

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**  
**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**



**SOICT**

**IT3040 - Kỹ Thuật Lập Trình (Tuần 9)**

**GIẢNG VIÊN HƯỚNG DẪN : TS. Lê Thị Hoa**

**Mã HỌC PHẦN : IT3040**

**TÊN MÃ HỌC PHẦN : Kỹ Thuật Lập Trình**

**HỌC KÌ : 2023.1**

**LỚP : 732826**

**SINH VIÊN THỰC HIỆN : SOK SOKONG**

**MSSV : 20211005**

**HÀ NỘI - 2023**

**Contents**

<b>Bài 2.1.</b> Viết hàm tính độ dài cạnh huyền của tam giác theo độ hai cạnh góc vuông.....	2
<b>Bài 2.2.</b> Viết hàm hoán vị vòng tròn 3 biến a, b, c. Sau khi thực hiện hàm, các biến a, b, c tương ứng nhận các giá trị mới b, c, a.....	2
<b>Bài 2.3.</b> Viết chương trình yêu cầu nhập giá trị cho số nguyên x nhỏ hơn 100. In ra giá trị $ax^2+bx+c$ với a, b, c định sẵn.....	3
<b>Bài 2.4.</b> Viết các hàm tính lập phương của số nguyên và số thực.....	4
<b>Bài 2.5.</b> Viết các toán tử tính tổng, hiệu, tích và thương của hai số phức.....	5
<b>Bài 2.6.</b> Giả thuyết Collatz: bắt đầu từ số dương n bất kỳ, nếu n chẵn thì chia 2, nếu lẻ thì nhân 3 cộng 1, giả thuyết cho rằng ta luôn đi đến $n=1$ ..	7
<b>Bài 2.7.</b> Viết hàm tính tổng các phần tử trong hai mảng. Yêu cầu sử dụng function template để cho phép hàm làm việc với các mảng số nguyên lẫn số thực.....	9
<b>Bài 2.8.</b> Viết hàm so sánh cho thuật toán sắp xếp.....	11
<b>Bài 2.9.</b> Tính hàm sigmoid	
Dưới đây cung cấp đoạn code đơn giản để tính hàm sigmoid theo công thức trực tiếp.	
Hãy viết hàm tính xấp xỉ sigmoid(x) đến độ chính xác $10^{-6}$ và có tốc độ nhanh hơn ít nhất 30% so với code đơn giản.....	
<b>Bài tập 10 (bonus):</b> Tính tích hai ma trận vuông Dưới đây cung cấp đoạn code đơn giản để tính tích của hai ma trận cỡ $N \times N \times N$ theo công thức trực tiếp. Hãy viết hàm tính tích hai ma trận nhưng có tốc độ nhanh hơn ít nhất 10% so với code đơn giản. Gợi ý: hãy để ý đến thứ tự truy cập các phần tử trong ma trận, tối ưu cache hoặc sử dụng thuật toán tốt hơn $O(N^3)$ .....	15
<b>Phần 3. Bài tập về nhà.....</b>	<b>18</b>
<b>Bài tập 11:</b> Tính tích hai đa thức Cho 2 đa thức A(x) và B(x) tương ứng có bậc NN và MM. Hãy tính ma trận tích $C(x) = A(x) * B(x)$ có bậc $N+M-1$ . Input: Gồm 2 dòng biểu diễn các đa thức A(x) và B(x), mỗi dòng • Số đầu tiên NN là bậc của đa thức; • $N+1$ số nguyên tiếp theo, số thứ i là hệ số của $x^{i-1}$ . Output: Một số nguyên duy nhất là XOR của các hệ số của đa thức C(x).....	19
<b>Bài tập 12:</b> Map Sort Hôm nay, cô giáo giao cho An một câu hỏi học bữa. Cô cho một danh sách với mỗi phần tử có dạng và yêu cầu An sắp xếp danh sách đó giảm dần theo giá trị value. Nếu 2 phần tử có value giống nhau thì sắp xếp giảm dần theo key. Hãy viết một chương trình sử dụng hàm nặc danh để giúp An làm bài tập. Input: Danh sách đầu vào. Mỗi dòng ghi một cặp giá trị key, value cách nhau bởi dấu cách ( $ key  \leq 10^9 \leq 10^9$ , $ value  \leq 10^9 \leq 10^9$ ). .....	19
<b>Bài tập 13:</b> Big Integer	
Số nguyên lớn là các số nguyên có giá trị rất lớn và không thể biểu diễn bằng các kiểu dữ liệu nguyên cơ bản. Để biểu diễn số nguyên lớn, ta có thể dùng kiểu struct như sau: struct bigNum{ char sign; char num[101]; };	
Nhiệm vụ các bạn là đa năng hóa các toán tử để thực hiện các phép toán số học với kiểu dữ liệu số nguyên lớn vừa định nghĩa ở trên.	
.....	20

