HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

C BASIC

TÌM KIẾM NHỊ PHÂN

ONE LOVE. ONE FUTURE.

NỘI DUNG

- Tìm kiếm nhị phân
- Bài tập kiểm tra tồn tại (P.06.13.01)
- Bài tập cặp số có tổng cho trước (P.06.13.02)

TÌM KIẾM NHỊ PHÂN

- Cho dãy $a_1, a_2, \ldots, a_n$ được sắp xếp theo thứ tự không giảm. Cho trước một giá trị k, hãy tìm chỉ số i sao cho $a_i = k$.
- Tìm kiếm nhị phân
  - Xét phần tử ở giữa của dãy a_m với $m = (1+n)/2$
  - Nếu $a_m = k$ thì trả về chỉ số m
  - Nếu $a_m < k$ thì lặp lại tìm kiếm nhị phân trên dãy $a_{m+1}, a_{m+2}, \ldots, a_n$
  - Ngược lại, nếu $a_m > k$ thì lặp lại tìm kiếm nhị phân trên dãy $a_1, a_2, \ldots, a_{m-1}$.

```
bSearch(a[1..n], L, R, k){
  if L > R then return -1; // not found
  if L = R then {
     if a[L] = k then return L; else return -1;
  }
  m = (L+R)/2;
  if a[m] = k then return m;
  if a[m] < k then return bSearch(a[1..n], m+1, R, k);
  else return bSearch(a[1..n], L, m-1, k);
}
```

BÀI TẬP KIỂM TRA TỒN TẠI (P.06.13.01)

- Cho dãy $a_1, a_2, \ldots, a_n$.Thi hành các truy vấn dạng:
  - check k: trả về 1 nếu k xuất hiện trong dãy đã cho và trả về 0, nếu ngược lại
- Dữ liệu
  - Dòng 1: ghi số nguyên dương n (1 <= n <= 100000)
  - Dòng 2 ghi $a_1, a_2, \ldots, a_n$ trong đó (1 <= a_i <= 1000000)
  - Các dòng tiếp theo, mỗi dòng ghi 1 truy vấn có định dạng mô tả ở trên
  - Dữ liệu đầu vào được kết thúc bởi 1 dòng chứa #
- Kết quả
  - Ghi ra trên mỗi dòng, kết quả của 1 truy vấn tương ứng ở đầu vào

stdin	stdout
5 1 3 3 3 4 check 3 check 3 check 10 check 5 check 8 #	1 1 0 0 0

BÀI TẬP KIỂM TRA TỒN TẠI – THIẾT KẾ THUẬT TOÁN - MÃ GIẢ

- Thuật toán
  - Sắp xếp dãy $a_1, a_2, \ldots, a_n$ theo thứ tự không giảm
  - Với mỗi truy vấn check k: thực hiện tìm kiếm giá trị k trên dãy đã cho bằng tìm kiếm nhị phân

```
Run() {
  read a[1..n] from the stdin;
  sort(a[1..n]) in a non-decreasing order;
  while true do {
     cmd = read a string from stdin;
     if cmd = "#" then break;
     if cmd = "check" then {
        k = read an integer from stdin;
        i = bSearch(a[1..n], 1, n, k);
        if i > -1 then write(1); else write(0);
     }
  }
}
```

BÀI TẬP KIỂM TRA TỒN TẠI - CODE

```
#include <stdio.h>
#define N 100001
int a[N];
int n;
void input(){
  scanf("%d",&n);
  for(int i = 1; i <= n; i++){
    scanf("%d",&a[i]);
  }
}
void swap(int i, int j){
  int t = a[i]; a[i] = a[j]; a[j] = t;
}
```

```
void heapify(int i, int n){
  int L = 2*i; int R = 2*i+1; int maxIdx = i;
  if(L <= n && a[maxIdx] < a[L]) maxIdx = L;
  if(R <= n && a[maxIdx] < a[R]) maxIdx = R;
  if(maxIdx != i){ swap(i,maxIdx); heapify(maxIdx,n); }
}
void buildHeap(){
  for(int i = n/2; i >= 1; i--) heapify(i,n);
}
void heapSort(){
  buildHeap();
  for(int i = n; i >= 2; i--){
    swap(1,i); heapify(1,i-1);
  }
}
```

