**Cloudy day**

import math

import os

import random

import re

import sys

import operator

def maximumPeople(people, tower\_pos, lclouds, lranges):

    clouds = []

    for c, r in zip(lclouds, lranges):

        clouds.append((max(c - r, 0), c + r))

    # sort by start

    clouds.sort(key=lambda x: x[0])

    towers = []

    for pos, p in zip(tower\_pos, people):

        towers.append((pos, p))

    towers.sort(key=lambda x: x[0])

    last\_tower\_pos = towers[-1][0]

    last\_cloud = clouds[-1][1]

    ghost\_pos = max(last\_tower\_pos, last\_cloud) + 100

    clouds.append((ghost\_pos, ghost\_pos))

    cend = -10 \* 9

    covered = 0

    uncovered = 0

    max\_covered = 0

    t\_idx = 0

    def count(pos, exc=False):

        res = 0

        nonlocal t\_idx

        # uses less than or less or equal operator

        op = operator.lt if exc else operator.le

        while (t\_idx < len(towers) and op(towers[t\_idx][0], pos)):

            res += towers[t\_idx][1]

            t\_idx += 1

        return res

    for start, end in clouds:

        if start > cend:

            covered += count(cend)

            max\_covered = max(max\_covered, covered)

            covered = 0

            uncovered += count(start, exc=True)

            cend = end

        # next cloud starts and ends before the next cloud

        elif start <= cend and end < cend:

            covered += count(start, exc=True)

            \_ = count(end)

        # or it start before but ends later

        elif start <= cend and end >= cend:

            covered += count(start, exc=True)

            max\_covered = max(max\_covered, covered)

            covered = 0

            \_ = count(cend)

            cend = end

    return max\_covered + uncovered

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    n = int(input().strip())

    p = list(map(int, input().rstrip().split()))

    x = list(map(int, input().rstrip().split()))

    m = int(input().strip())

    y = list(map(int, input().rstrip().split()))

    r = list(map(int, input().rstrip().split()))

    result = maximumPeople(p, x, y, r)

    fptr.write(str(result) + '\n')

    fptr.close()

**output:**

