LAB 09

Implement the above code and paste the screen shot of the output.

CODE:

```
#include <stdio.h>


int main() {
    int p[10], np, b[10], nb, ch;
    int c[10], d[10], alloc[10], flag[10];
    int i, j;
    printf("\nEnter the number of processes: ");
    scanf("%d", &np);
    printf("Enter the number of blocks: ");
    scanf("%d", &nb);
    printf("Enter the size of each process:\n");
    for (i = 0; i < np; i++) {
        printf("Process %d: ", i);
        scanf("%d", &p[i]);
    }
    printf("Enter the block sizes:\n");
    for (j = 0; j < nb; j++) {
        printf("Block %d: ", j);
        scanf("%d", &b[j]);
        c[j] = b[j];
        d[j] = b[j];
    }
    if (np <= nb) {
        printf("\n1. First Fit\n2. Best Fit\n3. Worst Fit");
        do {
```

```
        printf("\nEnter your choice: ");
        scanf("%d", &ch);
        switch (ch) {
            case 1:
                printf("\nFirst Fit\n");
                for (i = 0; i < np; i++) {
                    flag[i] = 1;
                    for (j = 0; j < nb; j++) {
                        if (p[i] <= b[j]) {
                            alloc[j] = p[i];
                            printf("\nProcess %d of size %d is allocated in block %d of size %d", i, p[i],
j, b[j]);
                            flag[i] = 0;
                            b[j] = 0;
                            break;
                        }
                    }
                }
                for (i = 0; i < np; i++) {
                    if (flag[i] != 0)
                        printf("\nProcess %d of size %d is not allocated", i, p[i]);
                }
                break;

            case 2:
                printf("\nBest Fit\n");
                for (i = 0; i < nb; i++) {
                    for (j = i + 1; j < nb; j++) {
```

```
            if (c[i] > c[j]) {
                int temp = c[i];
                c[i] = c[j];
                c[j] = temp;
            }
        }
    }
    printf("\nAfter sorting block sizes:\n");
    for (i = 0; i < nb; i++)
        printf("Block %d: %d\n", i, c[i]);
    for (i = 0; i < np; i++) {
        flag[i] = 1;
        for (j = 0; j < nb; j++) {
            if (p[i] <= c[j]) { alloc[j] = p[i]; printf("\nProcess %d of size %d is allocated
            in block %d of size %d", i, p[i],

j, c[j]);
                flag[i] = 0;
                c[j] = 0;
                break;
            }
        }
    }
    for (i = 0; i < np; i++) {
        if (flag[i] != 0)
            printf("\nProcess %d of size %d is not allocated", i, p[i]);
    }
    break;
```

```
            case 3:
                printf("\nWorst Fit\n");
                for (i = 0; i < nb; i++) {
                    for (j = i + 1; j < nb; j++) {
                        if (d[i] < d[j]) {
                            int temp = d[i];
                            d[i] = d[j];
                            d[j] = temp;
                        }
                    }
                }
                printf("\nAfter sorting block sizes:\n");
                for (i = 0; i < nb; i++)
                    printf("Block %d: %d\n", i, d[i]);

                for (i = 0; i < np; i++) {
                    flag[i] = 1;
                    for (j = 0; j < nb; j++) {
                        if (p[i] <= d[j]) {
                            alloc[j] = p[i];
                            printf("\nProcess %d of size %d is allocated in block %d of size %d", i, p[i],
j, d[j]);
                            flag[i] = 0;
                            d[j] = 0;
                            break;
                        }
                    }
```

```
            }
            for (i = 0; i < np; i++) {
               if (flag[i] != 0)
                  printf("\nProcess %d of size %d is not allocated", i, p[i]);
            }
            break;
         default:
            printf("Invalid Choice...!");
            break;
      }
   } while (ch <= 3);
  }
  return 0;
}
```

OUTPUT: