



```
Microsoft Visual Studio 偵錯主控台
number of test cases:2
i/s n q:i 10 3
enter the interger array:0 10 20 30 40 50 60 70 80 90
enter the interger target:40 100 60
target 40 in index 4
target 100 in index -1
target 60 in index 6
i/s n q:s 3 3
enter the string member:kirito starburst stream
enter the string target:kirito starburst C8763
target kirito in index 0
target starburst in index 1
target C8763 in index -1

C:\Users\user\Desktop\EX\Project1\Debug\Project1.exe (處理序 11644) 已結束，出現代碼 0。
若要在偵錯停止時自動關閉主控台，請啟用 [工具] -> [選項] -> [偵錯] -> [偵錯停止時，自動關閉主控台]。
按任意鍵關閉此視窗...
```

- 初始化與輸入：

- 開始時，首先讀取測試案例的數量。對於每個測試案例，讀取其資料類型（整數 i 或字串 s ）、數組大小 n 和查詢次數 q 。
- 然後，根據資料類型（ i 或 s ）選擇相應的數據類型，並讀入數組及查詢目標。

- 排序：

- 對於每個測試案例，無論是整數還是字串，首先都需要將數組進行排序，這是因為 **二分查找** 要求數組是有序的。
- 這裡使用 C++ 標準庫中的 `std::sort()` 函數對數組進行升序排序。

- 二分查找：

- 排序後，使用 `binary_search` 函數對每個查詢目標值進行查找。這會返回目標值在數組中的索引，若找不到則返回 `-1`。
- 查找的結果會以「目標值在索引 x 」的格式輸出。

- 結果輸出：

- 每次查找完成後，將結果輸出，告訴用戶目標值在數組中的位置。

```

#include<iostream>
using namespace std;

template <typename T>

int binary_search(T arr[], int size, T target) {
    int left = 0, right = size - 1;
    int mid;

    while (left<=right){
        mid = (left + right) / 2;
        if (arr[mid] == target) {
            return mid;
        }
        else if (arr[mid] > target) {
            right = mid - 1;
        }
        else if (arr[mid] < target){
            left = mid + 1;
        }
    }
    return -1;
}

void main() {
    int n, q;
    char t;
    int test;
    cout << "number of test cases:";
    cin >> test;
    while (test--) {
        cout << "i/s n q:";
        cin >> t >> n >> q;
        if (t == 'i') {
            int* arr = new int[n];
            int* target = new int[q];
            cout << "enter the interger array:";
            for (int i = 0; i < n; i++) {

```

```

        cin >> arr[i];
    }
    cout << "enter the interger target:";
    for (int i = 0; i < q; i++) {
        cin >> target[i];
    }
    for (int i = 0; i < q; i++) {
        cout << "target " << target[i] << " in index " <<
binary_search<int>(arr, n, target[i]) << endl;
    }
    delete[] arr;
    delete[] target;
}
else if (t == 's') {
    string* str = new string[n];
    string* target = new string[q];
    cout << "enter the string member:";
    for (int i = 0; i < n; i++) {
        cin >> str[i];
    }
    cout << "enter the string target:";
    for (int i = 0; i < q; i++) {
        cin >> target[i];
    }
    for (int i = 0; i < q; i++) {
        cout << "target " << target[i] << " in index " <<
binary_search<string>(str, n, target[i]) << endl;
    }
    delete[] str;
    delete[] target;
}
}
}

```