Problem Set 4

XUANQI WEI

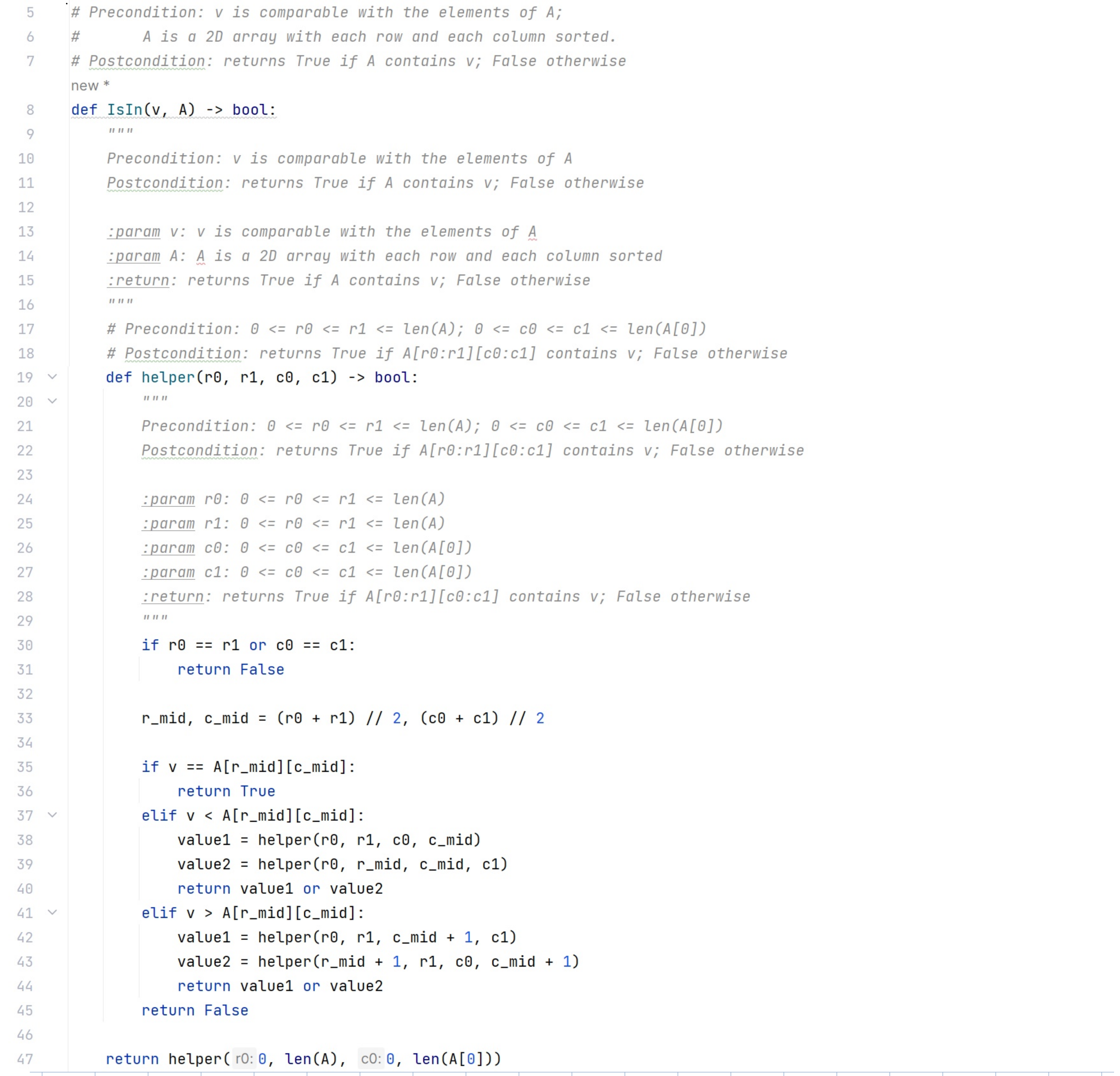
1009353209

Q3

(a). $0 \leq r_0 \leq r_1 \leq len(A)$

$0 \leq c_0 \leq c_1 \leq len(A[0])$.

