Question 1

Write a program that creates a doubly linked list of **Points** in the 2-D Cartesian plane. Basically, the linked list will store the x and y coordinate of each point.
Your program should support the following functions: createlist, insertbeg, insertend, deletebeg, deletelast, deletelist, printlist
Note: deleteList should delete all the elements of the list and then delete the list pointer as well.
All functions other than the deletelist and printlist should take **O(1) time**.

The structure should be something like this:

```
typedef struct _point {
        int x, y;
} point;

typedef struct _node {
        point a;
        struct _node* next;
        struct _node* prev;
} node;
```

Input:
Since you're expected to implement a doubly linked list, initial number of elements are not specified. Your program should first create an emptylist and take input until the user enters '0' (i.e. zero, without the quotes).
In other words, the program shall successfully terminate on receiving 0 as the input.
Each line of input could be of one of the operations, optionally followed by the coordinates (x and y integer values).

- createlist
  Initialises a doubly linked list. If the list is already created, do nothing.
- insertbeg x y
  Inserts a new node with x- and y-coordinate at the beginning of the linked list. If the list has already been not created (does not exist), call createlist.
- insertend x y
  Inserts a new node with x- and y-coordinate at the end of the linked list. If the list has already been not created (does not exist), call createlist.
- deletebeg
  Deletes the beginning (first) node and its corresponding point coordinates. If the list is empty, do nothing.
- deletelast
  Deletes the last node and its corresponding point coordinates. If the list is empty, do nothing.

- deletelist
  Deletes the entire list and frees up space (Deletes the pointer as well). If the list has already been deleted, do nothing.
- printlist K
  If K = 0, then print the entire forward list (from head to tail), or else if K = 1, print the list in reverse order (from tail to head)
- 0
  Termination of the input file

Output:
No output is required for any function, except printlist function.
For each 'printlist K' function, print the list in forward direction if K = 0, otherwise print it in reverse direction.
Each node shall be printed in a new line, with each line containing space-separated x- and y-coordinates. If the list is empty, then 'NULL' (without the quotes) shall be printed.

After printing the list each time (i.e. after every test case), a **blank line** shall be printed. (Refer sample input output)

(So that implies that for the case of 'NULL', the print statement shall be 'printf("NULL**\n\n**");'. For the other test cases, an extra '\n' shall be inserted after the complete output of that particular case)

Constraints:
1 ≤ X, Y ≤ 10^9

Sample Input:
```
createlist
insertbeg 6 7
insertend 8 5
insertend 4 6
insertbeg 9 9
printlist 0
deletebeg
deletelast
insertend 1 2
printlist 1
deletelist
printlist 0
createlist
insertbeg 5 5
printlist 1
0
```

Sample Output:

```
9 9
6 7
8 5
4 6

1 2
8 5
6 7

NULL

5 5
```