```
Exp. 18

arr = [64, 25, 12, 22, 11]

1) Insert! Sort

i = 1, j = i = 1 = 0 \Rightarrow 64 > 25 \Rightarrow shift = 64 \text{ to right}

& place 25 at 0.

arr = 25, 64, 12, 22, 11

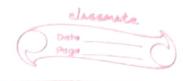
i = 2, j = 1 \Rightarrow 64 > 12 \Rightarrow shift = 64 \text{ right}

arr = 25, 12, 64, 22, 11

compare 25 & 12 \Rightarrow 25 > 12 \Rightarrow shift 25 right

place 12 at 0

\Rightarrow arr = 12, 25, 64, 22, 11
```



```
2) Select " sort
    i=0: find min. in 0-4 => 11@4
            -> swap 11 & arr[i] -> arr=11,25,12,5 22,64
     i=1: find min. in 1-4=> 12@2
           >> swap 12 & arr [i=1] => arr = 11,12,25, 22,64
     i=2: find min. in 2-4=>22@3
           \Rightarrow swap 22 & xr[2]=25 \Rightarrow axr=11, 12, 22, 25, 64
    i=3: find min. in 3-4 => 25@ 3
         => already in correct post; no swap needed.
 * Insert P Sort:
    builds sorted are in crementally
I shifts elements to make space for insert
-> requires many shifts to place each element.
-> potentially O(n) for nearly sorted arrays.
 * Selecti Sort:
-> Finds min. in each pass
-> Swaps min. with current unsorted post
-> fewer actual swaps as compared to shifts in insert
    sort.
```