**FALL – SEMESTER**
**Course Code:** MCSE504P
**Course-Title:** – Operating System
**DIGITAL ASSIGNMENT - V**
**(LAB)**

**Name: Nidhi Singh**
**Reg. No:22MAI0015**

**Slot-** L51+L52

**Faculty :** SALEEM DURAI M.A - SCOPE

---

**Dynamic memory allocation algorithms – First-fit, Best-fit, Worst-fit algorithms.**

**Program :-**

```c
#include <stdio.h>

void implimentBestFit(int blockSize[], int blocks, int processSize[], int processes)
{
    printf("\n\t\t----------------------22MAI0015-------------------\n");
    printf("Best fit with variable type :\n");
     int allocation[processes];
     for(int i = 0; i < processes; i++){
        allocation[i] = -1;
    }
  for (int i=0; i < processes; i++)
    {
        int indexPlaced = -1;
        for (int j=0; j < blocks; j++)
        {
            if (blockSize[j] >= processSize[i])
            {
                // place it at the first block fit to accomodate process
                if (indexPlaced == -1)
                    indexPlaced = j;

                // if any future block is better that is
                // any future block with smaller size encountered
                // that can accomodate the given process
                else if (blockSize[j] < blockSize[indexPlaced])
                    indexPlaced = j;
            }
        }

        // If we were successfully able to find block for the process
        if (indexPlaced != -1)
        {
            // allocate this block j to process p[i]
            allocation[i] = indexPlaced;

            // Reduce available memory for the block
            blockSize[indexPlaced] -= processSize[i];
        }
    }
```

