# Segment Trees

## CS 491 – Competitive Programming

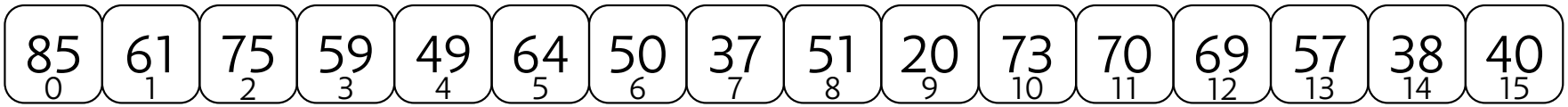
Dr. Mattox Beckman

University of Illinois at Urbana-Champaign
Department of Computer Science

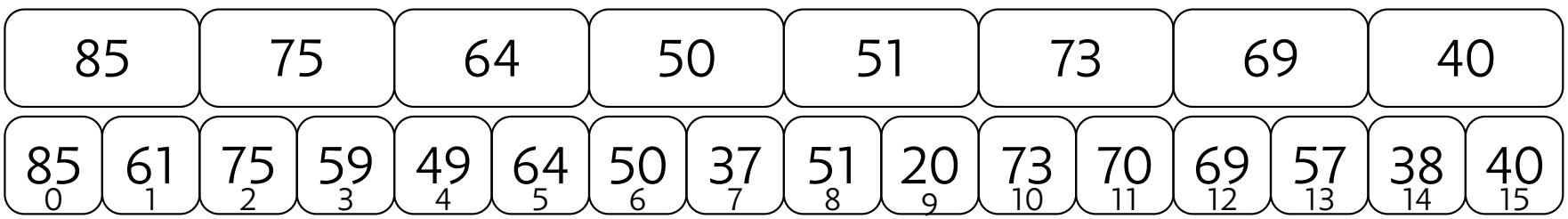
Fall 2023

# Running Example

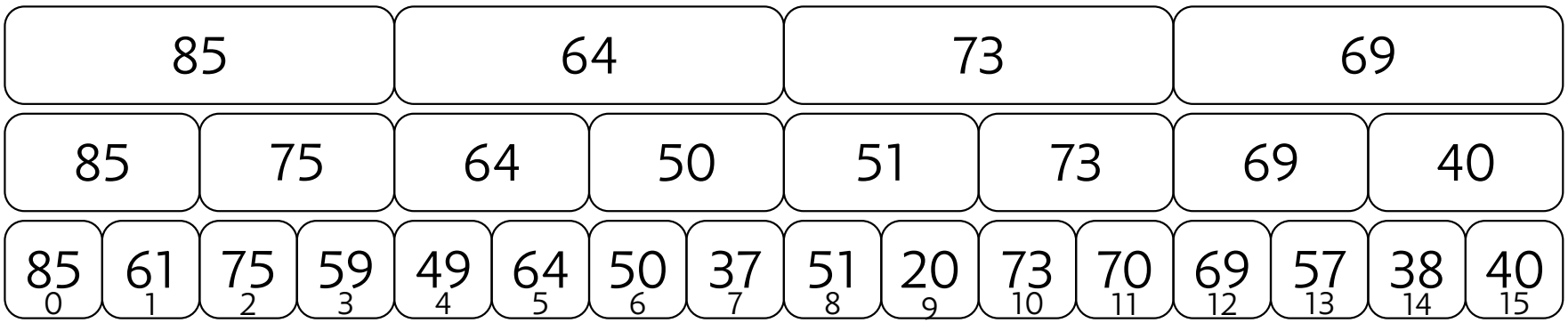
- Consider the following array:



