**Doubly Linked Lists and Sets**

We implement a set of functions based on the doubly linked lists.

The functions are as follows:

1.        DLList \*CreateDLListFromFileDlist(const char \*filename). This function creates a doubly linked list of integers by reading all integers from a text file named filename, and returns a pointer to the doubly linked list created. We assume that adjacent integers in the file filename are separated by one or more white space characters or a new line character. If filename is “stdin”, CreateDLListFromFileDlist (“stdin”) creates a doubly linked list by reading all integers from the standard input. We assume that each input line is an integer and an empty line denotes end of input.

2.       void printDLList(DLList \*u ). This function prints all the elements (integers) of a doubly linked list pointed by u in the order they appear in the list on the standard output, one element per line.

3.       DLList \*cloneList(DLList \*u). This function creates an identical copy of a doubly linked list u and returns a pointer to the list cloned.

4.       DLList \*setUnion(DLList \*u, DLList \*v). This function computes the union of the two sets of integers that are stored in the doubly linked lists pointed by u and v, respectively, and returns a pointer to the doubly linked list that stores the union. Each element (int) of a set is stored in a node of the corresponding doubly linked list.

       Given two sets A and B, the union of A and B is a set that contains all the distinct element of A and B. For example, assuming that A={2, 8, 5, 7} and B={5, 9, 6, 7},  A ꓴ B={2, 8, 5, 7, 9, 6}. Note that in a set, all the integers are not necessarily sorted.

5.      DLList \*setIntersection(DLList \*u, DLList \*v). This function computes the intersection of the two sets of integers that are stored in the doubly linked lists pointed by u and v, respectively, and returns a pointer to the doubly linked list that stores the intersection. Each element (int) of a set is stored in a node of the corresponding doubly linked list.

       For simplicity, we assume that all the elements of each input set are distinct for both set union and set intersection. Therefore, we do not need to check if a set contains duplicates.

Given two sets A and B, the intersection of A and B is a set that contains all the elements of A that are also in B. For example, assuming that A={2, 8, 5, 7} and B={5, 9, 6, 7},  A ꓵ B={5, 7}.

6.       void freeDLList(DLList \*u). This function frees the space occupied by all the nodes of the doubly linked list pointed by u.