

LAB 14

QUESTION: Write a C program to simulate the following file allocation strategies.

a) Sequential b) Indexed c) Linked

ANSWER:

CODE:

A) Sequential

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

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

```
int main() {
```

```
    int f[50], i, st, j, len, c, k;
```

```
    // Initialize all blocks to 0 (free)
```

```
    for (i = 0; i < 50; i++)
```

```
        f[i] = 0;
```

```
    do {
```

```
        printf("\nEnter the starting block and length of file: ");
```

```
        scanf("%d%d", &st, &len);
```

```
        // Check if requested blocks are free
```

```
        for (j = st; j < (st + len); j++) {
```

```
            if (f[j] != 0) {
```

```
                printf("Block %d is already allocated.\n", j);
```

```
                break;
```

```
            }
```

```
}

// If all blocks are free, allocate them
if (j == (st + len)) {
    for (k = st; k < (st + len); k++) {
        f[k] = 1;
        printf("%d -> Allocated\n", k);
    }
    printf("File allocated successfully.\n");
} else {
    printf("File allocation failed. Try again.\n");
}

printf("\nDo you want to enter more files? (Yes = 1 / No = 0): ");
scanf("%d", &c);

} while (c == 1);

return 0;
}
```

Output:

```
Enter the starting block and length of file: 5 4
5 -> Allocated
6 -> Allocated
7 -> Allocated
8 -> Allocated
File allocated successfully.

Do you want to enter more files? (Yes = 1 / No = 0): 0

-----
Process exited after 13.68 seconds with return value 0
Press any key to continue . . .
```

B) INDEXED

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

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

```
int main() {
```

```
    int f[50], i, j, k, indexBlock, n, blocks[50], c;
```

```
    // Initialize all disk blocks to 0 (free)
```

```
    for (i = 0; i < 50; i++)
```

```
        f[i] = 0;
```

```
    do {
```

```
        printf("\nEnter index block: ");
```

```
        scanf("%d", &indexBlock);
```

```
        if (f[indexBlock] == 0) {
```

```
            f[indexBlock] = 1;
```

```
            printf("Enter number of blocks for the file: ");
```

```
            scanf("%d", &n);
```

```
            printf("Enter the block numbers:\n");
```

```
            for (i = 0; i < n; i++)
```

```
                scanf("%d", &blocks[i]);
```

```
            // Check if any of the blocks are already allocated
```

```
            for (i = 0; i < n; i++) {
```

```
                if (f[blocks[i]] == 1) {
```

```
                    printf("Block %d is already allocated. Allocation failed.\n", blocks[i]);
```

```
        goto skip;
    }
}

// Allocate all blocks
for (j = 0; j < n; j++)
    f[blocks[j]] = 1;

printf("\nFile successfully indexed. Blocks:\n");
for (k = 0; k < n; k++)
    printf("%d -> %d : Allocated\n", indexBlock, blocks[k]);

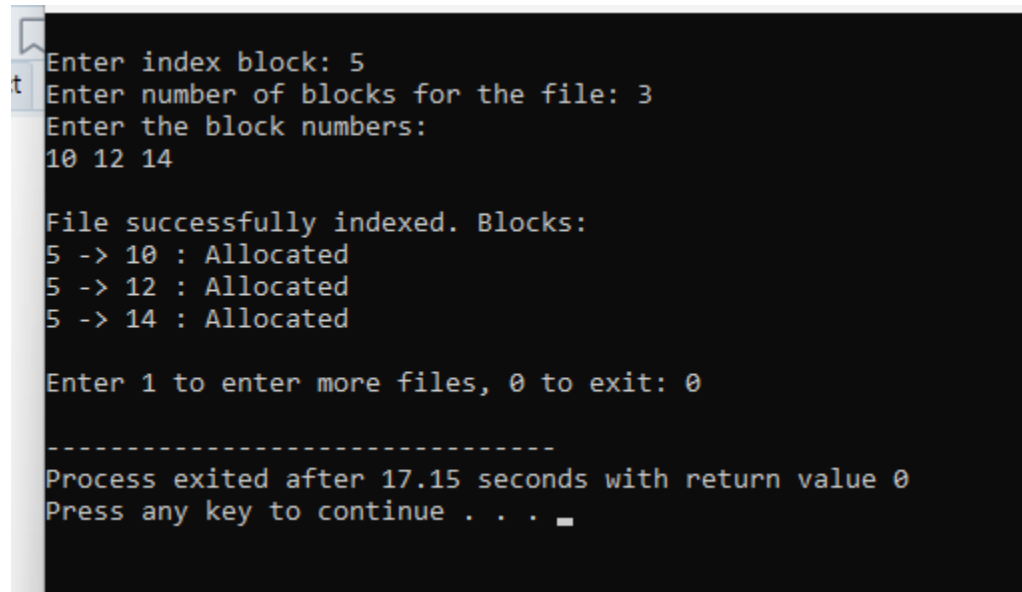
} else {
    printf("Index block already allocated.\n");
}

skip:
printf("\nEnter 1 to enter more files, 0 to exit: ");
scanf("%d", &c);

} while (c == 1);

return 0;
}
```

