```
********FIRST FIT********

Enter the no of memory blocks: 5
Enter size of memory block 1: 100
Enter size of memory block 2: 500
Enter size of memory block 3: 200
Enter size of memory block 4: 300
Enter size of memory block 5: 600

Enter the number of processes: 4
Enter size of process 1: 212
Enter size of process 2: 417
Enter size of process 3: 112
Enter size of process 4: 426
Block no                Block size              Process no              Process
1                       100                     Allocation failed       Allocation failed
2                       500                     1                       212
3                       200                     3                       112
4                       300                     Allocation failed       Allocation failed
5                       600                     2                       417
```

```
********BEST FIT********

Enter the no of memory blocks: 5
Enter size of memory block 1: 100
Enter size of memory block 2: 500
Enter size of memory block 3: 200
Enter size of memory block 4: 300
Enter size of memory block 5: 600

Enter the number of processes: 4
Enter size of process 1: 212
Enter size of process 2: 417
Enter size of process 3: 112
Enter size of process 4: 426

Block no                Block size              Process no              Process
1                       100                     Allocation failed       Allocation failed
2                       200                     3                       112
3                       300                     1                       212
4                       500                     2                       417
5                       600                     4                       426
```

```
********WORST FIT********

Enter the no of memory blocks: 5
Enter size of memory block 1: 100
Enter size of memory block 2: 500
Enter size of memory block 3: 200
Enter size of memory block 4: 300
Enter size of memory block 5: 600

Enter the number of processes: 4
Enter size of process 1: 212
Enter size of process 2: 417
Enter size of process 3: 112
Enter size of process 4: 426

Block no                Block size              Process no              Process
1                       600                     1                       212
2                       500                     2                       417
3                       300                     3                       112
4                       200                     Allocation failed       Allocation failed
5                       100                     Allocation failed       Allocation failed
```

```c
#include<stdio.h>
#include<math.h>
void BinPacking_NextFit(int weights[], int capacity, int n)
{
    int bin = 0;
    int remainingCapacity = capacity;
    for (int i = 0; i < n; i++)
    {
        if (weights[i] > remainingCapacity)
        {
            bin++;
            remainingCapacity = capacity - weights[i];
        }
        else
        {
            remainingCapacity -= weights[i];
        }
        if (remainingCapacity == 0 || i == n - 1)
        {
            bin++;
            remainingCapacity = capacity;
        }
    }
    printf("\nUsing Next Fit : %d", bin);
}

void BinPacking_FirstFit(int weights[], int capacity, int n)
{
    int bin=0, j;
    int remainingCapacity[n];
    for(int i=0; i<n; i++)
    {
        remainingCapacity[i] = capacity;
    }

    for(int i=0; i<n; i++)
    {
        for(j=0; j<bin; j++)
        {
            if(weights[i] <= remainingCapacity[j])
            {
                remainingCapacity[j] -= weights[i];
                break;
            }
        }
        if(j==bin)
        {
            remainingCapacity[bin] = capacity - weights[i];
            bin++;
        }
    }
    printf("\nUsing First Fit: %d",bin);
}

void BinPacking_BestFit(int weights[], int capacity, int n)
{
    int bin=0   , j;
    int remainingCapacity[n];
    for (int i = 0; i < n; i++)
    {
        remainingCapacity[i]=capacity;
    }

    for(int i=0; i<n; i++)
    {
        int minimum=capacity+1;
        int index=0;
        for(j=0; j<bin; j++)
        {
            if(weights[i] <= remainingCapacity[j] &&
             remainingCapacity[j]-weights[i] < minimum)
            {
                index=j;
                minimum = remainingCapacity[j]-weights[i];
            }
        }
        if(minimum==capacity+1)
        {
            remainingCapacity[bin] = capacity - weights[i];
            bin++;
        }
        else
        {
            remainingCapacity[index] -= weights[i];
        }
    }
    printf("\nUsing Best Fit : %d",bin);
}

void main()
{
    int n;
    int capacity;
    printf("BIN PACKING PROBLEM USING NEXT, FIRST AND BEST FIT\n\n");
    printf("Enter the number of items: ");
    scanf("%d", &n);
    int weights[n];
    float totalWeight=0;
    for(int i=0; i<n; i++)
    {
        printf("Enter the item %d: ",i+1);
        scanf("%d", &weights[i]);
        totalWeight += weights[i];
    }
    printf("\nEnter the maximum capacity of bin: ");
    scanf("%d", &capacity);
    int lowerBound = ceil(totalWeight/(float)capacity);
    printf("\nLower Bound(Min number of bins required): %d",
     lowerBound);
    printf("\nThe total number of bins required:");
    BinPacking_NextFit(weights, capacity, n);
    BinPacking_FirstFit(weights, capacity, n);
    BinPacking_BestFit(weights, capacity, n);
}
```

```
BIN PACKING PROBLEM USING NEXT, FIRST AND BEST FIT

Enter the number of items: 9
Enter the item 1: 5
Enter the item 2: 7
Enter the item 3: 5
Enter the item 4: 2
Enter the item 5: 4
Enter the item 6: 2
Enter the item 7: 5
Enter the item 8: 1
Enter the item 9: 6

Enter the maximum capacity of bin: 10

Lower Bound(Min number of bins required): 4
The total number of bins required:
Using Next Fit : 6
Using First Fit: 5
Using Best Fit : 5
```

```
DISK SCEDULING USING C-SCAN

Enter the number of tracks present in request queue: 7
Enter track number 1: 82
Enter track number 2: 170
Enter track number 3: 43
Enter track number 4: 140
Enter track number 5: 24
Enter track number 6: 16
Enter track number 7: 190

Enter the total number of tracks in the disk: 200

Enter the current position of R/W head: 50

The total number of track movement using C-SCAN is: 391
```

```
DISK SCEDULING USING FCFS

Enter the number of tracks present in request queue: 7
Enter track number 1: 82
Enter track number 2: 170
Enter track number 3: 43
Enter track number 4: 140
Enter track number 5: 24
Enter track number 6: 16
Enter track number 7: 190

Enter the current position of R/W head: 50
The total number of track movement using FCFS is: 642
```

```
DISK SCEDULING USING SSTF

Enter the number of tracks present in request queue: 7
Enter track number 1: 82
Enter track number 2: 170
Enter track number 3: 43
Enter track number 4: 140
Enter track number 5: 24
Enter track number 6: 16
Enter track number 7: 190

Enter the total number of tracks in the disk: 200

Enter the current position of R/W head: 50

The total number of track movement using SSTF is: 208
```

```
DISK SCEDULING USING SCAN

Enter the number of tracks present in request queue: 7
Enter track number 1: 82
Enter track number 2: 170
Enter track number 3: 43
Enter track number 4: 140
Enter track number 5: 24
Enter track number 6: 16
Enter track number 7: 190

Enter the total number of tracks in the disk: 200

Enter the current position of R/W head: 50

The total number of track movement using SCAN is: 332
```