07-12-20

Lab-8 (Week - 9)

Date
Page 22

Q> Program to implement linked list : insert-front, delete rear, display, count items, search, order.

```
#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 (" Mem full (x "); 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 deleteRear(Node first)
{   Node cur, prev;
    if (first == NULL) {
        printf("Empty list\n");
        return first; }
    if (first -> link == NULL) {
        printf("Del item = %d\n", first -> info);
        free(first);
        return NULL; }
    prev = NULL; cur = first;
    while (cur -> link != NULL) {
            prev = cur;
            cur = cur -> link; }
    printf("Del item = %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) {
```

```
        x++;
        temp = temp -> link; }
    printf("Total items = %d", x); }
void search (int key, Node first){
    Node cur;
    int j = 0;
    if (first == NULL) {
        printf("Empty list \n");
        return; }
    cur = first;
    while (cur != NULL) { j++;
        if ( key == cur -> info) { break; }
        cur = cur -> link; }
    if ( cur == NULL) {
        printf("search is failed");
        return; }
    printf("search successful & pos = %d", j); }
void swap (Node a, Node b) {
    int temp = a -> info;
    a -> info = b -> info;
    b -> info = temp; }
void bubblesort (Node first) {
    int s;
    Node cur, prev = NULL;
    if (first == NULL) {
```

```
        printf("Empty list");
        return; }
    do {  s = 0;
          cur = first;
        while (cur->link != prev) {
          if(cur -> info > cur -> link -> info){
              swap (cur, cur -> link);
              s = 1; }
          cur = cur -> link;
        } prev = cur;
    } while (s); }
void display (Node first) {
    Node temp;
    if (first == NULL)
      {printf("list empty "); }
    for (temp = first; temp != NULL; temp = temp->link)
    {  printf ("%d", temp -> info); } }
int main () {
    int item, choice, key;
    Node first = NULL;
    for (;;) {
        printf ("1. Insert 2. Delete 3. count 4. search
                5. Sort 6. Display > Exit \n");
        printf (" Enter choice : ");
        scanf ("%d", &choice);
```

```
switch (choice) {
    case 1: printf (" Enter item ");
            scanf ("%d", &item);
            first = insert_front (first, item);
            break;
    case 2: first = delete_rear (first);
            break;
    case 3: count (first);
            break;
    case 4: printf ("Enter item for search: ");
            scanf ("%d", &key);
            search (key, first);
            break;
    case 5: bubblesort (first);
            printf (" Items in sorted : ");
            display (first);
            break;
    case 6: printf (" List : \n");
            display (first);
            break;
    case 7: exit (0);
    default: printf (" Enter correct inst-
                    ruction ");
             break; } }

    return 0;
}
```

Date
Page 57

O/P->

1. Insert 2. Delete 3. Display 4. Sort 5. Count
6. Search 7. Exit

Enter choice: 1

Enter item: 23