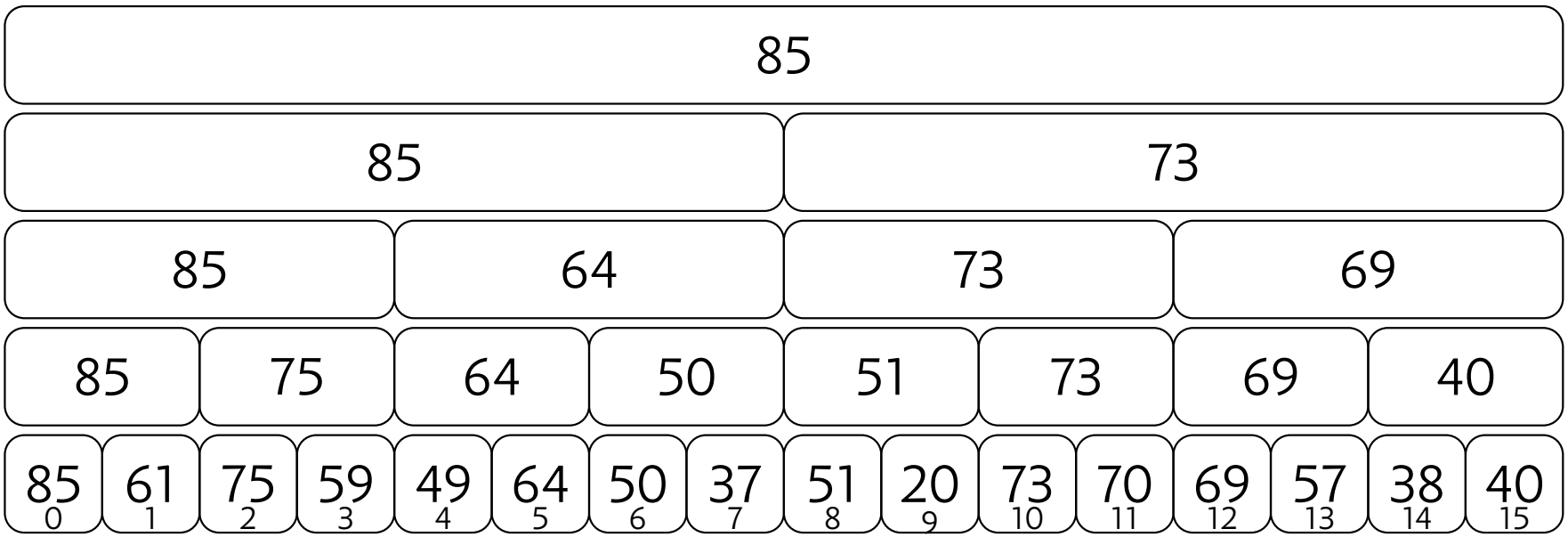
# Segment Trees, Level 1



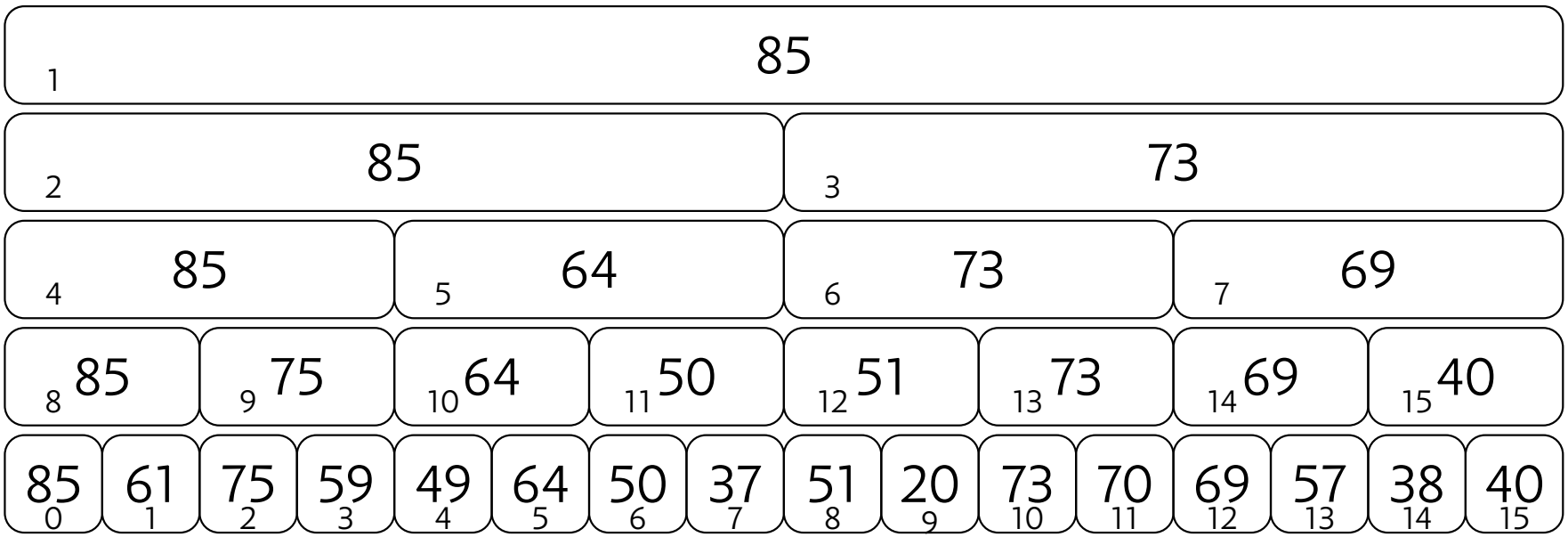
# Segment Trees, Level 2



# Segment Trees, Level 3 and 4



# Segment Trees, Numering the Elements



# Build the Segement Tree

- L and R give you the bounds with respect to the orignal array.
- This code gives you a max range.

```
1  void build(int p, int L, int R) {
2    if (L == R) // as L == R, either one is fine
3       st[p] = data[L]; // store the data
4    else {
5      // recursively compute the values
6      build(left(p) , L                 , (L + R) / 2);
7      build(right(p), (L + R) / 2 + 1, R             );
8      int p1 = st[left(p)], p2 = st[right(p)];
9      st[p] = (st[p1] >= st[p2]) ? p1 : p2;
10   } }
```

# Query the Tree

- L and R give you the bounds with respect to the orignal array.
- i and j give you the bounds for the query

```
1  int rmq(int p, int L, int R, int i, int j) {
2    if (i > R || j < L) return -1; // current segment outsid
3    if (L >= i && R <= j) return st[p];
4    // compute the min position in the left and right part o
5    int p1 = rmq(left(p) , L         , (L+R)/2, i, j);
6    int p2 = rmq(right(p), (L+R)/2+1, R      , i, j);
7    if (p1 == -1) return p2;
8    // if we try to access segment outside query
9    if (p2 == -1) return p1;
10   return (p1 >= p2) ? p1 : p2;
11 }
12
```