3. The following code is for the Partition method used by the QUICKSORT algorithm:

```
procedure PARTITION(A, p, r)
  x ← A[r]; i ← p − 1
  for j ← p to r − 1
    do if A[j] ≤ x
      then i ← i + 1
        exchange A[i] ↔ A[j]
  exchange A[i + 1] ↔ A[r]
  return i + 1
```

Suppose that PARTITION$(A, 1, 6)$ is called over the array $A = [8, 4, 2, 7, 1, 5]$ (assuming the array indexes from 1). What is the result?

(a) $A = [4, 2, 1, 5, 8, 7]$ and 4 is returned.

(b) $A = [1, 2, 4, 5, 7, 8]$ and 5 is returned.

(c) $A = [4, 2, 1, 5, 8, 7]$ and 5 is returned.

(d) $A = [4, 2, 1, 5, 7, 8]$ and 4 is returned.

4. A `deque` (double-ended queue) is implemented using an array called `items` and left and right indices called `left` (an index to the leftmost item) and `right` (an index to the rightmost item) respectively. The deque is cyclic (or "wraps around") so that all space in the array can be used.

The method `pushLeft` adds an item to the left end of the deque and is implemented as follows:

```
public void pushLeft(char c) throws Overflow {
  if (!isFull()) {

    << missing code >>

  }
  else throw new Overflow(``Pushing to full deque''.)
}
```

Which of the following is a correct implementation of the missing lines:

(a)
```
left = (left-1)%items.length;
items[left] = c;
```

(b)
```
left = (left+1)%(items.length-1);
items[left] = c;
```

(c)
```
left = left-1;
if (left=right+1) left = (left-1)%items.length;
items[left] = c;
```

(d)
```
left = left-1;
if (left == -1) left = items.length-1;
items[left] = c;
```

MST

Ans 3

A = 8 4 2 7 1 5

x = A[6] = 5

i = 0

For j = 1 to 5

---

j=1

A[j] = 8 ; so ib will fail

No change to A, i

---

j = 2 ; A[j] = 2

ib ✓ ; i = 1

1 ↔ 2

A = 4 8 2 7 1 5

---

j = 3 A[j] = 2

ib ✓ i = 2

A = 4 2 8 7 1 5

---

j = 4 A[j] = 7

ib fails

---

j = 5 A[j] = 1

ib ✓

i = 3

A = 4 2 1 7 8 5

(a) is correct

swap A[4] and A[r]

A = [4, 2, 1, 5, 8, 7] ; Ret(i+1) = 4