10. You are trying to write a code for selection sort and you come-up with the following code. However, there is a bug in the code. Identify that bug and explain why that is a bug and edit that part of the code to correct it. Later, analyze the run-time of the updated code:

```
void selectionSort(int arr[], int n)
{
    int i, j, min_idx, temp;

    // One by one move boundary of unsorted subarray
    for (i = 0; i < n-1; i++)
    {
        min_idx = i;
        for (j = 0; j < n; j++)
          if (arr[j] < arr[min_idx])
            min_idx = j;
```

j=0; //j isn't reset after the first iter so then it won't function properly on subsequent iters.

```
        temp = arr[i];
        arr[i] = arr[min_idx];
        arr[min_idx] = temp;

    }
}
```

for (i=0; i<n-1; i++)
↳ runs n-1 times

for(j=0; j<n; j++)
↑ runs n times

(n-1)(n) times worst case
n^2-n
$O(n^2)$

11) Explain the steps to come-up with the recurrence relation for merge sort and solve the recurrence relation to get the run-time of merge sort.

Mergesort calls itself twice, each time splitting in half, and also calls merge which has a constant amount of work per call depending on n.

mergesort(a, l, mid) mergesort(a, mid+1, r) merge(a, l, m, r)

n/2 n/2 cn

$T(n)=T(n/2)+T(n/2)+cn$
$=2T(\frac{n}{2})+cn$
$=2(2T(\frac{n}{4})+\frac{cn}{2})+cn$
$=4T(\frac{n}{4})+cn+cn$
$=4(2T(\frac{n}{8})+\frac{cn}{4})+2cn$
$=8T(\frac{n}{8})+3cn$
$=8(2T(\frac{n}{16})+\frac{cn}{8})+3cn$
$=16T(\frac{n}{16})+4cn$
...
$=2^kT(\frac{n}{2^k})+kcn$

base case: $T(1)=1$
$\frac{n}{2^k}=1 \quad k=\log_2 n$
$T(n)=2^{\log n}T(\frac{n}{2^{\log n}})+cn\log n$
$=nT(\frac{n}{n})+cn\log n$
$=n+cn\log n$

$T(\frac{n}{2})=2T(\frac{n}{4})+\frac{cn}{2}$
$T(\frac{n}{4})=2T(\frac{n}{8})+\frac{cn}{4}$
$T(\frac{n}{8})=2T(\frac{n}{16})+\frac{cn}{8}$

$O(n\log_2 n)$