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:

```
#include <stdio.h>
int main(){
 int b[] = {100, 45, 33, 45, 70};
 int pro[] = {20, 30, 50, 40, 10};
 int frag[5], flag[5];
for(int i = 0; i<5; i++)
{
      frag[i] = 0;
      flag[i] = 0;
}
for(int i = 0; i<5; i++)
{
    int min=-1;
    for(int j = 0; j<5; j++)
    {
        if(pro[i] < b[j] && flag[j] == 0)
        {
            if(b[j] < b[min] || min = -1)
                min = j;
        }
    }
    frag[min] = b[min] - pro[i];
    flag[min] = 1;
}
```

```
printf("The unusing fragments of blocks: \n");
for(int i=0; i<5; i++){
    printf("%d\n", frag[i]);
}
return 0;
}
```

OUTPUT:

The unusing fragments of blocks:

90
15
13
5
20.

Process	Process_size	Block_No	Fragment
P1	20	3	13
P2	30	2	15
P3	50	5	20
P4	40	4	5
P5	10	1	90

Sample Output:

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

Result: using C the best fit memory allocation algorithm is implemented.