



# 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

# THUẬT TOÁN ỨNG DỤNG

Giới thiệu chung  
Thư viện cấu trúc dữ liệu

ONE LOVE. ONE FUTURE.

- Giới thiệu chung
- Thư viện cấu trúc dữ liệu STL trong C++

# GIỚI THIỆU CHUNG

- Mục tiêu học phần
  - Tiếp cận một số cấu trúc dữ liệu và thuật toán nâng cao
  - Ứng dụng các kỹ thuật thuật toán và cấu trúc dữ liệu hiệu quả vào việc giải các bài toán tính toán phức tạp
  - Phân tích hiệu quả của thuật toán
  - Rèn luyện kỹ năng thực hành lập trình thuật toán
- Thực hành
  - Lập trình giải các bài toán tính toán ứng dụng
  - Nộp (Submit) source code lên hệ thống chấm điểm tự động qua các testcase
    - Mỗi bài tập sẽ được mô tả chi tiết về phát biểu bài toán, định dạng dữ liệu vào và kết quả đầu ra

- Chủ đề
  - Quay lui, nhánh và cận
  - Cấu trúc dữ liệu: ngăn xếp (stack), hàng đợi (queue), tập hợp (set), các tập rời nhau (disjoint set), hàng đợi ưu tiên (priority queue), cây phân đoạn (segment tree),
  - Kỹ thuật mảng cộng dồn, kỹ thuật 2 con trỏ, biểu diễn và xử lý trên bit
  - Thuật toán tham lam, chia để trị, quy hoạch động
  - Thuật toán trên đồ thị: DFS, BFS, Strongly Connected Components, Shortest Path, Minimum Spanning Tree, Max-Flow, Max-Matching
  - Thuật toán hình học
  - Thuật toán xử lý xâu

# GIỚI THIỆU CHUNG - Ví dụ minh họa (P.01.01.01)

- Cho 2 số nguyên a và b, hãy tính tổng của 2 số đó.
- **Dữ liệu**
  - Dòng 1 chứa 2 số nguyên a và b ( $0 \leq a, b \leq 10^{19}$ )
- **Kết quả**
  - Ghi ra giá trị là tổng của a và b

Stdin	Stdout
3 5	8

# GIỚI THIỆU CHUNG - Ví dụ minh họa (P.01.01.01)

- Cho 2 số nguyên a và b, hãy tính tổng của 2 số đó.
- **Dữ liệu**
  - Dòng 1 chứa 2 số nguyên a và b ( $0 \leq a, b \leq 10^{19}$ )
- **Kết quả**
  - Ghi ra giá trị là tổng của a và b

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    int a,b;
    cin >> a >> b;
    int res = a + b;
    cout << res;
    return 0;
}
```



# GIỚI THIỆU CHUNG - Ví dụ minh họa (P.01.01.01)

- Cho 2 số nguyên a và b, hãy tính tổng của 2 số đó.
- Dữ liệu
  - Dòng 1 chứa 2 số nguyên a và b ( $0 \leq a, b \leq 10^{19}$ )
- Kết quả
  - Ghi ra giá trị là tổng của a và b

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    int a,b;
    cin >> a >> b;
    int res = a + b;
    cout << res;
    return 0;
}
```

Tràn số khi a và b lớn

# GIỚI THIỆU CHUNG - Ví dụ minh họa (P.01.01.01)

- Cho 2 số nguyên a và b, hãy tính tổng của 2 số đó.
- **Dữ liệu**
  - Dòng 1 chứa 2 số nguyên a và b ( $0 \leq a, b \leq 10^{19}$ )
- **Kết quả**
  - Ghi ra giá trị là tổng của a và b

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    unsigned long long a,b;
    cin >> a >> b;
    unsigned long long res = a + b;
    cout << res;
    return 0;
}
```

# GIỚI THIỆU CHUNG - Ví dụ minh họa (P.01.01.01)

- Cho 2 số nguyên a và b, hãy tính tổng của 2 số đó.
- Dữ liệu
  - Dòng 1 chứa 2 số nguyên a và b ( $0 \leq a, b \leq 10^{19}$ )
- Kết quả
  - Ghi ra giá trị là tổng của a và b

a = 10000000000000000000, b = 10000000000000000000  
→ res = 1553255926290448384 (vẫn tràn số)

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    unsigned long long a,b;
    cin >> a >> b;
    unsigned long long res = a + b;
    cout << res;
    return 0;
}
```