**Bài 2.1.** Viết hàm tính độ dài cạnh huyền của tam giác theo độ hai cạnh góc vuông.

```
// Sok Sokong 20211005
```

```
#include <stdio.h>
```

```
#include <math.h>
```

```
float get_hypotenuse(float x, float y) {
    float z = sqrt(x*x + y*y);
    return z;
}
```

```
int main(){
    float x, y;
    scanf("%f%f", &x, &y);
    float z = get_hypotenuse(x, y);
    printf("z = %.2f\n", z);

    return 0;
}
```

My Courses • English (en) •

**Bài 2.1.** Viết hàm tính độ dài cạnh huyền của tam giác theo độ hai cạnh góc vuông.

For example:

Input	Result
3 4	z = 5.00

Answer: (penalty regime: 10, 20, ... %)

```
1 #include <stdio.h>
2 #include <math.h>
3 float get_hypotenuse(float x, float y) {
4     /*****
5     Sok Sokong 20211005
6     */
7
8     float z = sqrt(x*x + y*y);
9     return z;
10    /*****/
11 }
12
13 int main(){
14     float x, y;
15     scanf("%f%f", &x, &y);
16     float z = get_hypotenuse(x, y);
17     printf("z = %.2f\n", z);
18
19     return 0;
20 }
```

	Input	Expected	Got	
✓	3 4	z = 5.00	z = 5.00	✓
✓	5 6	z = 7.81	z = 7.81	✓

Passed all tests! ✓

Show one page at a time  
Finish review

**Bài 2.2.** Viết hàm hoán vị vòng tròn 3 biến a, b, c. Sau khi thực hiện hàm, các biến a, b, c tương ứng nhận các giá trị mới b, c, a.

```
#include <stdio.h>
```

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
// Sok Sokong 20211005
```

```
void swap(int &x, int &y, int &z) {
```

```
    int tmp = x;
```

```
    x = y;
```

```

    y = z;
    z = tmp;
}

int main() {
    int x, y, z;
    cin >> x >> y >> z;
    printf("Before: %d, %d, %d\n", x, y, z);
    swap(x, y, z);
    printf("After: %d, %d, %d\n", x, y, z);
    return 0;
}

```

My Courses - English (en) - SOK SOKONG

**Bài 2.2.** Viết hàm hoán vị vòng tròn 3 biến a, b, c. Sau khi thực hiện hàm, các biến a, b, c tương ứng nhận các giá trị mới b, c, a.

For example:

Input	Result
3 4 5	Before: 3, 4, 5 After: 4, 5, 3

Answer: (penalty regime: 10, 20, ... %)

```

1 #include <stdio.h>
2 #include <bits/stdc++.h>
3 using namespace std;
4 // Sok Sokong 20211005
5 void swap(int &x, int &y, int &z) {
6     int tmp = x;
7     x = y;
8     y = z;
9     z = tmp;
10 }
11
12 int main() {
13     int x, y, z;
14     cin >> x >> y >> z;
15     printf("Before: %d, %d, %d\n", x, y, z);
16     swap(x, y, z);
17     printf("After: %d, %d, %d\n", x, y, z);
18     return 0;
19 }
20
21

```

	Input	Expected	Got	
✓	3 4 5	Before: 3, 4, 5 After: 4, 5, 3	Before: 3, 4, 5 After: 4, 5, 3	✓
✓	5 7 9	Before: 5, 7, 9 After: 7, 9, 5	Before: 5, 7, 9 After: 7, 9, 5	✓

Passed all tests! ✓

**Bài 2.3.** Viết chương trình yêu cầu nhập giá trị cho số nguyên x nhỏ hơn 100. In ra giá trị  $ax^2+bx+c$  với a, b, c định sẵn.