Output:



```
Enter index block: 5
Enter number of blocks for the file: 3
Enter the block numbers:
10 12 14

File successfully indexed. Blocks:
5 -> 10 : Allocated
5 -> 12 : Allocated
5 -> 14 : Allocated

Enter 1 to enter more files, 0 to exit: 0

-----
Process exited after 17.15 seconds with return value 0
Press any key to continue . . .
```

C) LINKED

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

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

```
int main() {
```

```
    int f[50], p, i, j, k, a, st, len, n, c;
```

```
    // Initialize all blocks to 0 (free)
```

```
    for (i = 0; i < 50; i++)
```

```
        f[i] = 0;
```

```
    printf("Enter how many blocks are already allocated: ");
```

```
    scanf("%d", &p);
```

```
    printf("Enter the block numbers that are already allocated:\n");
```

```
    for (i = 0; i < p; i++) {
```

```
        scanf("%d", &a);
```

```
        if (a >= 0 && a < 50)
```

```
            f[a] = 1;
```

```
    }
```

```
    do {
```

```
        printf("\nEnter the starting index block and the length of the file: ");
```

```
        scanf("%d %d", &st, &len);
```

```
        int allocated = 1;
```

```
        // Check availability
```

```
        for (j = st; j < st + len; j++) {
```

```
            if (f[j] == 1) {
```

```
                allocated = 0;
```

```
                break;
```

```
            }
```

```
        }
```

```
        if (allocated) {
```

```
            // Allocate blocks
```

```
            for (j = st; j < st + len; j++) {
```

```
                f[j] = 1;
```

```
                printf("%d -> Allocated\n", j);
```

```
            }
```

```
            printf("File allocated successfully.\n");
```

```
        } else {
```

```
            printf("Requested blocks are already allocated. File cannot be allocated.\n");
```

```
        }
```

```
printf("\nDo you want to enter another file? (yes-1 / no-0): ");  
scanf("%d", &c);  
  
} while (c == 1);  
  
return 0;  
}
```

Output:

```
Enter how many blocks are already allocated: 3  
Enter the block numbers that are already allocated:  
3 4 5  
  
Enter the starting index block and the length of the file: 6 4  
6 -> Allocated  
7 -> Allocated  
8 -> Allocated  
9 -> Allocated  
File allocated successfully.  
  
Do you want to enter another file? (yes-1 / no-0): 1  
  
Enter the starting index block and the length of the file: 3 2  
Requested blocks are already allocated. File cannot be allocated.  
  
Do you want to enter another file? (yes-1 / no-0): 0  
  
-----  
Process exited after 42.49 seconds with return value 0  
Press any key to continue . . .
```