1. **Hãy xây dựng cấu trúc dữ liệu cho danh sách liên kết đơn để lưu trữ các số nguyên và viết các thao tác cơ bản như mô tả sau: (3.5 điểm)**
   - Khởi tạo: *public Node(int value); public LinkedList();*
   - Thêm một phần tử vào cuối danh sách: *public void AddLast(int value);*
   - Thêm một phần tử vào sau một phần tử khác trong danh sách: *public void AddAfter(Node pre, int value);*
   - Xóa một phần tử đầu danh sách: *public void RemoveFirst();*
   - Tìm kiếm một phần tử có dữ liệu là value trong danh sách: *public Node Find(int value);*
   - Sắp xếp danh sách tăng dần dùng giải thuật Selection Sort: *public void SelectionSort();*
2. **Trong chương trình chính, sử dụng class LinkedList ở câu 1, tạo ra 2 danh sách liên kết l1 và l2. Viết và gọi thực thi các chức năng sau: (5.5 điểm).**
   - **a.** Viết hàm nhập danh sách N số nguyên dương (có thể Random) bằng cách thêm từng phần tử vào cuối danh sách. Gọi thực thi cho 2 danh sách l1, l2.
   - **b.** Viết hàm xuất danh sách. Gọi thực thi cho 2 danh sách l1, l2.
   - **c.** Viết hàm thêm một phần tử (có dữ liệu là giá trị lớn nhất trong l2) vào sau phần tử có dữ liệu là số nguyên tố đầu tiên trong danh sách l1.
     Lưu ý: Nếu l1 không có số nguyên tố thì không thêm.
     Ví dụ:

Input	Output
L1: 4 5 4 7 5 6 L2: 3 2 **8** 4 5 *//Max =* ***8***	L1: 4 5 **8** 4 7 5 6

   - **d.** Viết hàm xóa k phần tử ở đầu danh sách (k được nhập từ bàn phím, với 0 < k < số phần tử của danh sách hiện hành). Gọi thực thi cho danh sách l2.
     Ví dụ:

Input	Output
L2: ***2 7 5 3*** 4 6 7 K = 4	L2: 4 6 7

   - **e.** Viết hàm tạo ra danh sách l3 bao gồm các phần tử chỉ có trong danh sách l1 mà không có trong danh sách l2. (Giả sử các phần tử trong l1, l2 đều có giá trị phân biệt)
     Ví dụ:

Input	Output
L1: 3 4 5 2 7 6 1 L2: 5 6 2 8 9 7	L3: 3 4 1

f. Sắp xếp danh sách l2 tăng dần.

-Hết-