Q7.

For this problem, you need to know how to implement linked lists.
You must use the template to do this lab.

You should implement the following functions by yourself within the class:

Operations	**Description**
A. void push_back(int x)	Insert a node to the end of the linked list, the node's value is x. E.g. $4 \rightarrow 8 \rightarrow 7 \rightarrow 6 \rightarrow null$ list.push_back(3) $4 \rightarrow 8 \rightarrow 7 \rightarrow 6 \rightarrow 3 \rightarrow null$
B. void push_front(int x)	Insert a node to the front of the linked list, the node's value is x. E.g. $8 \rightarrow 7 \rightarrow 6 \rightarrow 3 \rightarrow null$ list.push_front(4) $4 \rightarrow 8 \rightarrow 7 \rightarrow 6 \rightarrow 3 \rightarrow null$
C. void insert(int index,int x)	Insert a node to the linked list at position **index**, the node's value is **x**. The index of the first node in the linked list is 0. If the index is out of the range, do nothing. E.g. $6 \rightarrow 2 \rightarrow 7 \rightarrow 4 \rightarrow null$ list.insert(1, 8) $6 \rightarrow 8 \rightarrow 2 \rightarrow 7 \rightarrow 4 \rightarrow null$ list.insert(0, 1) $1 \rightarrow 6 \rightarrow 8 \rightarrow 2 \rightarrow 7 \rightarrow 4 \rightarrow null$ list.insert(6, 5) $1 \rightarrow 6 \rightarrow 8 \rightarrow 2 \rightarrow 7 \rightarrow 4 \rightarrow 5 \rightarrow null$
D. void remove(int index)	Remove the node with index **index** in the linked list. If the index is out of the range, do nothing. E.g. $19 \rightarrow 11 \rightarrow 45 \rightarrow 36 \rightarrow 14 \rightarrow null$ list.remove(0) $11 \rightarrow 45 \rightarrow 36 \rightarrow 14 \rightarrow null$ list.remove(2) $11 \rightarrow 45 \rightarrow 14 \rightarrow null$
E. void reverse()	Reverse the linked list.

	E.g. $1 \rightarrow 3 \rightarrow 5 \rightarrow 7 \rightarrow 9 \rightarrow null$ list.reverse() $9 \rightarrow 7 \rightarrow 5 \rightarrow 3 \rightarrow 1 \rightarrow null$
F. void swap(int index1, int index2)	Swap the two nodes which are at index1 and index2 respectively in the linked list. If one of the indices is out of range, do nothing. E.g. $0 \rightarrow 2 \rightarrow 4 \rightarrow 6 \rightarrow 8 \rightarrow null$ list.swap(0,1) $2 \rightarrow 0 \rightarrow 4 \rightarrow 6 \rightarrow 8 \rightarrow null$ list.swap(2,4) $2 \rightarrow 0 \rightarrow 8 \rightarrow 6 \rightarrow 4 \rightarrow null$ list.swap(3,3) $2 \rightarrow 0 \rightarrow 8 \rightarrow 6 \rightarrow 4 \rightarrow null$
G. void print()	Print all the elements in the linked list in order. E.g. $4 \rightarrow 8 \rightarrow 7 \rightarrow 6 \rightarrow 3 \rightarrow null$, you need to output **[4 8 7 6 3]**.

Input Format

Each line of inputs contains a different number of operations.
There will be a character representing the operation (b, f, i, d, r, s).

“ b ”: represent push_back(…)
There is a number after ‘ b ’, and that number is what we want to add to the end of the list.

“ f ”: represent push_front(…)
There is a number after ‘ f ’, and that number is what we want to add to the front of the list.

“ i ”: represent insert(…)
There are two numbers after ‘ i ’. The first number represents the index where we want to insert a node into the list, and the second number represents the value of the node we want to add to the list.

“ d ”: represent remove(…)
There is a number after ‘ d ’, and that number represents the index where we want to remove the node from the list.

“ r ”: represent reverse(…)
There is no number after ‘ r ’.

“ s ”: represent swap(…)
There are two numbers after ‘ s ’. The first number is the first index, and the second number is the second index.

Output Format

You must print all the content of the linked list in order after doing each test case.
See more detail from Sample Output.

Sample Input

```
f 1 b 2 d 0 d 0
b 1 f 2 i 0 3 r i 3 4
f 1 f 2 b 3 b 4 i 2 5 r
f 1 f 2 f 3 f 4 f 5 r
f 1 f 2 f 3 f 5 f 10 d 0 s 0 3
```

Sample Output

```
[]
[1 2 3 4]
[4 3 5 1 2]
[1 2 3 4 5]
[1 3 2 5]
```