**NAME : Shaikh Areeba Mohammed Ismail**
**ROLL NO : 46**
**CLASS/BATCH : TE-B-2**

**Practical No :- 05**
Write a program to simulate Memory placement strategies – best fit, first fit, and next fit and worst fit.

---

**// JAVA Code for BEST Fit Memory Allocation**

```
import java.util.Scanner;


public class

BestFitMemoryAllocation {

public static void

main(String[] args) {

        Scanner sc = new Scanner(System.in);


        // Input number of blocks and
processes

        System.out.print("Enter number of

memory blocks: ");         int nb =

sc.nextInt();         int[] blockSize =

new int[nb];         int[] blockAllocated

= new int[nb]; // 0 = free, 1 = allocated


        System.out.println("Enter sizes of
memory blocks:");
```

```
        for (int i = 0; i <
nb; i++) {
blockSize[i] =
sc.nextInt();
blockAllocated[i] = 0;
        }

        System.out.print("Enter number of processes: ");
int np = sc.nextInt();         int[] processSize = new int[np];
int[] allocation = new int[np]; // Stores block number for each
process (-1 if not allocated)         System.out.println("Enter
sizes of processes:");         for (int i = 0; i < np; i++) {
processSize[i] = sc.nextInt();             allocation[i] = -1;
// Initially, no allocation
        }

        // Best Fit allocation         for
(int i = 0; i < np; i++) {             int
bestIndex = -1;             for (int j = 0; j
< nb; j++) {                 if (blockSize[j]
>= processSize[i]) { // Block can hold process
if (bestIndex == -1 || blockSize[j] <
blockSize[bestIndex]) {
bestIndex = j;
                    }
                }
```

```
            }

            if (bestIndex != -1) {
// Allocate block to process
allocation[i] = bestIndex;
blockSize[bestIndex] -= processSize[i]; //
Reduce block size
            }
        }

        // Output allocation
        System.out.println("\nProcess
No.\tProcess Size\tBlock No.");
        for (int i = 0; i < np; i++) {
            System.out.print("   " + (i + 1) +
"\t\t" + processSize[i] + "\t\t");
if (allocation[i] != -1)

System.out.println(allocation[i] + 1);
            else
                System.out.println("Not
Allocated");
        }
    }
}
```

**OUTPUT:-**

```
Enter number of memory blocks: 5

Enter sizes of memory blocks:

100 500 200 300 600

Enter number of

processes: 4

Enter sizes of processes:

212 417 112 426


Process No.     Process Size     Block No.

1              212                4

2              417                2

3              112                3

4              426               5
```