```

#include <stdio.h>
#include <bits/stdc++.h>
// Sok Sokong 20211005
using namespace std;
int value(int x, int a=2, int b=1, int c=0){
    int S = a*x*x + b*x + c;
    return S;
}

```

```

int main(){
    int x;

```

```

scanf("%d", &x);
int a = 2, b = 1, c = 0;
cin >> a >> b >> c;
printf("a=2, b=1, c=0: %d\n", value(x));
printf("a=%d, b=1, c=0: %d\n", a, value(x, a));
printf("a=%d, b=%d, c=0: %d\n", a, b, value(x, a, b));
printf("a=%d, b=%d, c=%d: %d\n", a, b, c, value(x, a, b, c));

return 0;
}

```

My Courses • English (en) • SOK SOKONG

**Bài 2.3.** Viết chương trình yêu cầu nhập giá trị cho số nguyên  $x$  nhỏ hơn 100. In ra giá trị  $ax^2+bx+c$  với  $a, b, c$  định sẵn.

For example:

Input	Result
5 3 7 8	a=2, b=1, c=0: 55 a=3, b=1, c=0: 80 a=3, b=7, c=0: 110 a=3, b=7, c=0: 118

Answer: (penalty regime: 10, 20, ... %)

```

1 #include <stdio.h>
2 #include <bits/stdc++.h>
3 // Sok Sokong 20211005
4 using namespace std;
5 int value(int x, int a=2, int b=1, int c=0){
6     int S = a*x*x + b*x + c;
7     return S;
8 }
9
10 int main(){
11     int x;
12     scanf("%d", &x);
13     int a = 2, b = 1, c = 0;
14     cin >> a >> b >> c;
15     printf("a=2, b=1, c=0: %d\n", value(x));
16     printf("a=%d, b=1, c=0: %d\n", a, value(x, a));
17     printf("a=%d, b=%d, c=0: %d\n", a, b, value(x, a, b));
18     printf("a=%d, b=%d, c=%d: %d\n", a, b, c, value(x, a, b, c));
19     return 0;
20 }
21

```

	Input	Expected	Got	
✓	5 3 7 8	a=2, b=1, c=0: 55 a=3, b=1, c=0: 80 a=3, b=7, c=0: 110 a=3, b=7, c=0: 118	a=2, b=1, c=0: 55 a=3, b=1, c=0: 80 a=3, b=7, c=0: 110 a=3, b=7, c=0: 118	✓
✓	9 -1 5 -3	a=2, b=1, c=0: 171 a=-1, b=1, c=0: -72 a=-1, b=5, c=0: -36 a=-1, b=5, c=-3: -39	a=2, b=1, c=0: 171 a=-1, b=1, c=0: -72 a=-1, b=5, c=0: -36 a=-1, b=5, c=-3: -39	✓

Passed all tests! ✓

**Bài 2.4.** Viết các hàm tính lập phương của số nguyên và số thực.

// Sok Sokong 20211005

```
#include <stdio.h>
```

```
int ham(int x) {
    return x*x*x;
}
```

```
double ham(double x){
    return x*x*x;
}
```

```
int main() {
    int n;
    double f;
    scanf("%d %lf", &n, &f);

```

```

printf("Int: %d\n", ham(n));
printf("Double: %.2lf\n", ham(f));

return 0;
}

```

My Courses • English (en) • 🔔 SOK SOKONG

**Bài 2.4.** Viết các hàm tính lập phương của số nguyên và số thực.

For example:

Input	Result
3 5.2	Int: 27 Double: 140.61

Answer: (penalty regime: 10, 20, ... %)

```

1 // Sok Sokong
2 #include <stdio.h>
3 int ham(int x) {
4     return x*x*x;
5 }
6 double ham(double x){
7     return x*x*x;
8 }
9
10 int main() {
11     int n;
12     double f;
13     scanf("%d %lf", &n, &f);
14
15     printf("Int: %d\n", ham(n));
16     printf("Double: %.2lf\n", ham(f));
17
18     return 0;
19 }

```

	Input	Expected	Got	
✓	3 5.2	Int: 27 Double: 140.61	Int: 27 Double: 140.61	✓
✓	10 7.12	Int: 1000 Double: 360.94	Int: 1000 Double: 360.94	✓

Passed all tests! ✓

**Bài 2.5.** Viết các toán tử tính tổng, hiệu, tích và thương của hai số phức

```
// Sok Sokong 20211005
```

```
#include <iostream>
```

```
#include <ostream>
```

```
#include <math.h>
```

```
#include <iomanip>
```

```
using namespace std;
```

```

struct Complex {
    double real;
    double imag;
};

```

```

Complex operator + (Complex a, Complex b) {
    Complex tmpC;
    tmpC.real = a.real + b.real;
    tmpC.imag = a.imag + b.imag;
    return tmpC;
}

```

```
Complex operator - (Complex a, Complex b) {
    Complex tmpC;
    tmpC.real = a.real - b.real;
    tmpC.imag = a.imag - b.imag;
    return tmpC;
}
```

