```
private static <T> void insertInOrder(Queue<T> q, T x, Comparator<T> order) {
boolean in = false;
int idx = 0;
while (idx < q.length() && !in) {
if (order.compare(q.front(), x) > 0) {
q.enqueue(x);
in = true;
}
idx++;
T temp = q.dequeue();
q.enqueue(temp);
}
}
```

```
public void sort(Comparator<T> order) {

int n = order.length;

for (int j = 1; j < n; j++) {

int key = order[j];

int i = j-1;

while ((i > -1) && (order [i] > key)) {

order [i+1] = order [i];

i--;

}

order[i+1] = key;

}

}
```

Statement	Variable Values
`SortingMachine<Integer> sm = new SortingMachine1L<>(new IntegerGE());`	
	sm = `()`
`sm.add(0);`	
	sm = `(0(), ())`
`sm.add(2);`	
	sm = `(0(), (2(), ()))`
`sm.add(-1);`	
	sm = `(0((-1(), ()), (2(), ()))`
`sm.changeToExtractionMode();`	
	sm = `(0((-1(), ()), (2(), ()))`
`int i = sm.removeFirst();`	
	sm = `(0(), (2(), ()))` i = `-1`
`sm.clear();`	
	sm = `()` i = `-1`