Assignment 3

1. Implement the following functions of ADT Linked List using singly linked list as a header file:

init l(cur) – initialise a list

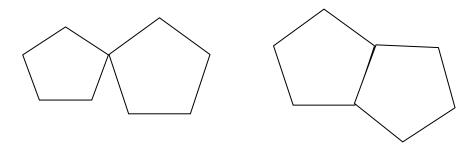
empty_l(head) - boolean function to return true if list pointed to by head is empty
atend_l(cur) - boolean function to return true if cur points to the last node in the list
insert_front(target, head) - insert the node pointed to by target as the first node of the list
pointed to by head

insert_after(target, prev) - insert the node pointed to by target after the node pointed to by prev
delete_front(head) - delete the first element of the list pointed to by head
delete_after(prev) - delete the node after the one pointed to by prev

- 2. Read integers from a file and arrange them in a linked list (a) in the order they are read, (b) in reverse order. Show the lists by printing by developing a function Print_list. The functions for (a) is Build_list and for (b) is Build_list_reverse.
- 3. Implement the following functions in a menu-driven C program using the data structure operation of Singly Linked List in the header file developed in problem 1:
- a) print a list (i) in the same order, (ii) in the reverse order.
- b) find the size of a list in number of nodes
- c) check whether two lists are equal
- d) search for a key in (i) an unordered list, (ii) an ordered list (Return the node if key is found and delete the node from original list)
- e) append a list at the end of another list.
- f) delete the nth node, last node and first node of a list.
- g) check whether a list is ordered
- h) merge two sorted lists
- i) insert a target node in the beginning, before a specified node and at the end of the list (sorted and unsorted).
- j) remove duplicates from a linked list (sorted and unsorted)
- k) swap elements of a list pairwise
- 1) move last element to front of a list

- m) delete alternate nodes of a list
- n) rotate a list
- o) delete a list.
- p) reverse a list.
- q) sort a list.
- 4. Write all the above operations of Single Linked List for the implementation using array. You need to develop Build_list and Build_list_reverse, as well as Print_list.
- 5. Repeat problems 1 and 3 for a circular single linked list, doubly linked list and circular doubly linked list. You need to develop Build_list and Build_list_reverse, as well as Print_list for each case.
- 6. Implement an application to find out the Inverted Index of a set of text files, given a set of keywords. Create a set of 6 text files having the keywords in different positions in the text files. The keywords may occur multiple times in a file. The inverted index file will list the key words along with the filenames in which they occur and how many times they occur in that file.
- 7. Write an application for adding, subtracting and multiplying very large numbers, say more than 70-digit integers, using (a) arrays and (b) linked lists to represent the large integers.
- 8. Given two polygons, say pentagons, find out whether they intersect or not.

Touching polygons:



Intersecting polygons:

