Module 1 Assignment

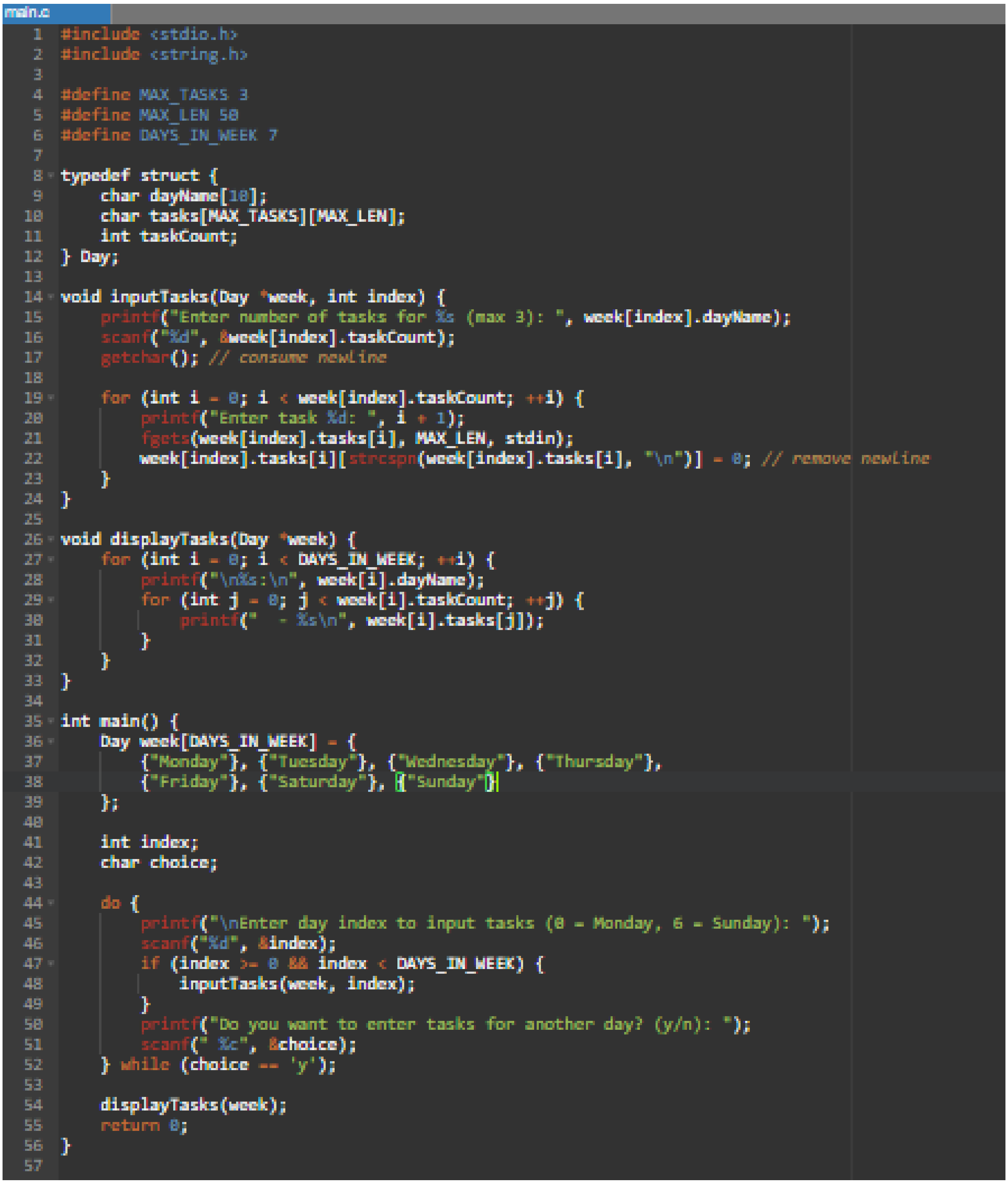
Topic 1: Structures – Calendar for a Week

Code:

