Experiment 7 : (single linked list)

Q1) Write a C program to perform the operations on a single linked list:

i) Insertion at beginning, ii) Deletion of 1st node iii) display all nodes

Q2) Write a C program to perform the operations on a single linked list:

i) insertion at end, i) deletion of last node iii) display all the nodes

Q3) Write a C program to perform the operations on a single linked list:

i) insertion of a new node after a node item ii) searching for a node item iii) deletion of a node item

iv) display all the nodes

**// C program for operations in the Singly Linked List**

#include <stdio.h>

#include <stdlib.h>

// Declaration of Node

struct node

{

int info;

struct node\* link;

};

struct node\* start = NULL;

**// Function to traverse the linked list**

void traverse()

{

struct node\* ptr;

if (start == NULL)

printf("\nList is empty or Underflow\n");

else

{

ptr = start;

while (ptr != NULL)

{

printf("%d ",ptr->info);

ptr = ptr->link;

}

}

}

**// Function to insert at the beginning of the linked list**

void insert\_beginning()

{

int item;

struct node\* fresh;

fresh = (struct node \*) malloc(sizeof(struct node));

if(fresh==NULL)

{

printf(“Overflow”);

exit(1);

}

printf("\nEnter the info : ");

scanf("%d", &item);

fresh->info = item;

fresh->link=NULL;

if(start == NULL)

{

start = fresh;

}

else

{

fresh->link = start;

start = fresh;

}

}

// **Function to insert at the end of the linked list**

void insert\_end()

{

int item;

struct node \*fresh, \*ptr;

fresh = (struct node \*) malloc(sizeof(struct node));

if(fresh==NULL)

{

printf(“Overflow”); exit(1);

}

printf("\nEnter the info : ");

scanf("%d", &item);

fresh->info = item;

fresh->link=NULL;

if(start == NULL)

start = fresh;

else

{

ptr = start;

while (ptr->link != NULL)

{

ptr = ptr->link;

}

ptr->link = fresh;

}

}

// **Function to insert at any specified position in the linked list**

void insert\_loc()

{

struct node \*ptr,\*prev,\*fresh;

int loc, item, i = 1;

fresh = (struct node \*) malloc(sizeof(struct node));

printf("\nEnter the info : ");

scanf("%d", &item);

fresh->info = item;

fresh->link = NULL

printf("\nEnter the location :");

scanf("%d", &loc);

ptr = start;

while ((i < loc) && (ptr!= NULL))

{

prev = ptr;

ptr = ptr->link;

i++;

}

prev->link = fresh;

fresh->link=ptr;

}

**// Function to delete the first node in Singly linked list**

void deletion\_beginning()

{

struct node\* ptr;

if (start == NULL)

{

printf("\nList is empty or Underflow\n");

}

else

{

ptr = start;

start = ptr->link;

free(ptr);

}

}

**// Function to delete last node of the linked list**

void deletion\_last()

{

struct node \*ptr, \*prev;

if (start == NULL)

{

printf("\nList is Empty or Underflow\n");

}

else

{

ptr = start;

while (ptr->link != NULL)

{

prev = ptr;

ptr = ptr->link;

}

prev->link = NULL;

free(ptr);

}

}

**// Function to delete from any specified position from the linked list**

void deletion\_loc()

{

struct node \*ptr,\*prev;

int loc, item, i = 1;

printf("\nEnter the location :");

scanf("%d", &loc);

ptr = start;

while ((i < loc) && (ptr!= NULL))

{

prev = ptr;

ptr = ptr->link;

i++;

}

prev->link = ptr->link;

}

int main()

{

int choice;

while (1)

{

printf("\n\t1 Traverse the list\n");

printf("\t2 Insertion at beginning\n");

printf("\t3 Insertion at end\n");

printf("\t4 Insertion at specific location\n");

printf("\t5 Deletion of first node\n");

printf("\t6 Deletion of last node\n");

printf("\t7 Deletion from specific location\n");

printf("\t8 Exit\n");

printf("\nEnter Choice :\n");

scanf("%d", &choice);

switch (choice)

{

case 1:

traverse();

break;

case 2:

insert\_beginning();

break;

case 3:

insert\_end();

break;

case 4:

insert\_loc();

break;

case 5:

deletion\_beginning();

break;

case 6:

deletion\_last();

break;

case 7:

deletion\_loc();

break;

case 8:

exit(1);

default:

printf("Incorrect Choice\n");

}

} // End of while true loop

return 0;

}