# Assignment 6

> **Name: Mohammed Varaliya**

> **Roll No: 54**

**Question: What do you understand by Reheap up? Write the recursive algorithm for the same.**

Reheap up, also known as "heapify up" or "sift up," is an operation used in a binary heap data structure to maintain the heap property after a new element has been added to the heap. The heap property requires that each parent node is greater than (in a max-heap) or less than (in a min-heap) its child nodes. When a new element is added, it may violate this property, and the reheap up operation ensures that the heap property is restored by comparing the newly added element with its parent and swapping them if necessary.

## Recursive Algorithm for Reheap Up

Here's a recursive algorithm for the reheap up operation in a max-heap:

1. **Input**: An array representing the heap, the index of the newly added element.

2. **Base Case**: If the index is 0 (the element is the root), return.

3. **Calculate Parent Index**: The parent index can be found using the formula: `parent_index = (index - 1) / 2`.

4. **Compare and Swap**:

   a. If the value at `index` is greater than the value at `parent_index`, swap them.

   b. Recursively call the reheap up function for the parent index.

## Pseudocode

```
function reheapUp(heap, index):
    if index == 0:
        return  // Base case: if at the root, nothing to do

    parent_index = (index - 1) // 2  // Calculate parent index

    if heap[index] > heap[parent_index]:  // For max-heap
        swap(heap[index], heap[parent_index])  // Swap the elements
        reheapUp(heap, parent_index)  // Recursively call for parent index
```

## Example

Let's say i have a max-heap represented as an array: `[20, 15, 10, 8, 5]`, and i want to add the element `17`.

1. After adding `17`, the heap looks like: `[20, 15, 10, 8, 5, 17]`.

2. The index of `17` is 5, and its parent index is 2 (`(5 - 1) // 2`).

3. Since `17` is greater than `10` (the parent), they are swapped.

4. The heap now looks like: `[20, 15, 17, 8, 5, 10]`.

5. The new index of `17` is 2. Its parent index is 0.

6. Since `17` is not greater than `20`, the process stops.

This ensures that the max-heap property is preserved.