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

#define MAX_TASKS 3
#define MAX_LEN 50
#define DAYS_IN_WEEK 7

typedef struct {
    char dayName[10];
    char tasks[MAX_TASKS][MAX_LEN];
    int taskCount;
} Day;

void inputTasks(Day *week, int index) {
    printf("Enter number of tasks for %s (max 3): ", week[index].dayName);
    scanf("%d", &week[index].taskCount);
    getchar(); // consume newline

    for (int i = 0; i < week[index].taskCount; ++i) {
        printf("Enter task %d: ", i + 1);
        fgets(week[index].tasks[i], MAX_LEN, stdin);
        week[index].tasks[i][strcspn(week[index].tasks[i], "\n")] = 0; // remove newline
    }
}
```

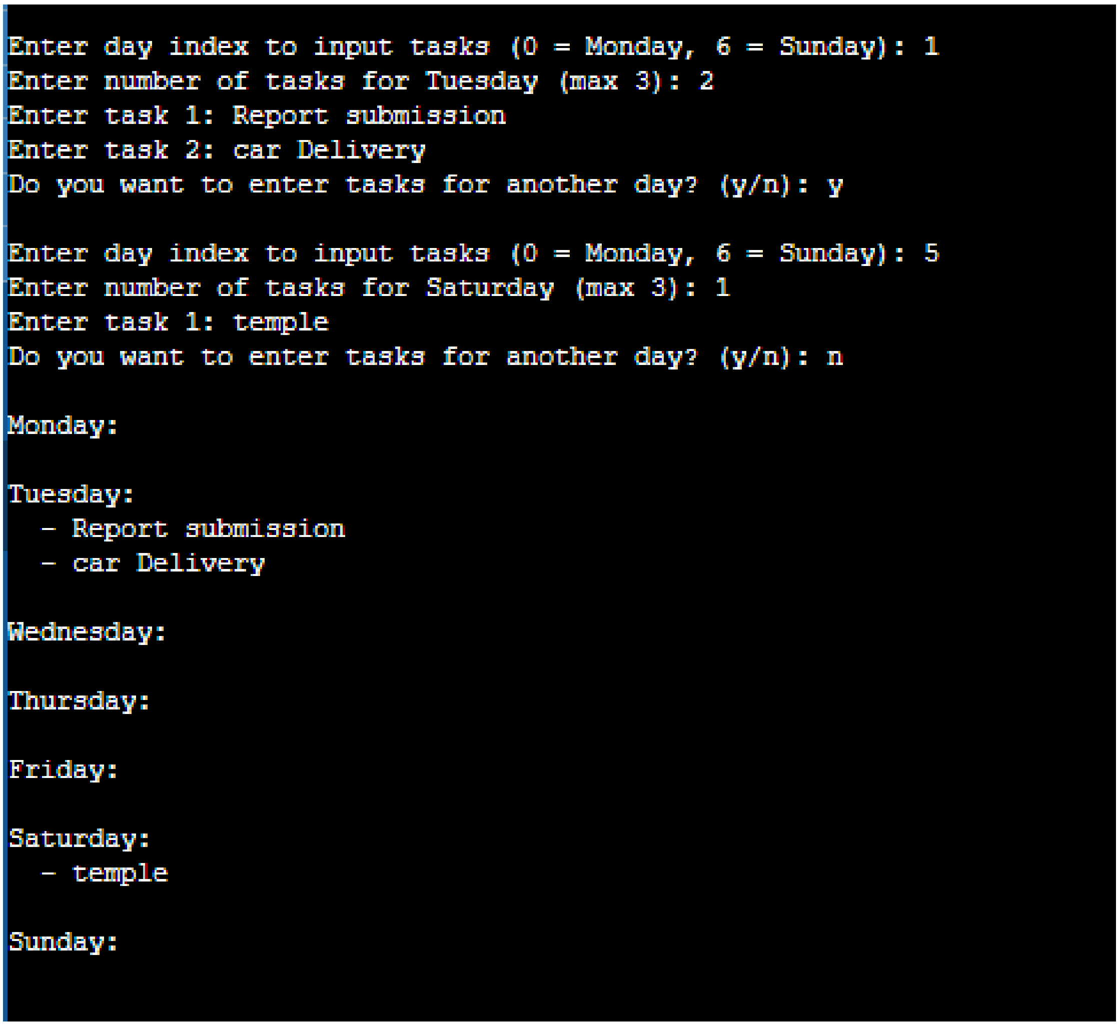
```
void displayTasks(Day *week) {
    for (int i = 0; i < DAYS_IN_WEEK; ++i) {
        printf("\n%s:\n", week[i].dayName);
        for (int j = 0; j < week[i].taskCount; ++j) {
            printf("  - %s\n", week[i].tasks[j]);
        }
    }
}

