PROGRAM CODE

```
#include <stdio.h>
#include <stdlib.h>

void ff(int n, float block[], int num, float proc[]) {
	float block1[n];

	for (int i=0; i<n; i++)
		block1[i] = block[i];

	printf("\nFIRST FIT");
	for(int i=0; i<num; i++)
		for(int j=0; j<n; j++)
			if (proc[i] <= block1[j]) {
				printf("\nProcess %d (%.2f KB) can be allocated at block %d (%.2f KB)
(Internal fragmentation = %.2f KB)", i+1, proc[i], j+1, block1[j], block1[j]-proc[i]);
				block1[j] = 0;
				break;
			} else if (n-j == 1)
				printf("\nProcess %d (%.2f KB) cannot be allocated", i+1, proc[i]);
	printf("\n");
}

void bf(int n, float block[], int num, float proc[]) {
	float block1[n];

	for (int i=0; i<n; i++)
		block1[i] = block[i];

	printf("\nBEST FIT");
	for(int i=0; i<num; i++) {
		int min = -1;

		for(int j=0; j<n; j++)
			if(proc[i] <= block1[j] && (min == -1 || block1[j] < block1[min]))
				min = j;

		if (min == -1)
			printf("\nProcess %d (%.2f KB) cannot be allocated", i+1, proc[i]);
		else {
			printf("\nProcess %d (%.2f KB) can be allocated at block %d (%.2f KB) (Internal
fragmentation = %.2f KB)", i+1, proc[i], min+1, block1[min], block1[min]-proc[i]);
			block1[min] = 0;
		}
	}
	printf("\n");
}

void wf(int n, float block[], int num, float proc[]) {
	float block1[n];

	for (int i=0; i<n; i++)
		block1[i] = block[i];

	printf("\nWORST FIT");
```

```
	for(int i=0; i<num; i++) {
		int max = -1;

		for(int j=0; j<n; j++)
			if(proc[i] <= block1[j] && (max == -1 || block1[j] > block1[max]))
				max = j;

		if (max == -1)
			printf("\nProcess %d (%.2f KB) cannot be allocated", i+1, proc[i]);
		else {
			printf("\nProcess %d (%.2f KB) can be allocated at block %d (%.2f KB) (Internal
fragmentation = %.2f KB)", i+1, proc[i], max+1, block1[max], block1[max]-proc[i]);
			block1[max] = 0;
		}
	}
	printf("\n");
}

void main() {
	int n, num, opt;
	printf("Enter the number of blocks: ");
	scanf("%d", &n);

	float block[n];

	for(int i=0; i<n; i++) {
		printf("Enter the size (in KB) of block %d: ", i+1);
		scanf("%f", &block[i]);
	}
	printf("\n");

	printf("Enter the number of processes: ");
	scanf("%d", &num);

	float proc[num];

	for(int i=0; i<num; i++) {
		printf("Enter the size (in KB) of process %d: ", i+1);
		scanf("%f", &proc[i]);
	}

	while(1) {
		printf("\n1. First Fit allocation scheme\n2. Best Fit allocation scheme\n3. Worst Fit
allocation scheme\n4. Exit\n");
		printf("Enter your option: ");
		scanf("%d", &opt);

		switch (opt) {
			case 1:
				ff(n, block, num, proc);
				break;
			case 2:
				bf(n, block, num, proc);
				break;
			case 3:
				wf(n, block, num, proc);
				break;
```

```
                        case 4:
                                printf("\nExit.\n");
                                exit(0);
                        default:
                                printf("\nInvalid option.\n");
                                break;
                }
        }
}
```

SAMPLE OUTPUT

Enter the number of blocks: 4

Enter the size (in KB) of block 1: 5

Enter the size (in KB) of block 2: 4

Enter the size (in KB) of block 3: 6

Enter the size (in KB) of block 4: 7

Enter the number of processes: 5

Enter the size (in KB) of process 1: 2

Enter the size (in KB) of process 2: 5

Enter the size (in KB) of process 3: 7

Enter the size (in KB) of process 4: 3

Enter the size (in KB) of process 5: 4

1. First Fit allocation scheme

2. Best Fit allocation scheme

3. Worst Fit allocation scheme

4. Exit

Enter your option: 1

FIRST FIT

Process 1 (2.00 KB) can be allocated at block 1 (5.00 KB) (Internal fragmentation = 3.00 KB)

Process 2 (5.00 KB) can be allocated at block 3 (6.00 KB) (Internal fragmentation = 1.00 KB)

Process 3 (7.00 KB) can be allocated at block 4 (7.00 KB) (Internal fragmentation = 0.00 KB)

Process 4 (3.00 KB) can be allocated at block 2 (4.00 KB) (Internal fragmentation = 1.00 KB)

Process 5 (4.00 KB) cannot be allocated

1. First Fit allocation scheme

2. Best Fit allocation scheme

3. Worst Fit allocation scheme

4. Exit

Enter your option: 2

BEST FIT

Process 1 (2.00 KB) can be allocated at block 2 (4.00 KB) (Internal fragmentation = 2.00 KB)

Process 2 (5.00 KB) can be allocated at block 1 (5.00 KB) (Internal fragmentation = 0.00 KB)

Process 3 (7.00 KB) can be allocated at block 4 (7.00 KB) (Internal fragmentation = 0.00 KB)

Process 4 (3.00 KB) can be allocated at block 3 (6.00 KB) (Internal fragmentation = 3.00 KB)

Process 5 (4.00 KB) cannot be allocated

1. First Fit allocation scheme

2. Best Fit allocation scheme

3. Worst Fit allocation scheme

4. Exit

Enter your option: 3

WORST FIT

Process 1 (2.00 KB) can be allocated at block 4 (7.00 KB) (Internal fragmentation = 5.00 KB)

Process 2 (5.00 KB) can be allocated at block 3 (6.00 KB) (Internal fragmentation = 1.00 KB)

Process 3 (7.00 KB) cannot be allocated

Process 4 (3.00 KB) can be allocated at block 1 (5.00 KB) (Internal fragmentation = 2.00 KB)

Process 5 (4.00 KB) can be allocated at block 2 (4.00 KB) (Internal fragmentation = 0.00 KB)

1. First Fit allocation scheme

2. Best Fit allocation scheme

3. Worst Fit allocation scheme

4. Exit

Enter your option: 4

Exit.