

题目描述：

按照环保公司要求，小明需要在沙化严重的地区进行植树防沙工作，初步目标是种植一条直线的树带。由于有些区域目前不适合种植树木，所以只能在一些可以种植的点来种植树木。

在树苗有限的情况下，要达到最佳效果，就要尽量散开种植，不同树苗之间的最小间距要尽量大。给你一个适合种植树木的点坐标和一个树苗的数量，请帮小明选择一个最佳的最小种植间距。

例如，适合种植树木的位置分别为 `1,3,5,6,7,10,13` 树苗数量是 `3`，种植位置在 `1,7,13`，树苗之间的间距都是 `6`，均匀分开，就达到了散开种植的目的，最佳的最小种植间距是 `6`

输入描述：

第 `1` 行表示适合种树的坐标数量

第 `2` 行是适合种树的坐标位置

第 `3` 行是树苗的数量

例如，

`7`

`1 5 3 6 10 7 13`

`3`

输出描述：

最佳的最小种植间距

补充说明：

位置范围为 `1~100000000`，种植树苗的数量范围 `2~100000000`，用例确保种植的树苗数量不会超过有效种植坐标数量。

示例 1

输入：

`7`

`1 5 3 6 10 7 13`

3

输出:

6

说明:

```
import sys
```

```
# 读取
```

```
n = int(sys.stdin.readline().strip())
```

```
positions = list(map(int, sys.stdin.readline().strip().split()))
```

```
m = int(sys.stdin.readline().strip())
```

```
def check(minDis,positions,m):
```

```
    count = 1
```

```
    curPos = positions[0]
```

```
    for i in range(n):
```

```
        if positions[i] - curPos >= minDis:
```

```
            count +=1
```

```
            curPos = positions[i]
```

```
    return count >= m
```

```
positions.sort()
```

```
low_pos = 1
```

```
high_pos = positions[-1] - positions[0]
```

```
res = 0
```

```
while low_pos <= high_pos:
```

```
    mid_pos = (low_pos+high_pos) >> 1
```

```
    if check(mid_pos, positions, m):
```

```
        res = mid_pos
```

```
        low_pos = mid_pos + 1
```

```
    else:
```

```
        high_pos = mid_pos - 1
```

```
sys.stdout.write(str(res)+"\n")
```