Heaven's Light is Our Guide

Computer Science & Engineering
Rajshahi University of Engineering & Technology

Lab Manual

Module- 04

Course Title: Sessional based on CSE 1201
Course No. : CSE 1202

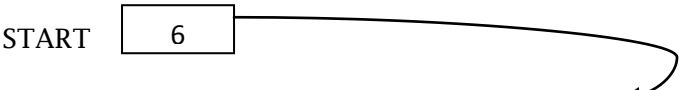
Experiment No. 4

Name of the Experiment: Linked Lists

Duration: 1 cycle

Background Study: Chapter 5 (Theory and Problems of Data Structures Written by Seymour Lipschutz)

Let we have a LIST



	1	2	3	4	5	6	7	8	9	10	11	12
INFO			U	E	C	R	T	E			S	
LINK		∧	8	∧	11	3	5	7	∧	∧	4	∧

*∧ indicates NULL

Fig. 4.1

Problem I: Traversing a linked list.
Algorithm4.1: LIST is a linked list in memory. This algorithm traverses LIST, applying an operation PROCESS to each element of LIST. The variable PTR points to a node currently being processed.

1. Set PTR:=START
2. Repeat steps 3 and 4 while PTR≠NULL
3. Apply PROCESS to INFO[PTR]
4. SET PTR := LINK[PTR]
   [End of Repeat 2 loop]
5. Exit

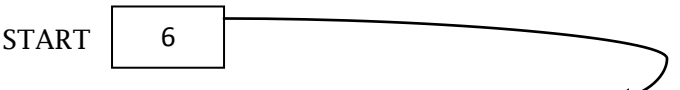
Flow Chart: Draw a flow chart.
Exercise:

1) Print the information at each node of the list
2) Find the total number of elements in a list

Problem II: Searching a linked list (LIST is unsorted).
Algorithm4.2: SEARCH (INFO, LINK, START, ITEM, LOC)
LIST is a linked list in memory. This algorithm finds the location LOC of the node where ITEM first appears in LIST, or sets LOC-NULL.

1. Set PTR:= START
2. Repeat steps 3 and 4 while PTR≠NULL
3. If ITEM = INFO[PTR] then:
   Set LOC:=PTR and Exit.
   Else:
   SET PTR := LINK[PTR]
   [End of If statement]
   [End of Repeat 2 loop]
4. [Search is unsuccessful] Set LOC:=NULL
5. Exit

Flow Chart: Draw a flow chart.



	1	2	3	4	5	6	7	8	9	10	11	12
INFO			201	402	325	101	301	251			385	
LINK		∧	8	∧	11	3	5	7	∧	∧	4	∧

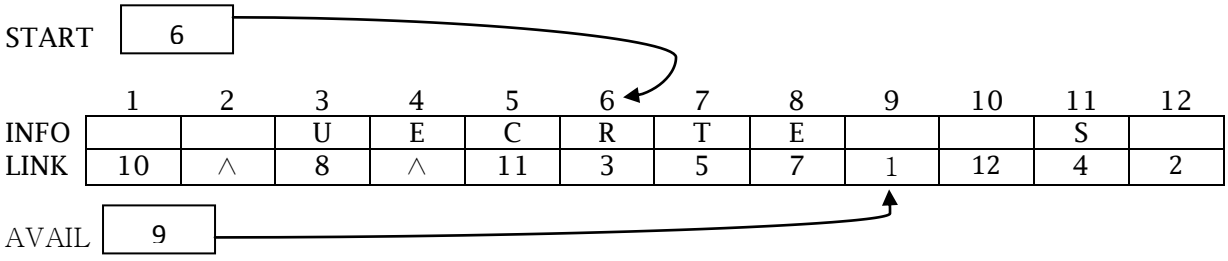
*∧ indicates NULL

Fig 4.2

Problem III: Searching a linked list (LIST is sorted in ascending order, see fig. 4.1).
Algorithm4.3: SRCHSL (INFO, LINK, START, ITEM, LOC)
LIST is a sorted linked list in memory. This algorithm finds the location LOC of the node where ITEM first appears in LIST, or sets LOC-NULL.

1. Set PTR:= START
2. Repeat steps 3 and 4 while PTR≠NULL
3. If ITEM<INFO [PTR], then:
   Set PTR:= LINK[PTR]
   Else if ITEM = INFO [PTR] then:
   Set LOC:=PTR and Exit.
   Else:
   SET LOC:= NULL, and Exit
   [End of If statement]
   [End of Repeat 2 loop]
4. [Search is unsuccessful] Set LOC:=NULL
5. Exit

Flow Chart: Draw a flow chart.



*∧ indicates NULL

Fig. 4.3

Problem IV: Insertion into a Linked List (beginning of a list)
Algorithm4.4: INSFIRST (INFO, LINK, START, AVAIL, ITEM)
This algorithm inserts ITEM as the first node in the list.

1. [OVERFLOW?] If AVAIL = NULL, then Write: OVERFLOW, and Exit
2. [Remove first node from AVAIL list]
   Set NEW := AVAIL and AVAIL:= LINK [AVAIL]
3. Set INFO [NEW] := ITEM.
4. Set LINK [NEW] := START.
5. Set START := NEW.
6. Exit

Flow Chart: Draw a flow chart.

Problem V: Insertion into a Linked List (Inserting after a given node)
Algorithm4.5: INSLOC (INFO, LINK, START, AVAIL, LOC, ITEM)
This algorithm inserts ITEM so that ITEM follows the node with location LOC or insert ITEM as the first node when LOC = NULL.

1. [OVERFLOW?] If AVAIL = NULL, then Write: OVERFLOW, and Exit
2. [Remove first node from AVAIL list]
   Set NEW := AVAIL and AVAIL:= LINK [AVAIL]
3. Set INFO [NEW] := ITEM.
4. If LOC = NULL, then:
   Set LINK [NEW] := START and START := NEW.
   Else:
   Set LINK [NEW] := LINK [LOC] and LINK [LOC] := NEW.
   [End of If statement]
5. Exit

Flow Chart: Draw a flow chart.

Problem V: Insertion into a Linked List (Inserting into a Sorted Linked List)
Algorithm4.6: FINDA (INFO, LINK, START, LOC, ITEM)
This algorithm finds the location LOC of the last node in a sorted list such that INFO [LOC] < ITEM or sets LOC = NULL.

1. [LIST EMPTY?] If START = NULL, then: Set LOC := NULL, and Return.
2. [Special Case?] If ITEM <INFO [START], Then: Set LOC := NULL, and Return.
3. Set SAVE := START and PTR:= LINK [START]
4. Repeat steps 5 and 6 while PTR ≠ NULL.
5. If ITEM<INFO [PTR], then:
   Set LOC := SAVE, and Return.
   [End of If statement]
6. Set SAVE := PTR and PTR := LINK [PTR].
   [End of Repeat 4 loop]
7. Set LOC := SAVE.
8. Return.

Algorithm4.7: INSSRT (INFO, LINK, START, AVAIL, ITEM)
This algorithm inserts ITEM into a sorted list.

1. Call **FINDA (INFO, LINK, START, LOC, ITEM).**
2. Call **INSLOC (INFO, LINK, START, AVAIL, LOC, ITEM)**
3. Exit.

Flow Chart: Draw a flow chart.

MORE PROBLEMS

1. Programming Problems of Chapter 5 of "Data Structures" by Seymour Lipschutz.

LAB REPORT: You have to submit all assigned problems in next lab.