```c
//Linked list with searching and counting
#include<stdio.h>
#include<stdlib.h>
struct node
{
    int info;
    struct node *link;
};
typedef struct node *NODE;
NODE getnode()
{
    NODE x;
    x=(NODE)malloc(sizeof(struct node));
    if(x==NULL)
    {
        printf("memory full\n");
        exit(0);
    }
    return x;
}
void freenode(NODE x)
{
    free(x);
}
NODE insert_front(NODE first,int item)
{
    NODE temp;
    temp=getnode();
    temp->info=item;
    temp->link=NULL;
    if(first==NULL)
        return temp;
    temp->link=first;
    first=temp;
    return first;
}
NODE delete_rear(NODE first)
```

```
NODE delete_rear(NODE first)
{
    NODE cur,prev;
    if(first==NULL)
    {
        printf("List is empty cannot delete\n");
        return first;
    }
    if(first->link==NULL)
    {
        printf("Item deleted is %d\n",first->info);
        free(first);
        return NULL;
    }
    prev=NULL;
    cur=first;
    while(cur->link!=NULL)
    {
        prev=cur;
        cur=cur->link;
    }
    printf("Item deleted is %d\n",cur->info);
    free(cur);
    prev->link=NULL;
    return first;
}
void count(NODE first)
{
    int c=0;
    if(first==NULL){
        printf("Empty list\n");
        return;
    }
    NODE temp;
    temp=first;
    while(temp!=NULL){
        c++;
```

```
2365            c++;
2366            temp=temp->link;
2367        }
2368        printf("Total items = %d\n",c);
2369    }
2370    void search(int key,NODE first)
2371    {
2372        NODE cur;
2373        int j=0;
2374        if(first==NULL){
2375            printf("List is empty\n");
2376            return;
2377        }
2378        cur=first;
2379        while(cur!=NULL){
2380            j++;
2381            if(key==cur->info)
2382                break;
2383            cur=cur->link;
2384        }
2385        if(cur==NULL){
2386            printf("Search is unsuccessful\n");
2387            return;
2388        }
2389        printf("Search is successful and position of element is %d\n",j);
2390    }
2391    void swap(NODE a,NODE b)
2392    {
2393        int temp = a->info;
2394        a->info = b->info;
2395        b->info = temp;
2396    }
2397    void bubbleSort(NODE first)
2398    {
2399        int swapped;
2400        NODE cur;
2401        NODE prev = NULL;
```

```
2401        NODE prev = NULL;
2402
2403        if (first == NULL){
2404            printf("Empty Linked List\n");
2405            return;
2406        }
2407        do{
2408            swapped = 0;
2409            cur = first;
2410            while (cur->link != prev){
2411                if (cur->info > cur->link->info){
2412                    swap(cur, cur->link);
2413                    swapped = 1;
2414                }
2415                cur = cur->link;
2416            }
2417            prev=cur;
2418        }
2419        while (swapped);
2420    }
2421    void display(NODE first)
2422    {
2423        NODE temp;
2424        if(first==NULL)
2425            printf("List empty cannot display items\n");
2426        for(temp=first;temp!=NULL;temp=temp->link)
2427        {
2428            printf("%d\n",temp->info);
2429        }
2430    }
2431    int main()
2432    {
2433        int item,choice,key;
2434        NODE first=NULL;
2435        for(;;){
2436            printf("\n1:Insert_front\n2.Delete_rear\n3.count\n4.Search\n5.Sort\n6.Display_list\n7.Exit\n");
2437            printf("Enter the choice : ");
```

```
2437        printf("Enter the choice : ");
2438        scanf("%d",&choice);
2439        switch(choice)
2440        {
2441            case 1:printf("enter the item at front-end : ");
2442                scanf("%d",&item);
2443                first=insert_front(first,item);
2444                break;
2445            case 2:first=delete_rear(first);
2446                break;
2447            case 3:count(first);
2448                break;
2449            case 4:printf("Enter the item to be searched : ");
2450                scanf("%d",&key);
2451                search(key,first);
2452                break;
2453            case 5:bubbleSort(first);
2454                printf("Items In Sorted Order are\n");
2455                display(first);
2456                break;
2457            case 6:printf("List : \n");
2458                display(first);
2459                break;
2460            case 7:exit(0);
2461            default:printf("Enter correct instruction!!!\n");
2462                break;
2463        }
2464    }
2465    return 0;
2466 }
2467
2468
```

```
"C:\Users\Shreshtha Aggarwal\Desktop\1stpro\bin\Debug\1stpro.exe"

1:Insert_front
2.Delete_rear
3.count
4.Search
5.Sort
6.Display_list
7.Exit
Enter the choice : 1
enter the item at front end : 8

1:Insert_front
2.Delete_rear
3.count
4.Search
5.Sort
6.Display_list
7.Exit
Enter the choice : 1
enter the item at front end : 7

1:Insert_front
2.Delete_rear
3.count
4.Search
5.Sort
6.Display_list
7.Exit
Enter the choice : 1
enter the item at front end : 9

1:Insert_front
2.Delete_rear
3.count
4.Search
5.Sort
6.Display_list
7.Exit
Enter the choice : 6
list :
9
7
8

1:Insert_front
2.Delete_rear
3.count
4.Search
5.Sort
6.Display_list
```

"C:\Users\Shreshtha Aggarwal\Desktop\1stpro\bin\Debug\1stpro.exe"

```
7.Exit
Enter the choice : 5
Items In Sorted Order are
7
8
9

1:Insert_front
2.Delete_rear
3.count
4.Search
5.Sort
6.Display_list
7.Exit
Enter the choice : 2
Item deleted is 9

1:Insert_front
2.Delete_rear
3.count
4.Search
5.Sort
6.Display_list
7.Exit
Enter the choice : 3
Total items = 2

1:Insert_front
2.Delete_rear
3.count
4.Search
5.Sort
6.Display_list
7.Exit
Enter the choice : 4
Enter the item to be searched : 8
Search is successful and position of element is 2

1:Insert_front
2.Delete_rear
3.count
4.Search
5.Sort
6.Display_list
7.Exit
Enter the choice : 7

Process returned 0 (0x0)   execution time : 48.615 s
Press any key to continue.
```