Total Point: 25 points

Consider the following declaration for a “**node**” in a singly-linked list and implementation for the void function “**fun**” when answering this question, both written in C:

```
struct node
{
  int data;
  node * next;
};

void fun(struct node* start)
{
  if(start == NULL)
     return;
  printf("%d  ", start->data);

  if(start->next != NULL )
     fun(start->next->next);
  printf("%d  ", start->data);
}
```

Write (implement) the following functions:

1. A void function called “**add_back**” that has three formal parameters: A node pointer to the front of the list called “**front**” that is passed by reference; a node pointer to the back of the class called “**back**” that is passed by reference; and an integer called “**x**” that holds the value to be added to the list (x is passed by value).

2. The **main program** which creates a single-link list with a pointer to the front and a pointer to the back of the list. The main should create the following list:

1->2->3->4->5->6

3. Convert the “**node**” declaration and function “**fun**” implementation into a C++ declaration and implementation, respectively. (hint: use class instead of struct, and change printf’s to cout’s)

Call the program “**know_recursion_and_lists.cpp**”. Remember, to comment your program, and to submit the program to Canvas before the due date and time.

See the sample main program below (on next page):

```
int main()
{
        node* front = 0;
        node* back = 0;

        for (int i = 1; i <= 6; i++)
        {
                insert(front, back, i);
        }

        fun(front);
        return 0;
}
```