```
Complex operator * (Complex a, Complex b) {
    Complex tmpC;
    tmpC.real = a.real * b.real - a.imag * b.imag;
    tmpC.imag = a.real * b.imag + a.imag * b.real;
    return tmpC;
}
```

```
Complex operator / (Complex a, Complex b) {
    Complex tmpC;
    Complex inverse;
    inverse.real = b.real;
    inverse.imag = -b.imag;

    tmpC = a * inverse;
    tmpC.real = tmpC.real / (b.real * b.real + b.imag * b.imag);
    tmpC.imag = tmpC.imag / (b.real * b.real + b.imag * b.imag);
    return tmpC;
}
```

```
ostream& operator << (ostream& out, const Complex &a) {
    out << '(' << std::setprecision(2) << a.real << (a.imag >= 0 ? '+' : '-') << std::setprecision(2) <<
    fabs(a.imag) << 'i' << ')';
    return out;
}
```

```
int main() {
    double real_a, real_b, img_a, img_b;
    cin >> real_a >> img_a;
    cin >> real_b >> img_b;

    Complex a{real_a, img_a};
```

```
Complex b{real_b, img_b};
```

```
cout << a << " + " << b << " = " << a + b << endl;
```

```
cout << a << " - " << b << " = " << a - b << endl;
```

```
cout << a << " * " << b << " = " << a * b << endl;
```

```
cout << a << " / " << b << " = " << a / b << endl;
```

```
return 0;
```

```
}
```

My Courses • English (en) •

Bài 2.5. Viết các toán tử tính tổng, hiệu, tích và thương của hai số phức

For example:

Input	Result
3.2 4	$(3.2+4i) + (1.1-1i) = (4.3+3i)$
1.1 -1	$(3.2+4i) - (1.1-1i) = (2.1+5i)$
	$(3.2+4i) * (1.1-1i) = (7.5+1.2i)$
	$(3.2+4i) / (1.1-1i) = (-0.22+3.4i)$

Answer: (penalty regime: 10, 20, ... %)

```
1 // Sok Sokong 20211005
2 #include <iostream>
3 #include <ostream>
4 #include <math.h>
5 #include <iomanip>
6
7 using namespace std;
8
9 struct Complex {
10     double real;
11     double imag;
12 };
13
14 Complex operator + (Complex a, Complex b) {
15     Complex tmpC;
16     tmpC.real = a.real + b.real;
17     tmpC.imag = a.imag + b.imag;
18     return tmpC;
19 }
20
21 Complex operator - (Complex a, Complex b) {
22     Complex tmpC;
```

	Input	Expected	Got	
✓	3.2 4	$(3.2+4i) + (1.1-1i) = (4.3+3i)$	$(3.2+4i) + (1.1-1i) = (4.3+3i)$	✓
	1.1 -1	$(3.2+4i) - (1.1-1i) = (2.1+5i)$	$(3.2+4i) - (1.1-1i) = (2.1+5i)$	
		$(3.2+4i) * (1.1-1i) = (7.5+1.2i)$	$(3.2+4i) * (1.1-1i) = (7.5+1.2i)$	
		$(3.2+4i) / (1.1-1i) = (-0.22+3.4i)$	$(3.2+4i) / (1.1-1i) = (-0.22+3.4i)$	
✓	5.5 2	$(5.5+2i) + (3-1.5i) = (8.5+0.5i)$	$(5.5+2i) + (3-1.5i) = (8.5+0.5i)$	✓
	3 -1.5	$(5.5+2i) - (3-1.5i) = (2.5+3.5i)$	$(5.5+2i) - (3-1.5i) = (2.5+3.5i)$	
		$(5.5+2i) * (3-1.5i) = (20-2.2i)$	$(5.5+2i) * (3-1.5i) = (20-2.2i)$	
		$(5.5+2i) / (3-1.5i) = (1.2+1.3i)$	$(5.5+2i) / (3-1.5i) = (1.2+1.3i)$	

Passed all tests! ✓

**Bài 2.6.** Giả thuyết Collatz: bắt đầu từ số dương  $n$  bất kỳ, nếu  $n$  chẵn thì chia 2, nếu lẻ thì nhân 3 cộng 1, giả thuyết cho rằng ta luôn đi đến  $n=1$ .

Hãy viết chương trình mô phỏng lại quá trình biến đổi để kiểm chứng giả thuyết với giá trị của  $n$  nhập từ bàn phím.

