Announcements –

PA1 available, due 01/28, 11:59p.

Today's Plan –

- Another sorting algorithm
- Runtime of recursive algorithms
- Correctness of recursive algorithms

Warm-up…

Task: Given an array where 1st and 2nd halves are sorted, return sorted array.

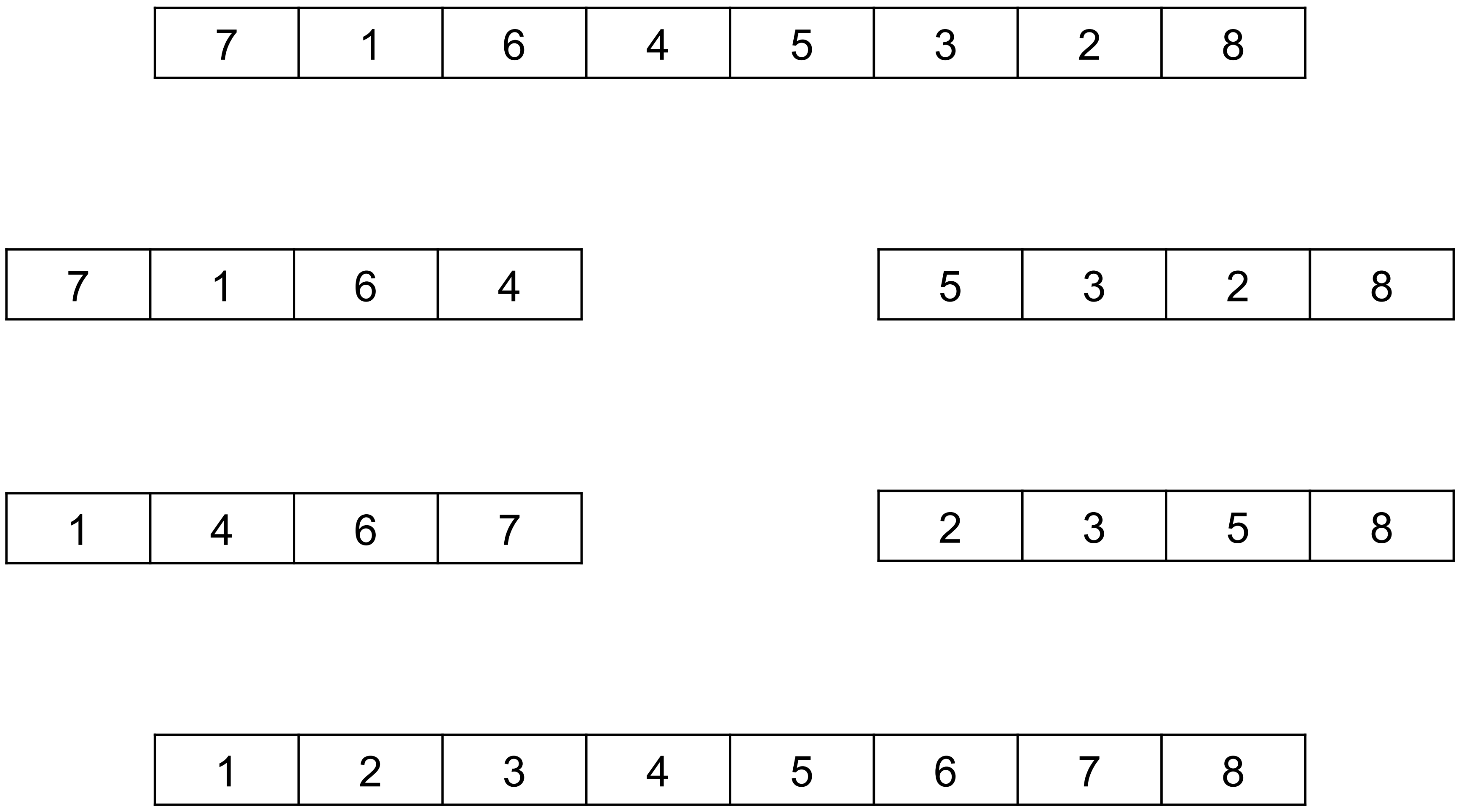
1	4	6	7	2	3	5	8


```
void merge(vector<int> & A,int fir,int sec, int secsize){
int firsave = fir; int firend = sec; int secend = sec + secsize;
vector<int> temp;
while (fir < firend && sec < secend){
    if (A[fir] < A[sec]) {
        temp.push_back(A[fir]); fir++;}
    else {
        temp.push_back(A[sec]); sec++;}}
if (fir == firend){
    while(sec != secend) {temp.push_back(A[sec]); sec++;}}
else {
    while(fir != firend) {temp.push_back(A[fir]); fir++;}}
for (int i = 0; i < temp.size(); i++)
    A[firsave+i] = temp[i];
}
```

Run time:

New…

Task: sort this array…



mergeSort

Task: sort this array

7	1	6	4	5	3	2	8

```
void mergeSort(vector<T> & A, int L, int R){








}
```

RT:

Recurrences – self referential functions

You know some already…

A recurrence for mergeSort's runtime:

```
void mergeSort(vector<T> & A, int L, int R){
        if (R > L) {
                int M = (R + L)/2;
                mergeSort(A, L, M);
                mergeSort(A, M+1, R);
                merge(A, L, M+1, R-M); }
}
```

Finding a closed form (two approaches, there are others):

1) Expand and generalize:

Finding a closed form (two approaches, there are others):

2) Recursion Tree:

Correctness of Recursive functions:

7	1	6	4	5	3	2	8

```
void mergeSort(vector<T> & A, int L, int R){
       if (R > L) {
              int M = (R + L)/2;
              mergeSort(A, L, M);
              mergeSort(A, M+1, R);
              merge(A, L, M+1, R-M); }
}
```