

Ex. No.: 10b)
Date: 10/4/25

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

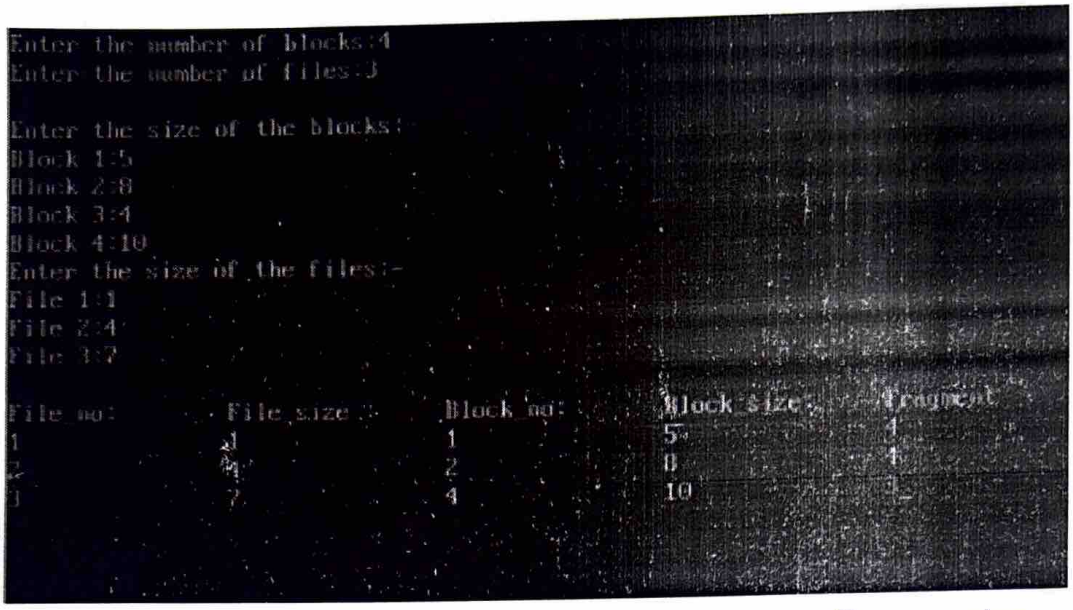
Program Code:

```
# include <stdio.h>
# define Max 25
int main()
{
    int frag [MAX] , b [MAX] , f [MAX] , bf [MAX] ,
                                        ff [MAX] ;

    int i, j, nb, nf, temp;
    for (i=0; i<nb; i++)
    {
        scanf ("%d ", & b[i]);
        bf [i] = 0;
        printf ("Enter the no: of files process :");
        scanf ("%d" , & nf);
```

```
printf ("Enter the size of each files :\n");
for (i=0 ; i<n ; 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 && b[j] >= f[i]) {
            ff (i)=j ;
            bf [j] = 1;
            frag [i] = b[j] - f[i];
            break ;
        }
    }
    if (j==nb) {
        ff[i] = -1;
    }
}
for (i=0 ; i<nf ; i++){
    printf ("%d \t \t %d \t \t", i+1", f[i]);
```

```
if (ff [i] != -1)
        b(ff [i], frag[i])

else
        printf ("Not allocated (\t-\t\t-\n");
    }
}
```

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
```

The fragment of the block

80
15
23
5
20

Process No	Process size	Block No	Frag
P1	20	1	80
P2	30	2	15
P3	50	5	20
P4	40	4	5
P5	10	3	23

Result:

Using the first fit memory allocation algorithm is implemented