Linked and sequential file allocation

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

struct sequential
{
   char filename[16];
   int startblock;
   int length;
};

int numberoffilesS = 5;
struct sequential s[5] = {{"count", 0, 2}, {"tr", 14, 3}, {"mail", 19, 6}, {"list", 28, 4}, {"t", 6,
    2}};

struct linked
{
   char filename[16];
   int startblock;
   int endblock;
};

int numberoffilesL = 1;
struct linked l[1] = {{"jeep", 9, 25}};

struct block
{
   int blocknumber;
   struct block *next;
} blockentry[30];

void sequentialfile()
{
   char filename[20];
   printf("Enter the sequential file name to be searched: ");
   scanf("%s", filename);
   for(int i = 0; i < numberoffilesS; i++)
```

```
    {
        if(strcmp(filename, s[i].filename) == 0)
        {
            printf("\nFile name\tStart Block\tNumber of blocks\tBlocks occupied\n");
            printf("%s\t\t%d\t\t%d\t\t", s[i].filename, s[i].startblock, s[i].length);
            int j = 0;
            do
            {
                printf("%d", s[i].startblock + j);
                j++;
            } while(j < s[i].length && printf("->"));
            printf("\n");
            return;
        }
    }
    return;
}

void linkedfile()
{
    int numberofblocks = 0;
    char filename[20];
    printf("\nEnter the linked file name to be searched: ");
    scanf("%s", filename);
    for(int i = 0; i < numberoffilesL; i++)
    {
        if(strcmp(filename, l[i].filename) == 0)
        {
            printf("\nFile name\tStart Block\tEnd Block\tBlocks occupied\n");
            printf("%s\t\t%d\t\t%d\t", l[i].filename, l[i].startblock, l[i].endblock);
            struct block *blockptr = &blockentry[l[i].startblock];
            do
            {
                printf("%d", blockptr->blocknumber);
                blockptr = blockptr->next;
                numberofblocks++;
            } while(blockptr != NULL && printf("->"));
            printf("\nNumber of blocks occupied = %d\n", numberofblocks);
            return;
        }
```

```
    }
    return;
}

int main()
{
    sequentialfile();
    blockentry[9].blocknumber = 9; blockentry[9].next = &blockentry[16];
    blockentry[16].blocknumber = 16; blockentry[16].next = &blockentry[1];
    blockentry[1].blocknumber = 1; blockentry[1].next = &blockentry[25];
    blockentry[25].blocknumber = 25; blockentry[25].next = NULL;
    linkedfile();
    return 0;
}
```

Including Index:

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

struct sequential
{
   char filename[16];
   int startblock;
   int length;
};

int numberoffilesS = 5;
struct sequential s[5] = {{"count", 0, 2}, {"tr", 14, 3}, {"mail", 19, 6}, {"list", 28, 4}, {"t", 6,
    2}};

struct linked
{
   char filename[16];
   int startblock;
   int endblock;
};

int numberoffilesL = 1;
struct linked l[1] = {{"jeep", 9, 25}};

struct indexed
{
   char filename[16];
   int indexblock;
};

int numberoffilesE = 1;
struct indexed e[1] = {{"bike", 10}};

struct block
{
   int blocknumber;
   struct block *next;
```

```
} blockentry[30];

void sequentialfile()
{
   char filename[20];
   printf("Enter the sequential file name to be searched: ");
   scanf("%s", filename);
   for(int i = 0; i < numberoffilesS; i++)
   {
      if(strcmp(filename, s[i].filename) == 0)
      {
         printf("\nFile name\tStart Block\tNumber of blocks\tBlocks occupied\n");
         printf("%s\t\t%d\t\t%d\t\t", s[i].filename, s[i].startblock, s[i].length);
         int j = 0;
         do
         {
            printf("%d", s[i].startblock + j);
            j++;
         } while(j < s[i].length && printf("->"));
         printf("\n");
         return;
      }
   }
   return;
}

void linkedfile()
{
   int numberofblocks = 0;
   char filename[20];
   printf("\nEnter the linked file name to be searched: ");
   scanf("%s", filename);
   for(int i = 0; i < numberoffilesL; i++)
   {
      if(strcmp(filename, l[i].filename) == 0)
      {
         printf("\nFile name\tStart Block\tEnd Block\tBlocks occupied\n");
         printf("%s\t\t%d\t\t%d\t", l[i].filename, l[i].startblock, l[i].endblock);
         struct block *blockptr = &blockentry[l[i].startblock];
         do
```

```
            {
                printf("%d", blockptr->blocknumber);
                blockptr = blockptr->next;
                numberofblocks++;
            } while(blockptr != NULL && printf("->"));
            printf("\nNumber of blocks occupied = %d\n", numberofblocks);
            return;
        }
    }
    return;
}

void indexedfile()
{
    char filename[20];
    printf("\nEnter the indexed file name to be searched: ");
    scanf("%s", filename);
    for(int i = 0; i < numberoffilesE; i++)
    {
        if(strcmp(filename, e[i].filename) == 0)
        {
            printf("\nFile name\tIndex Block\tBlocks occupied\n");
            printf("%s\t\t%d\t", e[i].filename, e[i].indexblock);
            struct block *blockptr = &blockentry[e[i].indexblock];
            int j = 0;
            do
            {
                printf("%d", blockptr->blocknumber);
                j++;
                blockptr = blockptr->next;
            } while(blockptr != NULL && printf("->"));
            printf("\n");
            return;
        }
    }
    return;
}

int main()
{
```

```
    sequentialfile();
    blockentry[9].blocknumber = 9; blockentry[9].next = &blockentry[16];
    blockentry[16].blocknumber = 16; blockentry[16].next = &blockentry[1];
    blockentry[1].blocknumber = 1; blockentry[1].next = &blockentry[25];
    blockentry[25].blocknumber = 25; blockentry[25].next = NULL;
    blockentry[10].blocknumber = 10; blockentry[10].next = &blockentry[11];
    blockentry[11].blocknumber = 11; blockentry[11].next = &blockentry[12];
    blockentry[12].blocknumber = 12; blockentry[12].next = NULL;
    linkedfile();
    indexedfile();
    return 0;
}
```