int main() {
    Day week[DAYS_IN_WEEK] = {
        {"Monday"}, {"Tuesday"}, {"Wednesday"}, {"Thursday"},
        {"Friday"}, {"Saturday"}, {"Sunday"}
    };
    int index;
    char choice;
    do {
        printf("\nEnter day index to input tasks (0 = Monday, 6 = Sunday): ");
        scanf("%d", &index);
        if (index >= 0 && index < DAYS_IN_WEEK) {
            inputTasks(week, index);
        }
        printf("Do you want to enter tasks for another day? (y/n): ");
        scanf(" %c", &choice);
    } while (choice == 'y');

    displayTasks(week);
    return 0;}
```

Code Snippet:



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

#define MAX_TASKS 3
#define MAX_LEN 50
#define DAYS_IN_WEEK 7

typedef struct {
    char dayName[10];
    char tasks[MAX_TASKS][MAX_LEN];
    int taskCount;
} Day;

void inputTasks(Day *week, int index) {
    printf("Enter number of tasks for %s (max 3): ", week[index].dayName);
    scanf("%d", &week[index].taskCount);
    getchar(); // consume newline

    for (int i = 0; i < week[index].taskCount; ++i) {
        printf("Enter task %d: ", i + 1);
        fgets(week[index].tasks[i], MAX_LEN, stdin);
        week[index].tasks[i][strcspn(week[index].tasks[i], "\n")] = 0; // remove newline
    }
}

void displayTasks(Day *week) {
    for (int i = 0; i < DAYS_IN_WEEK; ++i) {
        printf("\n%s:\n", week[i].dayName);
        for (int j = 0; j < week[i].taskCount; ++j) {
            printf("  - %s\n", week[i].tasks[j]);
        }
    }
}

int main() {
    Day week[DAYS_IN_WEEK] = {
        {"Monday"}, {"Tuesday"}, {"Wednesday"}, {"Thursday"},
        {"Friday"}, {"Saturday"}, {"Sunday"}
    };

    int index;
    char choice;

    do {
        printf("\nEnter day index to input tasks (0 = Monday, 6 = Sunday): ");
        scanf("%d", &index);
        if (index >= 0 && index < DAYS_IN_WEEK) {
            inputTasks(week, index);
        }
        printf("Do you want to enter tasks for another day? (y/n): ");
        scanf(" %c", &choice);
    } while (choice == 'y');

    displayTasks(week);
    return 0;
}
```

Output Snippet:



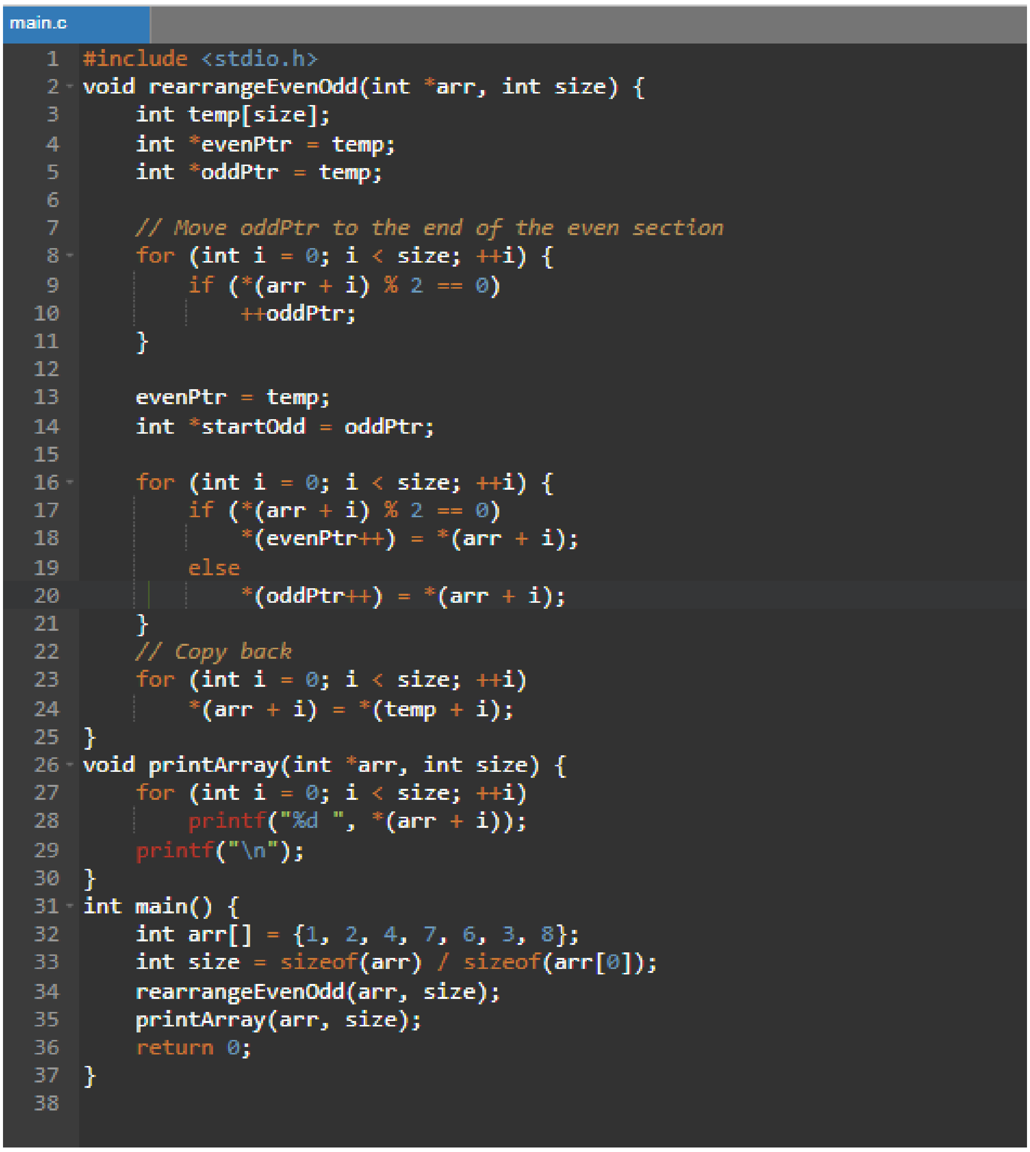
```
Enter day index to input tasks (0 = Monday, 6 = Sunday): 1
Enter number of tasks for Tuesday (max 3): 2
Enter task 1: Report submission
Enter task 2: car Delivery
Do you want to enter tasks for another day? (y/n): y

Enter day index to input tasks (0 = Monday, 6 = Sunday): 5
Enter number of tasks for Saturday (max 3): 1
Enter task 1: temple
Do you want to enter tasks for another day? (y/n): n

Monday:

Tuesday:
  - Report submission
  - car Delivery

Wednesday:

Thursday:

Friday:

Saturday:
  - temple

Sunday:
```

Topic 2: Pointers – Rearranging Array with Only Pointers

Code:

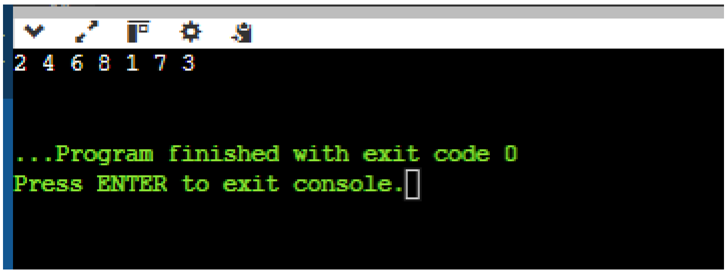
```
#include <stdio.h>
void rearrangeEvenOdd(int *arr, int size) {
    int temp[size];
    int *evenPtr = temp;
    int *oddPtr = temp;
    for (int i = 0; i < size; ++i) {
        if (*(arr + i) % 2 == 0)
            ++oddPtr;
    }
    evenPtr = temp;
    int *startOdd = oddPtr;
    for (int i = 0; i < size; ++i) {
        if (*(arr + i) % 2 == 0)
            *(evenPtr++) = *(arr + i);
        else
            *(oddPtr++) = *(arr + i);}
    for (int i = 0; i < size; ++i)
        *(arr + i) = *(temp + i);}
void printArray(int *arr, int size) {
    for (int i = 0; i < size; ++i)
        printf("%d ", *(arr + i));
    printf("\n");}
int main() {
    int arr[] = {1, 2, 4, 7, 6, 3, 8};
    int size = sizeof(arr) / sizeof(arr[0]);
    rearrangeEvenOdd(arr, size);
    printArray(arr, size);
    return 0;}
```