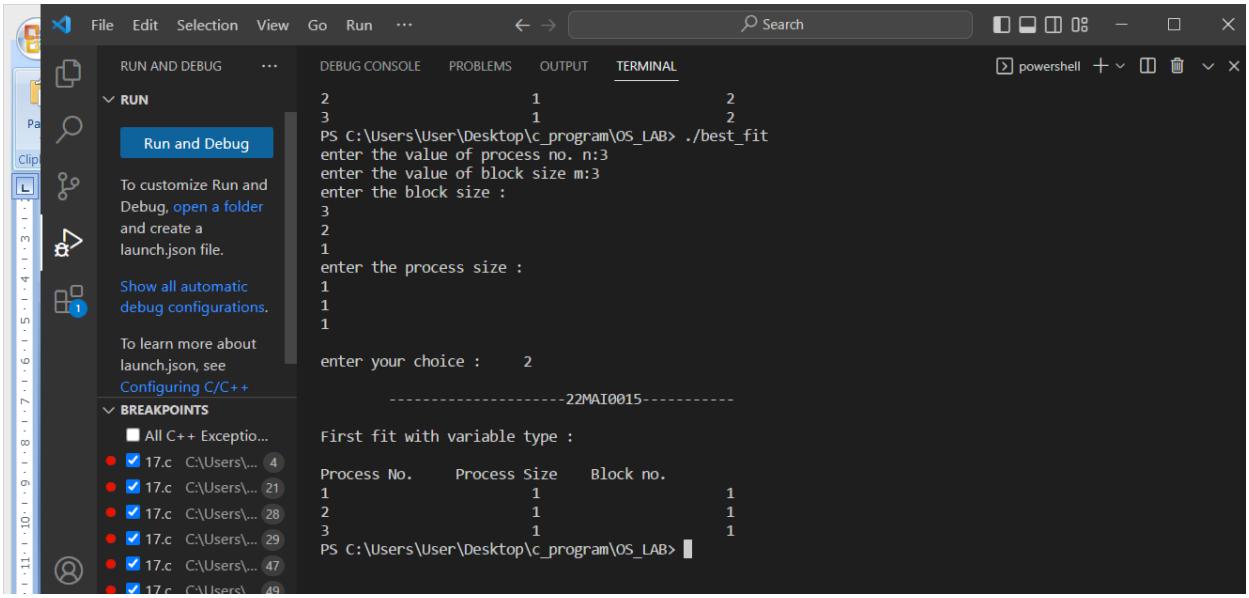
```
    printf("\nProcess No.\tProcess Size\tBlock no.\n");
    for (int i = 0; i < processes; i++)
    {
        printf("%d \t\t\t %d \t\t\t", i+1, processSize[i]);
        if (allocation[i] != -1)
            printf("%d\n",allocation[i] + 1);
        else
            printf("Not Allocated\n");
    }
}
 void implimentFirstFit(int blockSize[], int blocks, int processSize[], int processes)
{
    printf("\n\t--------------------22MAI0015----------\n");
    printf("\nFirst fit with variable type :\n");
    // This will store the block id of the allocated block to a process
    int allocate[processes];

    // initially assigning -1 to all allocation indexes
    // means nothing is allocated currently
    for(int i = 0; i < processes; i++)
    {
        allocate[i] = -1;
    }

    // take each process one by one and find
    // first block that can accomodate it
    for (int i = 0; i < processes; i++)
    {
        for (int j = 0; j < blocks; j++) {
            if (blockSize[j] >= processSize[i])
            {
                // allocate block j to p[i] process
                allocate[i] = j;

                // Reduce size of block j as it has accomodated p[i]
                blockSize[j] -= processSize[i];

                break;
            }
        }
    }

    printf("\nProcess No.\tProcess Size\tBlock no.\n");
    for (int i = 0; i < processes; i++)
    {
        printf("%d \t\t\t %d \t\t\t", i+1, processSize[i]);
        if (allocate[i] != -1)
            printf("%d\n",allocate[i] + 1);
        else
            printf("Not Allocated\n");
    }
}
 void implimentWorstFit(int blockSize[], int blocks, int processSize[], int processes)
{
    printf("\n-------------------22MAI0015-------------------");
    printf("\nWorst fit with variable type :\n");
    int allocation[processes];
    for(int i = 0; i < processes; i++){
```

```
        allocation[i] = -1;
    }
   for (int i=0; i<processes; i++)
    {
        int indexPlaced = -1;
        for (int j=0; j<blocks; j++)
        {
            if (blockSize[j] >= processSize[i])
            {
                 if (indexPlaced == -1)
                    indexPlaced = j;
                 else if (blockSize[indexPlaced] < blockSize[j])
                    indexPlaced = j;
            }
        }
         if (indexPlaced != -1)
        {
             allocation[i] = indexPlaced;
             blockSize[indexPlaced] -= processSize[i];
        }
    }

    printf("\nProcess No.\tProcess Size\tBlock no.\n");
    for (int i = 0; i < processes; i++)
    {
        printf("%d \t\t\t %d \t\t\t", i+1, processSize[i]);
        if (allocation[i] != -1)
            printf("%d\n",allocation[i] + 1);
        else
            printf("Not Allocated\n");
    }
}
int main()
{
    int n,m;
    printf("enter the value of process no. n:");
    scanf("%d",&n);
    printf("enter the value of block size m:");
    scanf("%d",&m);
    int blockSize[m];
    int processSize[n];
    printf("enter the block size :\n");
      for(int i=0;i<m;i++)
      {
        scanf("%d", &blockSize[i]);
      }
    printf("enter the process size :\n");
    for(int i=0;i<n;i++)
      {
        scanf("%d", &processSize[i]);
      }
     int ch;
        printf("\nenter your choice :\t");
        scanf("%d", &ch);
        switch(ch)
        {
            case 1:
                 implimentBestFit(blockSize, m, processSize, n);
```

```
            break;
        case 2:
            implimentFirstFit(blockSize, m, processSize, n);
        break;
        case 3:
            implimentWorstFit(blockSize, m, processSize, n);
        break;
        default :
         printf("enter correct choice");

    }
return 0 ;
}
```

**Output :-**

**1. First Fit**




**2. Best Fit**

```
PS C:\Users\User\Desktop\c_program\OS_LAB> gcc best_fit.c -o best_fit.exe
PS C:\Users\User\Desktop\c_program\OS_LAB> ./best_fit
enter the value of process no. n:3
enter the value of block size m:3
enter the block size :
1
2
3
enter the process size :
1
1
1

enter your choice :     1

                    ----------22MAI0015----------
Best fit with variable type :

Process No.     Process Size    Block no.
1                       1                       1
2                       1                       2
3                       1                       2
PS C:\Users\User\Desktop\c_program\OS_LAB> ./best_fit
```

### 3. Worst Fit

```
PS C:\Users\User\Desktop\c_program\OS_LAB> ./best_fit
enter the value of process no. n:4
enter the value of block size m:5
enter the block size :
10
50
100
40
20
enter the process size :
10
10
10
40

enter your choice :     3

-------------------22MAI0015-------------------
Worst fit with variable type :

Process No.     Process Size    Block no.
1                       10                      3
2                       10                      3
3                       10                      3
4                       40                      3
PS C:\Users\User\Desktop\c_program\OS_LAB>
```