(b)



```
# Precondition: v is comparable with the elements of A;
#       A is a 2D array with each row and each column sorted.
# Postcondition: returns True if A contains v; False otherwise
new *
def IsIn(v, A) -> bool:
    """
    Precondition: v is comparable with the elements of A
    Postcondition: returns True if A contains v; False otherwise

    :param v: v is comparable with the elements of A
    :param A: A is a 2D array with each row and each column sorted
    :return: returns True if A contains v; False otherwise
    """
    # Precondition: 0 <= r0 <= r1 <= len(A); 0 <= c0 <= c1 <= len(A[0])
    # Postcondition: returns True if A[r0:r1][c0:c1] contains v; False otherwise
    def helper(r0, r1, c0, c1) -> bool:
        """
        Precondition: 0 <= r0 <= r1 <= len(A); 0 <= c0 <= c1 <= len(A[0])
        Postcondition: returns True if A[r0:r1][c0:c1] contains v; False otherwise

        :param r0: 0 <= r0 <= r1 <= len(A)
        :param r1: 0 <= r0 <= r1 <= len(A)
        :param c0: 0 <= c0 <= c1 <= len(A[0])
        :param c1: 0 <= c0 <= c1 <= len(A[0])
        :return: returns True if A[r0:r1][c0:c1] contains v; False otherwise
        """
        if r0 == r1 or c0 == c1:
            return False

        r_mid, c_mid = (r0 + r1) // 2, (c0 + c1) // 2

        if v == A[r_mid][c_mid]:
            return True
        elif v < A[r_mid][c_mid]:
            value1 = helper(r0, r1, c0, c_mid)
            value2 = helper(r0, r_mid, c_mid, c1)
            return value1 or value2
        elif v > A[r_mid][c_mid]:
            value1 = helper(r0, r1, c_mid + 1, c1)
            value2 = helper(r_mid + 1, r1, c0, c_mid + 1)
            return value1 or value2
        return False

    return helper(r0: 0, len(A), c0: 0, len(A[0]))
```

(c). Let n be the number of elements in matrix A, where A is a 2D array with each row and each element sorted, v is comparable with elements in A.

Let r_0, r_1 be the beginning row and ending row index (exclude).

Let c_0, c_1 be the beginning column and ending column index (exclude).

Let r_0, r_1, c_0, c_1 be input of the helper function.

Let $r\_mid = (r_0 + r_1) // 2$; $c\_mid = (c_0 + c_1) // 2$, where $A[c\_mid][r\_mid]$ is the (not exact) middle of A.

The Precondition: $0 \le r_0 \le r_1 \le len(A)$.; $0 \le c_0 \le c_1 \le len(A[0])$.

The Postcondition: returns True if $A[r_0:r_1][c_0:c_1]$ contains 1. else, returns False.

Let $P(n)$: If precondition holds, postcondition will be satisfied by the return value.

Base Case: $n=0$.

When $n=0$, there is no element in A, which A is an empty list.

Thus. $r_0 = r_1 = 0$, $c_0 = c_1 = 0$. v can't appear. the function terminates as written and it'll return False.

Inductive Step. Let $n \in \mathbb{N}$. $n > 0$.

Inductive Hypothesis: Assume $\forall k \in \mathbb{N}$, $0 \le k < n$. $P(k)$ holds.

WTS: $P(n)$ holds.

Case 1: v is in A.

① $v = A[r\_mid][c\_mid]$.

Since $v = A[r\_mid][c\_mid]$. from line 36, it terminates and return True.

② $v > A[r\_mid][c\_mid]$.

Since A is sorted and $v > A[r\_mid][c\_mid]$, gives. v is bigger than $A[r_0:r\_mid][c\_mid:c_1]$. and v is bigger than $A[r_0:r_1][c_0:c\_mid]$ as well.

Since we obtain that $0 \le r0 \le r\_mid+1 \le r1 \le len(A)$ and $0 \le c0 \le c\_mid+1 \le c1 \le len(A[0])$. by I.H., it gives the function terminates and return True.

③ v< A[r_mid][c_mid].

Since v< A[r_mid][c_mid] and A is sorted, gives v is smaller than A[r_mid+1][c0, c_mid+1] and v is smaller than A[r0:r1][c_mid+1:c1]

Since $0 \le r0 \le r\_mid+1 \le r1 \le len(A)$ and $0 \le c0 \le c\_mid+1 \le c1 \le len(A[0])$. by I.H., it gives the function terminates and returns True as written in the code in (b).

Case 2: v is not in A.

From the code in (b), we obtain that after checking v is not in the middle, v is not bigger that A[r_mid][c_mid], and v is not smaller than A[r_mid][c_mid], the function terminates and return False. as stated in line 45.

Therefore, I've proved the function terminates and the postcondition is satisfied Since Helper is correct. and IsIn fully relay on Helper, when the pre. of IsIn. is correct, the pre. of Helper is correct as well, which r0, r1, c0, c1 are valid inputs.

Therefore, IsIn is also correct.

■