# GIỚI THIỆU CHUNG - Ví dụ minh họa (P.01.01.01)

- Cho 2 số nguyên a và b, hãy tính tổng của 2 số đó.
- Dữ liệu
  - Dòng 1 chứa 2 số nguyên a và b ( $0 \leq a, b \leq 10^{19}$ )
- Kết quả
  - Ghi ra giá trị là tổng của a và b

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    unsigned long long a,b, a1, b1,a2,b2;
    cin >> a >> b;
    a1=a/10;    b1=b/10;
    a2 = a%10;    b2 = b%10;
    unsigned long long c1 = a1+b1+(a2+b2)/10;
    unsigned long long c2 = (a2+b2)%10;
    if(c1 > 0) cout << c1 << c2;
    else cout << c2;
    return 0;
}
```

**SOLVED!!**



# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++

- Thư viện STL của C++
  - Vector, List
  - String
  - Stack, Queue
  - Set
  - Map
  - Priority queue cài đặt bằng heap



# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - vector

- Mảng động

iterator: con trỏ nhúng trong vector

- Lưu các phần tử tuyến tính
- Các thao tác: truy cập, thêm mới, loại bỏ các phần tử

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    vector<int> V;
    V.push_back(1); V.push_back(2);
    for(int i = 3; i <= 10; i++) V.push_back(i);
    for(int i = 0; i < V.size(); i++) cout << V[i] << " ";
    cout << endl;
    V.erase(V.begin(), V.begin() + 3);
    for(int i = 0; i < V.size(); i++) cout << V[i] << " ";
    cout << endl;
}
```



iteretor<int>

```
1 2 3 4 5 6 7 8 9 10
4 5 6 7 8 9 10
```

# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - List

- Danh sách liên kết đôi

đuyệt qua danh sách liên kết dùng con tr `list<int>::iterator p`

- Lưu các phần tử tuyến tính
- Các thao tác: thêm phần tử vào đầu, cuối, vào sau 1 vị trí, loại bỏ 1 phần tử khỏi danh sách

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    list<int> L;
    for(int v = 1; v <= 5; v++) L.push_back(v);
    list<int>::iterator p; //p là con tr tr vào các ph n t trong danh sách
    p = L.begin();
    advance(p,2); //ti n con tr lên 2 v trí
    L.insert(p,2,6); //insert 2 occurrences of 6 after position p
    for(p = L.begin(); p!= L.end(); p++) cout << *p << " ";
}
```

1 2 6 6 3 4 5

# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - List

- Danh sách liên kết đôi
  - Lưu các phần tử tuyến tính
  - Các thao tác: thêm phần tử vào đầu, cuối, vào sau 1 vị trí, loại bỏ 1 phần tử khỏi danh sách

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    list<int> L;
    for(int v = 1; v <= 5; v++) L.push_back(v);
    list<int>::iterator p;
    p = L.begin(); advance(p,2);
    cout << "item at position p is " << *p << endl;
    L.erase(p); //remove the item at position p
    for(p = L.begin(); p != L.end(); p++) cout << *p << " ";
}
```



```
item at position p is 3
1 2 4 5
```



# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - String

- Biểu diễn chuỗi các ký tự
- Thao tác: gán, ghép xâu, thay thế xâu con, trích xuất xâu con,...

