Data una lista L1 definire una funzione **ricorsiva** in C che restituisca il numero occorrenze del valore massimo della lista (se la lista è vuota vuota, restituire 0). Si discuta la complessita’ in spazio e tempo della funzione.

Esempio: L1: 4 🡪 4🡪 23 🡪 8 🡪 6 🡪 23🡪 86 🡪 86 🡪 23 🡪 86

OUTPUT: 3

Data una lista L1 definire una funzione **ricorsiva** in C che 1 se la lista è ordinana in modo strettamente crescente (ovvero un elemento è minore del successivo) altrimenti 0.Si discuta la complessita’ in spazio e tempo della funzione.

Esempio: L1: 4 🡪 5🡪 23 🡪 80 🡪 86 🡪 86

OUTPUT: 0

Date due liste L1 ed L2 definire una funzione **ricorsiva** in C che calcoli il numero di elementi uguali nella stessa posizione. Si discuta la complessita’ in spazio e tempo della funzione.

Esempio: L1: 3 🡪 42🡪 23 🡪 17 🡪 8 🡪 16 🡪 1

L2: 3 🡪 42🡪20 🡪 17 🡪 2 🡪 1 🡪 2 🡪16 🡪 7 🡪 2

OUTPUT: 3

Date due liste L1 ed L2 definire una funzione **ricorsiva** in C (fino alla fine della piu’corta)che restituisca in output una nuova lista (si utilizzi la funzione newnode vista a lezione) contenente, posizione per posizione, la differenza della (contenuto del) nodo di L1e di L2, ma solo se tale valore è multiplo della posizione dei nodi. Si discuta la complessita’ in spazio e tempo della funzione.

Esempio: L1: 3 🡪 4🡪 3 🡪 7 🡪 8 🡪 6 🡪 1

L2: 1 🡪 4🡪2 🡪 7 🡪 2 🡪 12 🡪 2 🡪16 🡪 7 🡪 2

OUTPUT: 2🡪0🡪0🡪6