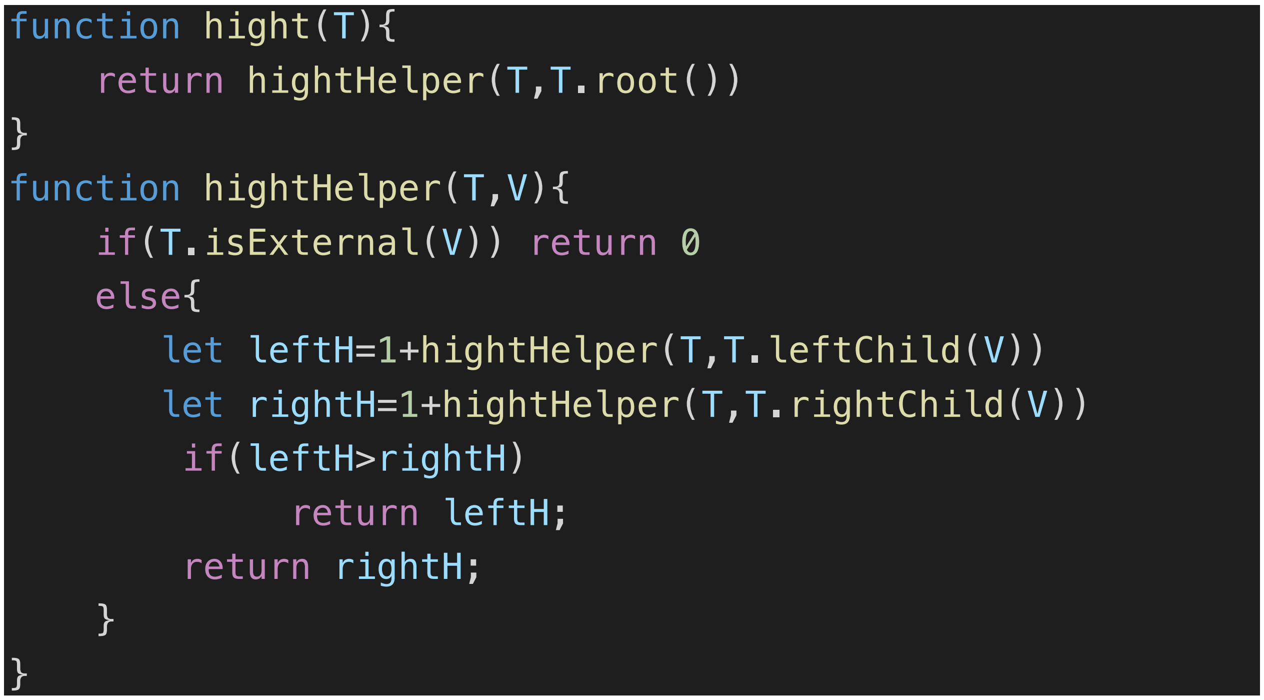
```
Algorthim Height (T)
        Return heightHelper(T,T.root()).                O(1)


Algorthim heightHelper(T,V)
             If(T.isExternal(V)) return 0;                 O(1)
Else
      leftH:=1+heightHelper(T,T.leftChiled()).            O(n)
     rightH:= 1+heightHelper(T,T.rightChiled())           O(n)
     if(leftH>rightH)  return leftH;                    O(1)
    else return rightH;                                 O(1)
over all time complexity is O(n)
```



```
function hight(T){
    return hightHelper(T,T.root())
}
function hightHelper(T,V){
    if(T.isExternal(V)) return 0
    else{
       let leftH=1+hightHelper(T,T.leftChild(V))
       let rightH=1+hightHelper(T,T.rightChild(V))
        if(leftH>rightH)
             return leftH;
        return rightH;
    }
}
```

```
Algorthim eulerTour(T,p)
   leftH:=0.
   rightH:=0
   If T.isExternal return 0;
   Else
     vistPreorder(T,p)
      leftH=1+eulerTour(T,T.leftChiled(p))
     visitInOrder(T,p)
      rightH=1+ eulerTour(T,T.rightChiled(p))
     if(leftH>rightH) return leftH
     else return righth
 return Math.max(leftH,rightH)

Algorthim height(T)
  Return eulerTour(T,T.root())
```

```
height(T) {
        return this.eulerTour(T, T.root())
    }
    eulerTour(T, p) {
       let leftH=0;
       let rightH=0;
        if (T.isExternal(p)) {
            return 0
        } else {
             this.visitPreOrder(T, p);
            leftH = 1+this.eulerTour(T, T.leftChild(p));
            this.visitInOrder(T, p);
           rightH = 1+this.eulerTour(T, T.rightChild(p));
            if(leftH>rightH) return leftH
            return rightH
        }
        return Math.max(leftH,leftH)
    }
```