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    string s1 = "hello";
    string s2 = s1 + " world";
    cout << "s1 = " << s1 << ", s2 = " << s2 << endl;
    string ss = s2.substr(2,6);
    cout << "s2 = " << s2 << ", length = " << s2.length() << endl;
    cout << "s2.substring(2,6) = " << ss << endl;
    s2.replace(6, 5, "abc");
    cout << "new s2 = " << s2 << endl;
}
```



```
s1 = hello, s2 = hello world
s2 = hello world, length = 11
s2.substring(2,6) = llo wo
new s2 = hello abc
```

# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - Stack

- Cấu trúc tuyến tính
- Thao tác: thêm và loại bỏ phần tử với nguyên tắc Last In First Out - LIFO

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    stack<int> S;
    for(int i = 1; i <= 5; i++){
        S.push(i); cout << "PUSH " << i << endl;
    }
    while(!S.empty()){
        int e = S.top(); S.pop(); cout << "POP " << e << endl;
    }
}
```



```
PUSH 1
PUSH 2
PUSH 3
PUSH 4
PUSH 5
POP 5
POP 4
POP 3
POP 2
POP 1
```

# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - Queue

- Cấu trúc tuyến tính
- Thao tác: thêm và loại bỏ phần tử với nguyên tắc First In First Out - FIFO

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    queue<int> Q;
    for(int e = 1; e <= 5; e++){
        Q.push(e); cout << "Queue push " << e << endl;
    }
    while(!Q.empty()){
        int e = Q.front(); Q.pop(); cout << "Queue POP " << e << endl;
    }
}
```



```
Queue push 1
Queue push 2
Queue push 3
Queue push 4
Queue push 5
Queue POP 1
Queue POP 2
Queue POP 3
Queue POP 4
Queue POP 5
```

# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - Set

- Lưu các phần tử không lặp giá trị **không** **cs p x p**
- Thao tác: thêm, loại bỏ, tìm kiếm thực hiện nhanh

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    set<string> S;
    S.insert("abc"); S.insert("def"); S.insert("xyz");
    S.insert("abc");

    set<string>::iterator p; duy t các ph n t c a set dùng iterator
    for(p = S.begin(); p != S.end(); p++) cout << *p << " ";
    cout << endl;
}
```

abc def xyz

# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - Set

- Lưu các phần tử không lặp giá trị **cây AVL**
- Thao tác: thêm, loại bỏ, tìm kiếm thực hiện nhanh

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    set<string> S;
    S.insert("abc"); S.insert("def"); S.insert("xyz");
    string s1 = "def";
    set<string>::iterator p = S.find(s1);
    if(p == S.end())
        cout << "xau " << s1 << " does not exist" << endl;
    else
        cout << "xau " << s1 << " exists in S" << endl;
}
```

xau def exists in S

# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - Set

- Lưu các phần tử không lặp giá trị
- Thao tác: thêm, loại bỏ, tìm kiếm thực hiện nhanh

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    set<string> S;
    S.insert("abc"); S.insert("def"); S.insert("xyz");
    string s1 = "xyz";
    S.erase(s1);
    set<string>::iterator p;
    for(p = S.begin(); p != S.end(); p++) cout << *p << " ";
    cout << endl;
}
```



abc def

# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - Set

- Cấu trúc set trong C++ cung cấp hàm `upper_bound(k)`: trả về con trỏ đến phần tử nhỏ nhất mà lớn hơn  $k$  trong tập hợp. Nếu  $k$  lớn hơn hoặc bằng phần tử lớn nhất thì hàm trả về con trỏ đến vị trí sau phần tử cuối cùng của tập hợp

bài toán truy vấn các phần tử như như còn lại như như.


```
#include <bits/stdc++.h>
using namespace std;
int main() {
    set<int> S;
    for(int v = 1; v <= 5; v++) S.insert(2*v);
    set<int>::iterator p = S.upper_bound(3);
    cout << "upper_bound(3) = " << *p << endl;
    p = S.upper_bound(4);
    cout << "upper_bound(4) = " << *p << endl;
    p = S.upper_bound(10);
    if(p == S.end()) cout << "no upper_bound of 10" << endl;
}
```

upper\_bound(3) = 4  
upper\_bound(4) = 6  
no upper\_bound of 10

# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - Set

- Cấu trúc set trong C++ cung cấp hàm `lower_bound(k)`: trả về con trỏ đến phần tử có giá trị bằng  $k$  (nếu  $k$  thuộc tập hợp) hoặc phần tử nhỏ nhất mà lớn hơn  $k$  trong tập hợp (nếu  $k$  không thuộc tập hợp). Nếu  $k$  lớn hơn phần tử lớn nhất thì hàm trả về con trỏ đến vị trí sau phần tử cuối cùng của tập hợp

