HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

# C BASIC


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

# C BASIC

## SORTING AND APPLICATIONS (PART 2)

ONE LOVE. ONE FUTURE.

# CONTENT

- Sorting vectors of integers (P.05.12.01)
- Sorting a list of records (P.05.12.02)
- Finding the greatest common subset of two sets (P.05.12.03)

# SORTING VECTORS OF INTEGERS (P.05.12.01)

- Given a sequence of vectors (each vector consists of m integers) $a_1$, $a_2$, . . ., $a_n$. Sort the given sequence of vectors in non-decreasing order
- Data
  - Line 1: Two positive integers *n* and *m* (1 <= *n* <= 100000, 1 <= *m* <= 10)
  - Line *i*+1 (*i* = 1, 2, …, n): m elements of vector $a_i$ (values of each elements are from 1 to 100)
- Result
  - Line *i* (*i* = 1, 2, …, n): *m* element of vector $a_i$ in the sorted sequence (SPACE separator)

| stdin | stdout |
|---|---|
| 6 3<br>10 9 7<br>5 10 2<br>10 9 1<br>4 7 3<br>7 5 10<br>7 9 3 | 4 7 3<br>5 10 2<br>7 5 10<br>7 9 3<br>10 9 1<br>10 9 7 |

# SORTING VECTORS OF INTEGERS - PSEUDOCODE

- Using heap sort
- Using pointers and dynamic memory allocation:
  - a[i] is the pointer to the $i^{th}$ vector

```
cmp(a, i, j){// compare a[i] và a[j]
   for k = 1 to m do {
     if a[i][k] < a[j][k] then return -1;
     else if a[i][k] > a[j][k] then return 1;
   }
   return 0; //  equal
}
```

```
Heapify(a, i, n){
  L = 2*i; R = 2*i+1; maxIdx = i;
  if L <= n and cmp(a,L,maxIdx) = 1 then maxIdx = L;
  if R <= n and cmp(a,R,maxIdx) = 1 then maxIdx = R;
  if maxIdx != i then {
     swap(a[i], a[maxIdx]); Heapify(maxIdx, n);
  }
}
BuildHeap(){
  for i = n/2 downto 1 do Heapify(i, n);
}
HeapSort(){
  BuildHeap();
  for i = n downto 2 do {
     swap(a[1], a[i]); Heapify(1, i-1);
  }
}
```

# SORTING VECTORS OF INTEGERS - PSEUDOCODE

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

```
int cmp(int i, int j){
  for(int k = 1; k <= m; k++){
    if(a[i][k] < a[j][k]) return -1;
    else if(a[i][k] > a[j][k]) return 1;
  }
  return 0;
}
void swap(int i, int j){
  int* tmp = a[i]; a[i] = a[j]; a[j] = tmp;
}
```

# SORTING A LIST OF RECORDS (P.05.12.02)

- Each participant in a competition has 2 types of information:
  - code: is the candidate code (string of characters length from 2 to 10)
  - score: score (an integer from 0 to 1000000, scores of participants are different)
- Write a program to sort participants in descending order of scores
- Data
  - Each line contains 2 information and the code and score of a candidate
  - Input data ends with a line containing #
- Result
  - Write out on each line of code the score of a candidate in the sorted list (SPACE separator).

| stdin | stdout |
|---|---|
| S00001 27412<br>S00002 22981<br>S00003 32561<br>S00004 10915<br>S00005 17566<br># | S00003 32561<br>S00001 27412<br>S00002 22981<br>S00005 17566<br>S00004 10915 |

# SORTING A LIST OF RECORDS - PSEUDOCODE

- Data structure definition

```
struct Candidate{
  code;
  score;
}
```

- Use an array of pointers, each pointer points to 1 record (dynamic allocation)
- When swapping two elements, only swap two pointers (no need to swap the memory contents of the two records).
- Heap sorting algorithm is applied

```
Heapify(a, i, n){
  L = 2*i; R = 2*i+1; minIdx = i;
  if L <= n and a[L].score < a[minIdx].score then minIdx = L;
  if R <= n and a[R].score < a[minIdx].score then minIdx = R;
  if minIdx != i then {
     swap(a[i], a[minIdx]); heapify(minIdx, n);
  }
}
BuildHeap(){
  for i = n/2 downto 1 do Heapify(i, n);
}
HeapSort(){
  BuildHeap();
  for i = n downto 2 do {
     swap(a[1], a[i]); Heapify(1, i-1);
  }
}
```

# FINDING THE GREATEST COMMON SUBSET OF TWO SETS (P.05.12.03)

- Given two sets of integers $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_m\}$. Find the subset of A and B that has the most elements.
- Data
  - Line 1: Two positive integers n and m ($1 <= n, m <= 100000$)
  - Line 2: n positive integers $a_1, a_2, \ldots, a_n$ ($1 <= a_i <= 1000000$)
  - Line 3: m positive integers $b_1, b_2, \ldots, b_m$ ($1 <= b_i <= 1000000$)
- Result
  - The number of elements of the found set

| stdin | stdout |
|---|---|
| 6 6<br>7 3 10 1 2 8<br>6 2 8 10 5 7 | 4 |

# FINDING THE GREATEST COMMON SUBSET OF TWO SETS - PSEUDOCODE

- Sort two lists $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_m$ in ascending order
- Use two 2 indices $i$ and $j$ iterating from left to right over $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_m$, respectively
  - If $a_i = b_j$ then increase counter by 1, increase i and j by 1
  - If $a_i < b_j$ then increase i by 1
  - If $a_i > b_j$ then inrcease j by 1

```
maxCommonSubset(a, b, n, m){
  sắp xếp a và b theo thứ tự tăng dần;
  cnt = 0;
  i = 1; j = 1;
  while i <= n and j <= m do {
    if a[i] = b[j] then cnt = cnt + 1;
    else if a[i] > b[j] then j = j + 1;
    else i = i + 1;
  }
  return cnt;
}
```

HUST

hust.edu.vn fb.com/dhbkhn

# THANK YOU !