6-(c)

```
import random
import sys
def Indexing(input_list,m):
    l = []
    for n in range(m):
        l.append([])
    for i in range(len(input_list)):
        l[input_list[i]-1].append(i)
    return l

def binary_search(t, d):
    start = 0
    end = len(d) - 1
    while start <= end:
        m = (start + end) // 2
        if d[m] == t:
            return True
        elif d[m] < t:
            start = m + 1
        else:
            end = m - 1
    return None

def right_majority(check_num,input_dic, start, end):
    a = input_dic[check_num-1]
    times = 0
    for i in range(end-start+1):
        if (binary_search(i+start-1,a)):
            times = times + 1
    if (times > (end-start+1)/2):
        return True
    else :
        return False

def random_majority(input_list, input_dic, start, end):
    li2 = input_list[start-1:end-1].copy()
    for k in range(20):
        r=random.randrange(0,len(li2))
        if(right_majority(li2[r],input_dic,start,end)):
            return li2[r]
    return 0
def input_form():
    input_1 = sys.stdin.readline().strip().split()
    number_of_number = int(input_1[0])
    range_of_number = int(input_1[0])
    input_list = list(map(int,sys.stdin.readline().strip().split()))
```

```
    number_of_chunks = int(sys.stdin.readline().strip())
    input_list_2 = [sys.stdin.readline().strip() for i in range(number_of_chunks)]
    return    number_of_number,    range_of_number,    input_list,    number_of_chunks,
input_list_2

def output_form(n_of_chunks, chunks,input_list, input_dic):
    for i in range(n_of_chunks):
        r = chunks[i].split()
        start = int(r[0])
        end = int(r[1])
        result = random_majority(input_list,input_dic, start, end)
        if (result != 0):
            print("yes", result)
        else :
            print("no")

if __name__ == "__main__":
    n_of_number, r_of_number, inp_list, n_of_chunks, chunk_list = input_form()
    indexed_list = Indexing(inp_list,r_of_number)
    output_form(n_of_chunks, chunk_list, inp_list, indexed_list)
    pass
```