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    set<int> S;
    for(int v = 1; v <= 5; v++) S.insert(2*v);
    set<int>::iterator p = S.lower_bound(3);
    cout << "lower_bound(3) = " << *p << endl;
    p = S.lower_bound(4);
    cout << "lower_bound(4) = " << *p << endl;
    p = S.lower_bound(11);
    if(p == S.end()) cout << "no lower_bound of 11" << endl;
    else cout << "lower_bound of 11 = " << *p << endl;
}
```



```
lower_bound(3) = 4
lower_bound(4) = 4
no lower_bound of 11
```



# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - Map

- Cấu trúc dữ liệu lưu cặp khóa, giá trị
- Thao tác: thêm cặp khóa, giá trị; truy vấn giá trị tương ứng với 1 khóa cho trước.

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    map<string, int> M;
    M["abc"] = 1; M["def"] = 2; M["xyzt"] = 10;
    string k = "abc";
    cout << "value of key " << k << " = " << M[k] << endl;
    for(map<string,int>::iterator p = M.begin(); p != M.end(); p++)
        cout << p->first << " is mapped to value " << p->second << endl;
    string k1 = "1234";
    cout << "value of " << k1 << " = " << M[k1] << endl;
}
```

*string k1 ud li uc a khóa*  
*int: k1 ud li uc a value* (key,value)

value of key abc = 1  
xyzt is mapped to value 10  
abc is mapped to value 1  
def is mapped to value 2  
value of 1234 = 0

# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - Priority Queue

- Lưu các phần tử, truy xuất phần tử có khóa lớn nhất/nhỏ nhất một cách hiệu quả cài đặt bằng heap

```
#include <bits/stdc++.h>
#define pii pair<int,int>
using namespace std;
int main() {
    priority_queue<int> pq;
    pq.push(5);    pq.push(1);    pq.push(100);    pq.push(30);
    while(!pq.empty()){
        int e = pq.top(); pq.pop();
        cout << "pq pop " << e << endl;
    }
}
```



```
pq pop 100
pq pop 30
pq pop 5
pq pop 1
```

# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - Priority Queue

- Lưu các phần tử, truy xuất phần tử có khóa lớn nhất/nhỏ nhất một cách hiệu quả

```
#include <bits/stdc++.h>
#define pii pair<int,int>
using namespace std;
int main() {
    priority_queue<pii> PQ;
    PQ.push(make_pair(4,-40));
    PQ.push(make_pair(1,-10));
    PQ.push(make_pair(9,-900));
    while(!PQ.empty()){
        pii e = PQ.top(); PQ.pop();
        cout << "PQ pop (" << e.first << "," << e.second << ")" << endl;
    }
}
```



```
PQ pop (9,-900)
PQ pop (4,-40)
PQ pop (1,-10)
```

# THƯ VIỆN CẤU TRÚC DỮ LIỆU STL TRONG C++ - Priority Queue

- Lưu các phần tử, truy xuất phần tử có khóa lớn nhất/nhỏ nhất một cách hiệu quả

```
#include <bits/stdc++.h>
#define pii pair<int,int>
using namespace std;
int main() {
    priority_queue<pii, vector<pii>, greater<pii> > PQ;
    PQ.push(make_pair(4,-40));
    PQ.push(make_pair(1,-10));
    PQ.push(make_pair(9,-900));
    while(!PQ.empty()){
        pii e = PQ.top(); PQ.pop();
        cout << "PQ pop (" << e.first << "," << e.second << ")" << endl;
    }
}
```

pq là hàng đợi ưu tiên lưu các cặp (int, int)  
ưu tiên lưu các cặp có khóa nhỏ hơn  
(khóa m c nhỏ là phần tử ưu tiên)

PQ pop (1, -10)  
PQ pop (4, -40)  
PQ pop (9, -900)



A large graphic on the left side of the slide. It features a dark blue background with a circular pattern of red dots of varying sizes, creating a sense of depth and movement. The word "HUST" is centered within this graphic in a bold, white, sans-serif font.

# HUST

# THANK YOU !