```c
#include<stdio.h>
#include<stdlib.h>
#include "Linked_List.h"
#define _CRT_SECURE_NO_WARNINGS

int main() {
    struct Linked_List* start = NULL;
    int arr[100] , count = 0 , temp , number = 0;
    printf("Enter The numbers for the linked list.\nEnter -1 for ending the list.\nThe list: "); //Input from user to start building the list
    for (int i = 0; i < 100; i++){
        scanf_s("%d", &arr[i]);
        if (arr[i] != -1) {
            Build_Linked_List(&start, arr[i]);
            count++;
        }
        else
            break;
    }
    Print_Linked_List(start);
    printf("\n\nEnter number to the linked list and the funcation will organization in the right order all numbers: ");
    scanf_s("%d", &arr[count+1]);
    Build_Linked_List(&start, arr[count+1]);
    Sort_Linked_List(start, count + 1);
    Print_Linked_List(start);
    printf("\n\nEnter a number and the funcation will find if the exists: ");
    scanf_s("%d", &number);
    temp = Search_Number(start, number);
    if (temp>0) //Index exists
        printf("\nindex in list: %d", temp);
    else  //Index not exists
        printf("\n%d dosent exists in the list.", number);
    Reverse_Linked_List(&start);
    printf("\n\nReversed Linked list:");
    Print_Linked_List(start);
    temp = 0;
    Sum_Linked_List(start, &temp);
    printf("\n\nSum of all numbers in the list: %d", temp);
    temp = 1;
    Multiplication_Even_Indexs_Linked_List(start, &temp, 0);
    printf("\n\nMultiplication all numbers that are in the even places on the list: %d\n", temp);
    return 0;
}

```

```h
#pragma once
#include<stdio.h>
#include<stdlib.h>
#define _CRT_SECURE_NO_WARNINGS

struct Linked_List{
    int Data;
    struct Linked_List* Head_To_Tail;
};

void Build_Linked_List(struct Linked_List**,int);
void Sort_Linked_List(struct Linked_List*,int);
void Swap_Numbers(struct Linked_List*,struct Linked_List*);
int Search_Number(struct Linked_List*,int);
void Reverse_Linked_List(struct Linked_List**);
void Sum_Linked_List(struct Linked_List*,int*);
void Multiplication_Even_Indexs_Linked_List(struct Linked_List*,int*,int);
void Print_Linked_List(struct Linked_List*);
```

```
1  #include "Linked_List.h"
2
3  void Build_Linked_List(struct Linked_List** Building, int Move_To_Data){    //
     Byliding the list
4      struct Linked_List* Temp_Struct = (struct Linked_List*)malloc(sizeof(struct
         Linked_List));
5      Temp_Struct->Data = Move_To_Data;   //Entring number to data
6      Temp_Struct->Head_To_Tail = *Building;  //Entering the struct from main to
         const struct when we sending the struct adders from main
7      *Building = Temp_Struct;
8  }
9
10 void Sort_Linked_List(struct Linked_List* Start_To_End,int count) { //Puting
     the number in right order
11     struct Linked_List* Temp_1;
12     struct Linked_List* Temp_2 = NULL;
13     for (int i = 0; i < count; i++){
14         Temp_1 = Start_To_End;  //Struct=Struct
15         while (Temp_1->Head_To_Tail != Temp_2) {    //while first numeber not
             eqvual other number
16             if (Temp_1->Data > Temp_1->Head_To_Tail->Data)
17                 Swap_Numbers(Temp_1, Temp_1->Head_To_Tail); //Making swap
18             Temp_1 = Temp_1->Head_To_Tail; //moving to next number
19         }
20         Temp_2 = Temp_1; // changing the other temp to last temp its for
             departure plan
21     }
22 }
23
24 void Swap_Numbers(struct Linked_List* Number_1, struct Linked_List* Number_2)
     {  //Regular swap
25     int temp = Number_1->Data;
26     Number_1->Data = Number_2->Data;
27     Number_2->Data = temp;
28 }
29
30 int Search_Number(struct Linked_List* From_Start_To_End, int Index){    //
     Search number in the list
31     struct Linked_List* current = From_Start_To_End;
32     int count = 0;
33     while (current != NULL){    //If list end
34         count++;
35         if (current->Data == Index) //If the number is in the index
36             return count;
37         current = current->Head_To_Tail; //To the next number
38     }
39     return 0; //Not in the list
40 }
41
42 void Reverse_Linked_List(struct Linked_List** From_Start_To_End){   //Reverse
     the list
43     struct Linked_List* Right_Order = *From_Start_To_End;
44     struct Linked_List* Bad_Order = NULL;
45     struct Linked_List* Right_Order_Next = NULL;
46     while (Right_Order != NULL) {   //Ending the reverse
47         Right_Order_Next = Right_Order->Head_To_Tail;   //We start process like
```

```
            a funcation that make swap ,
48          Right_Order->Head_To_Tail = Bad_Order;          //taking the last
              number and puting in the start ,
49          Bad_Order = Right_Order;                        //and the first to end
              by changing they index like ,
50          Right_Order = Right_Order_Next;                 //in swap funcation (2
              funcation above)
51      }
52      *From_Start_To_End = Bad_Order;
53  }
54
55  void Sum_Linked_List(struct Linked_List* From_Start_To_End, int* Sum) { //
      regular sum faction
56      if (!From_Start_To_End)
57          return;
58      Sum_Linked_List(From_Start_To_End->Head_To_Tail, Sum);
59      *Sum += From_Start_To_End->Data;
60  }
61
62  void Multiplication_Even_Indexs_Linked_List(struct Linked_List*
      From_Start_To_End, int* Sum, int Index) { //Multiplicatio funcation like sum
      funcation above but only when index are even
63      if (!From_Start_To_End)
64          return;
65      Index++;
66      if (Index % 2 == 0)
67          *Sum *= From_Start_To_End->Data;
68      Multiplication_Even_Indexs_Linked_List(From_Start_To_End->Head_To_Tail,
          Sum, Index);
69  }
70
71  void Print_Linked_List(struct Linked_List* From_Start_To_End) { //regular Print
      faction
72      struct Linked_List* temp = From_Start_To_End;
73      int count = 0;
74      printf("\nThe List is: ");
75      while (temp != NULL)
76      {
77          printf("%d ", temp->Data);
78          temp = temp->Head_To_Tail;
79          count++;
80      }
81      printf("\nThe Lenth is: %d", count);
82  }
83
84
```