ADS LAB write up Binomial heap

Suraj Pardeshi 1BM18CS114

```
class Binomial_Tree :
    def __init__(self,key):
        self.key = key
        self.children = []
        self.order = 0
class Binomial_heap :
    def __init__(self):
        self.trees = []

def get_min(self):
    if not self.trees:
        return None

    least = self.trees[0].key
    for tree in self.trees:
        if tree.key < least:
            least = tree.key
    return least

def merge(self, h):
    self.combine_roots(h)
    if not self.trees:
        return
    i = 0
    while i < len(self.trees)-1:
        current = self.trees[i]
        after = self.trees[i+1]
        if current.order == after.order:
            if (i+1 < len(self.trees)) >1
            and self.trees[i+2].order == after.order):
                after_after = self.trees[i+2]
```

```
            if other.key < other_other.key:
                other.add_at_end(other_other)
                del self.trees[i+2]
            else:
                other_other.add_at_end(other)
                del self.trees[i+1]
        else:
            if current.key < other.key:
                current.add_at_end(other)
                del self.trees[i+1]
            else:
                other.add_at_end(current)
                del self.trees[i]
        i = i+1

def insert(self, key):
    g = BinomialHeap()
    g.trees.append(BinomialTree(key))
    self.merge(g)
```