WRITTEN 1.

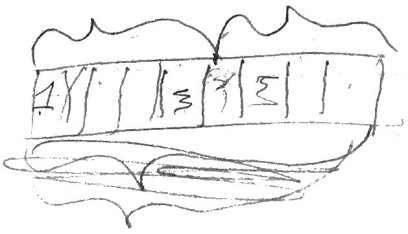
```
public static void printLots (ArrayList<Integer> L, ArrayList<Integer> P) {
	Iterator<Integer> iter = P.iterator();

	while(iter.hasNext()) {
		int indx = iter.next();
		//System.out.println(indx);
		if (indx > L.size()) {
			System.out.println("Out of bound");
			throw new IndexOutOfBoundsException();
		}

		Iterator<Integer> iterl = L.iterator();
		int counter = 0;
		while (iterl.hasNext()) {
			counter++;
			int l = iterl.next();
			if(counter==indx+1) {
				System.out.println(l);
			}
		}
	}
}
```

WRITTEN 2.

```
public static void Intersect (List<Integer> L, List<Integer> P) {

	Iterator<Integer> iter = P.iterator();
	while(iter.hasNext()) {
		int num = iter.next();
		if (L.contains(num)) {
			System.out.println(num);
		}
	}
}
```

Written #3

```
private int size;
private int top1, top2;
private int arr[];
```



```
public TwoStacks (int size) {
    size = size;
    arr = new int[size]
    top1 = 0;
    top2 = size - 1;

public void push1(int x) {
    if (top1 < top2) {
        arr[top1] = x;
        size top1 ++; ; }
    else {
        System.out.println("stackOverflow") } }

public void pop1() {
    if (top1 == 0) { System.out.println("Empty")
                     return null;
    else { top1 --;
        v = arr[top1];
        return (v) ; }

public void push2(int x) {
    if (top1 < top2) {
        arr[top2] = x;
        top2--; ; }
    else { print("stack Overflow") }
```

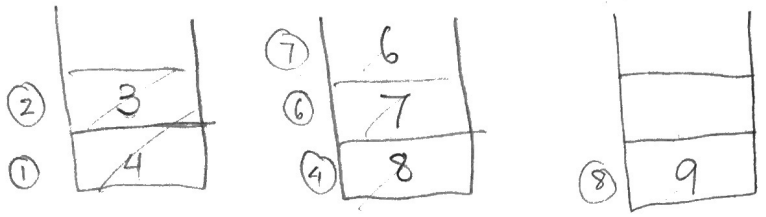
```
public T pop2( ) {
    if( top2 == size - 1) {
        Print ( " Stack 2 is Empty " ) ;
                return null;
    else {
        top2 ++ ;
        v = arr [top2] ;
        return(v)

public   isEmpty1 ( ) {
        if ( top 1 == 0 ) {
            return True; }
        return False ;

public   is Empty 2()
        if ( top2 == size-1) {
            return True ; }
        return False ;   }

public  size 1 ( ) {
        return top 1 - 1; }

Public size 2 ( ) {
        return size - top2 - 1 ; }

}
```

4) a)

5	9	6	7	2	8	1	3	4

moves:

in circles:



15	14	13	12	11	10	9	5	3
9	8	7	6	5	4	3	2	1

b) Example of non-arrangable train yard

1	9	8	7	6	5	4	3	2