Code Snippet:



```c
#include <stdio.h>
void rearrangeEvenOdd(int *arr, int size) {
    int temp[size];
    int *evenPtr = temp;
    int *oddPtr = temp;

    // Move oddPtr to the end of the even section
    for (int i = 0; i < size; ++i) {
        if (*(arr + i) % 2 == 0)
            ++oddPtr;
    }

    evenPtr = temp;
    int *startOdd = oddPtr;

    for (int i = 0; i < size; ++i) {
        if (*(arr + i) % 2 == 0)
            *(evenPtr++) = *(arr + i);
        else
            *(oddPtr++) = *(arr + i);
    }
    // Copy back
    for (int i = 0; i < size; ++i)
        *(arr + i) = *(temp + i);
}
void printArray(int *arr, int size) {
    for (int i = 0; i < size; ++i)
        printf("%d ", *(arr + i));
    printf("\n");
}
int main() {
    int arr[] = {1, 2, 4, 7, 6, 3, 8};
    int size = sizeof(arr) / sizeof(arr[0]);
    rearrangeEvenOdd(arr, size);
    printArray(arr, size);
    return 0;
}

```

Output Snippet:



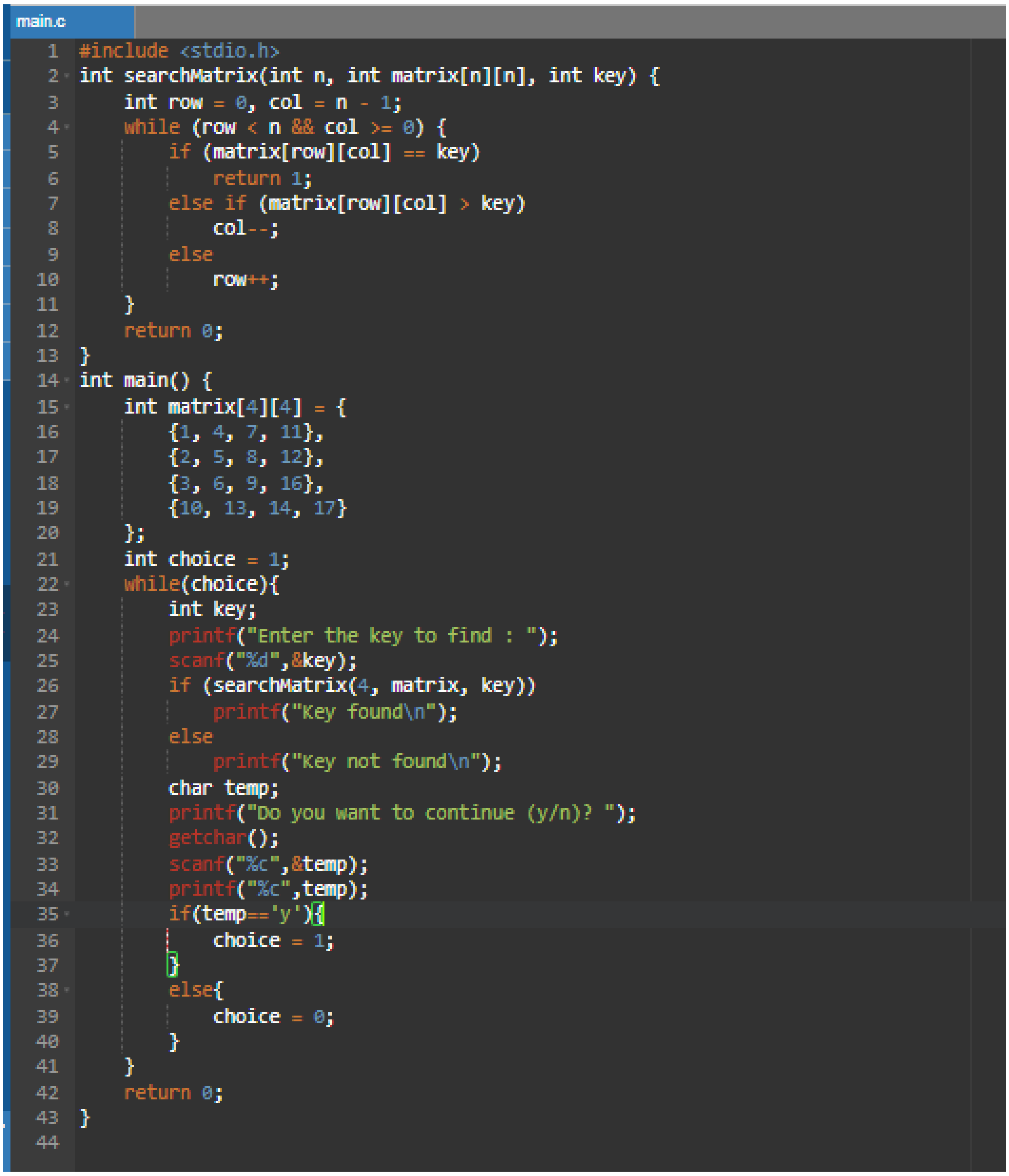
```
2 4 6 8 1 7 3


...Program finished with exit code 0
Press ENTER to exit console.
```

