Ex. No.: 10a)
Date: 11/4/25

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:

```
#include <stdio.h>
int main() {
    int n,m;
    printf("Enter the no of block");
    scanf("%d",&n);
    printf("Enter no of processes");
    scanf("%d",&m);
    int blocks[n];
    int process[m];
    int allocation[m];
    for (int i=0; i<m; i++) {
        allocation[i] = -1;
    }
    for (int i=0;i<n;i++)
    {
        scanf("%d",&blocks[i]);
    }
```

```
for ( int i=0; i<m; i++) {
    printf ("Enter process %d size : ", i+1);
    scanf ("%d", &process[i]);
  }
int best_index;
for (int =0; i<m; i++)
{
  best_index = -1;
  for ( int J=0; J<n; J++)
  {
    if (blocks[J] >= process[i])
    {
      if (blocks[J] < block [best_index])
      {
        best_index = J;
        allocation [i] = best_index;
        blocks [best_index] -= process[i];
      }
    }
  }
}
printf ("\n Process no.    Process Size    Block No");
for (int i=0; i<m; i++)
{
  if (allocation[i] != -1)
  {
    printf ("\n%d \t\t %d \t\t %d", i+1, process[i];
    allocation[i]+1);
  } else {
    printf ("\n %d \t\t %d \t\t Not Allocated", P+1, process[i]);
  }
}
}
```

OUTPUT :-

Enter no of blocks : 4

Block size :

B1 - 100
B2 - 500
B3 - 150
B4 - 300

Enter no of processes : 3

Process size :

P1 - 99
P2 - 211
P3 - 300

Process No	Process size	Block No
P1	99	B1
P2	211	B4
P3	300	B2

Sample Output:

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

Result:

Hence the best fit memory allocation technique is executed successfully.