تمرین سری ۵

محمدمهدی محمدی‌پیام - ۹۴۱۰۰۰۶۸

مساله ۳

با زمان $O(n)$ می‌توان یافت. برای این‌کار باید از یک رهیافت بازگشی استفاده کنیم. برای سادگی فرض میکنیم کلید هر راس متمایز است. به ازای هر راس در نظر برسی میکنیم حداقل کلید زیر درخت سمت چپ از کلید راس بیشتر و حداکثر کلید زیردرخت سمت راست از کلید راس کمتر باشد.

برای این کار از تابع CHECK-NODE استفاده میکنیم که مقدار بازگشتی آن جفت کمترین، بیشترین زیردرخت با ریشه راس ما است. درصورتی که تابع CHECK-NODE مقدار NILL بازگرداند یعنی زیردرخت آن BST نیست.

```
CHECK-NODE(Node n):

    min = n.key
    max = n.key

    if n.right != NILL:
        check-right = CHECK-NODE(n.right)
        min = min{n.key, check-right.min}
    if n.left != NILL:
        check-left = CHECK-NODE(n.left)
        max = max{n.key, check-left.max}
    if check-right == NILL or check-left == NILL:
        return NILL

    if check-right.max > n.key or check-left.min < n.key:
        return NILL

    return {min, max}
```

واضح است که به ازای هر گره تابع CHECK-NODE یکبار صدا زده می‌شود.

مساله ۸

از تابع زیر استفاده می‌کنیم.

```
MEDIAN(A):
    low = min A
    high = max A
    while low < high:
        size = number of elements in A between [low, high]
        r = random in [1, size]
        p = (r+low)th item in A
        lowIndex = number of elements in A at most p

        if lowIndex >= n/2:
            high = p
        else
            low = p
    return low
```

تابع در هر مرحله از حلقه طول low تا high را به ¾ کاهش می‌دهد بنابراین زمان اجرای کد $O(n \log n)$ است.