Topic 3: Arrays – Search in Sorted 2D Matrix

Code:

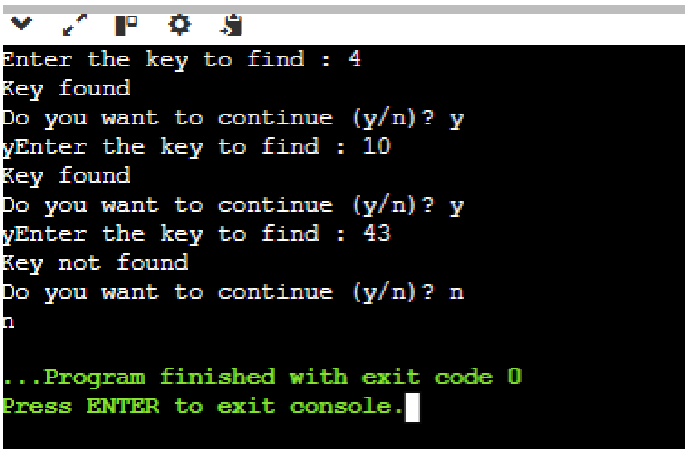
```
#include <stdio.h>


int searchMatrix(int n, int matrix[n][n], int key) {
    int row = 0, col = n - 1;
    while (row < n && col >= 0) {
        if (matrix[row][col] == key)
            return 1;
        else if (matrix[row][col] > key)
            col--;
        else
            row++;
    }
    return 0;
}
int main() {
    int matrix[4][4] = {
        {1, 4, 7, 11},
        {2, 5, 8, 12},
        {3, 6, 9, 16},
        {10, 13, 14, 17}
    };
    int key = 14;
    if (searchMatrix(4, matrix, key))
        printf("Key found\n");
    else
        printf("Key not found\n");
    return 0;
}
```

Code Snippet:



```c
#include <stdio.h>
int searchMatrix(int n, int matrix[n][n], int key) {
    int row = 0, col = n - 1;
    while (row < n && col >= 0) {
        if (matrix[row][col] == key)
            return 1;
        else if (matrix[row][col] > key)
            col--;
        else
            row++;
    }
    return 0;
}
int main() {
    int matrix[4][4] = {
        {1, 4, 7, 11},
        {2, 5, 8, 12},
        {3, 6, 9, 16},
        {10, 13, 14, 17}
    };
    int choice = 1;
    while(choice){
        int key;
        printf("Enter the key to find : ");
        scanf("%d",&key);
        if (searchMatrix(4, matrix, key))
            printf("Key found\n");
        else
            printf("Key not found\n");
        char temp;
        printf("Do you want to continue (y/n)? ");
        getchar();
        scanf("%c",&temp);
        printf("%c",temp);
        if(temp=='y'){
            choice = 1;
        }
        else{
            choice = 0;
        }
    }
    return 0;
}

```

Output Snippet:



```
Enter the key to find : 4
Key found
Do you want to continue (y/n)? y
yEnter the key to find : 10
Key found
Do you want to continue (y/n)? y
yEnter the key to find : 43
Key not found
Do you want to continue (y/n)? n
n

...Program finished with exit code 0
Press ENTER to exit console.
```