BÀI TẬP KIỂM TRA TỒN TẠI - CODE

```
int bSearch(int i, int j, int k){
  if(i > j) return -1;
  if(i == j){
    if(a[i]==k) return i; else return -1;
  }
  int m = (i+j)/2;
  if(a[m] == k) return m;
  if(a[m] < k) return bSearch(m+1,j,k);
  else return bSearch(i,m-1,k);
}
```

```
int main(){
  input();
  heapSort();
  char cmd[30];
  while(1){
    scanf("%s",cmd);
    if(strcmp(cmd,"#")==0) break;
    else if(strcmp(cmd,"check")==0){
      int k; scanf("%d",&k);
      int i = bSearch(1,n,k);
      if(i > 0) i = 1; else i = 0;
      printf("%d\n",i);
    }
  }
  return 0;
}
```

BÀI TẬP CẶP SỐ CÓ TỔNG CHO TRƯỚC (P.06.13.02)

- Cho dãy $a_1, a_2, \ldots, a_n$ và giá trị Q. Hãy đếm số M các cặp 2 chỉ số (i, j) sao cho $1 <= i < j <= n$ vào $a_i + a_j = Q$.
- Dữ liệu
  - Dòng 1: ghi số nguyên dương n và Q ($1 <= n, Q <= 1000000$)
  - Dòng 2 ghi $a_1, a_2, \ldots, a_n$ trong đó ($1 <= a_i <= 1000000$)
- Kết quả
  - Ghi ra giá trị M

stdin	stdout
5 8 4 6 5 3 2	2

BÀI TẬP CẶP SỐ CÓ TỔNG CHO TRƯỚC – THUẬT TOÁN – MÃ GIẢ

- Thuật toán
  - Sắp xếp dãy $a_1, a_2, \ldots, a_n$ theo thứ tự không giảm
  - Duyệt dãy từ trái qua phải, với mỗi chỉ số i, ta thực hiện tìm kiếm nhị phân giá trị $Q - a_i$ trên dãy $a_{i+1}, a_{i+2}, \ldots, a_n$.

```
Run() {
  read a[1..n] and Q from the stdin;
  sort(a[1..n]) in a non-decreasing order;
  cnt = 0;
  for i = 1 to n-1 do {
    idx = bSearch(a[1..n], i+1, n, Q – a[i]);
    if idx > 0 then
      cnt = cnt + 1;
  }
  write(cnt);
}
```

BÀI TẬP CẶP SỐ CÓ TỔNG CHO TRƯỚC – CODE

```
#include <stdio.h>
#define N 1000001
int a[N];
int n, Q;
void input(){
  scanf("%d%d",&n,&Q);
  for(int i = 1; i <= n; i++){
    scanf("%d",&a[i]);
  }
}
void swap(int i, int j){
  int t = a[i]; a[i] = a[j]; a[j] = t;
}
```

```
void heapify(int i, int n){
  int L = 2*i; int R = 2*i+1; int maxIdx = i;
  if(L <= n && a[maxIdx] < a[L]) maxIdx = L;
  if(R <= n && a[maxIdx] < a[R]) maxIdx = R;
  if(maxIdx != i){ swap(i,maxIdx); heapify(maxIdx,n); }
}
void buildHeap(){
  for(int i = n/2; i >= 1; i--) heapify(i,n);
}
void heapSort(){
  buildHeap();
  for(int i = n; i >= 2; i--){
    swap(1,i); heapify(1,i-1);
  }
}
```

BÀI TẬP CẶP SỐ CÓ TỔNG CHO TRƯỚC – CODE

```
int bSearch(int i, int j, int k){
  if(i > j) return -1;
  if(i == j){
    if(a[i]==k) return i; else return -1;
  }
  int m = (i+j)/2;
  if(a[m] == k) return m;
  if(a[m] < k) return bSearch(m+1,j,k);
  else return bSearch(i,m-1,k);
}
```

```
int main(){
  input();
  heapSort();
  int cnt = 0;
  for(int i = 1; i <= n-1; i++){
    int idx = bSearch(i+1,n,Q-a[i]);
    if(idx > -1) cnt = cnt + 1;
  }
  printf("%d",cnt);
  return 0;
}
```

HUST

hust.edu.vn fb.com/dhbkhn

THANK YOU !