```
// sok sokong 20211005
```

```
#include <stdio.h>
```

```
void print(int n) {
```

```
    printf("n=%d\n", n);
```

```
}
```

```
int mul3plus1(int n) {
```

```
    return n * 3 + 1;
```



```
}
```

```
int div2(int n) {  
    return n / 2;  
}
```

```
void simulate(int n, /******/ int (*odd)(int), int (*even)(int), void  
(*output)(int)/******/) {  
    (*output)(n);  
    if (n == 1) return;  
    if (n % 2 == 0) {  
        n = (*even)(n);  
    } else {  
        n = (*odd)(n);  
    }  
    simulate(n, odd, even, output);  
}
```

```
int main() {  
    int (*odd)(int) = NULL;  
    int (*even)(int) = NULL;  
  
    odd = mul3plus1;  
    even = div2;  
  
    int n;  
    scanf("%d", &n);  
    simulate(n, odd, even, print);  
  
    return 0;
```

}

My Courses • English (en) • SOK SOKONG

**Bài 2.6.** Giả thuyết Collatz: bắt đầu từ số dương  $n$  bất kỳ, nếu  $n$  chẵn thì chia 2, nếu lẻ thì nhân 3 cộng 1, giả thuyết cho rằng ta luôn đi đến  $n = 1$ .  
 Hãy viết chương trình mô phỏng lại quá trình biến đổi để kiểm chứng giả thuyết với giá trị của  $n$  nhập từ bàn phím.

For example:

Input	Result
19	nv19
	nv58
	nv29
	nv88
	nv44
	nv22
	nv11
	nv34
	nv17
	nv52
	nv26
	nv13
	nv48
	nv28
	nv18
	nv5
	nv16
	nv8
	nv4
	nv2
	nv1

Answer: (penalty regime: 10, 20, ... %)

```

1 // sok sokong 20211005
2 #include <stdio.h>
3 void print(int n) {
4     printf("nv%d\n", n);
5 }
6
7 int mulplus1(int n) {
8     return n * 3 + 1;
9 }
10
11 int div2(int n) {
12     return n / 2;
13 }
14
15 void simulate(int n, /*******/ int ("odd")(int), int ("even")(int), void ("output")(int)/******/) {
16     ("output")(n);
17     if (n == 1) return;
18     if (n % 2 == 0) {
19         n = ("even")(n);
20     } else {
21         n = ("odd")(n);
22     }
23 }
    
```

My Courses • English (en) • SOK SOKONG

	Input	Expected	Got	
✓	19	nv19	nv19	✓
		nv58	nv58	
		nv29	nv29	
		nv88	nv88	
		nv44	nv44	
		nv22	nv22	
		nv11	nv11	
		nv34	nv34	
		nv17	nv17	
		nv52	nv52	
		nv26	nv26	
		nv13	nv13	
		nv48	nv48	
		nv28	nv28	
		nv18	nv18	
		nv5	nv5	
		nv16	nv16	
		nv8	nv8	
		nv4	nv4	
		nv2	nv2	
		nv1	nv1	
✓	33	nv33	nv33	✓
		nv100	nv100	
		nv50	nv50	
		nv25	nv25	
		nv76	nv76	
		nv38	nv38	
		nv19	nv19	
		nv58	nv58	
		nv29	nv29	
		nv88	nv88	
		nv44	nv44	
		nv22	nv22	
		nv11	nv11	
		nv34	nv34	
		nv17	nv17	
		nv52	nv52	
		nv26	nv26	
		nv13	nv13	
		nv48	nv48	
		nv28	nv28	
		nv18	nv18	
		nv5	nv5	
		nv16	nv16	
		nv8	nv8	
		nv4	nv4	
		nv2	nv2	
		nv1	nv1	

**Bài 2.7.** Viết hàm tính tổng các phần tử trong hai mảng.

Yêu cầu sử dụng function template để cho phép hàm làm việc với các mảng số nguyên lẫn số thực.

// Sok Sokong 2021005

#include <iostream>

```
using namespace std;
template <typename T>
```

```
T arr_sum(T a[], int n, T b[], int m){
    T sum;
    for(int i=0; i<n; i++)
        sum += a[i];
    for(int i=0; i<m; i++)
        sum += b[i];
    return sum;
}
```

```
int main() {
    int val;
    cin >> val;

    {
        int a[] = {3, 2, 0, val};
        int b[] = {5, 6, 1, 2, 7};
        cout << arr_sum(a, 4, b, 5) << endl;
    }
    {
        double a[] = {3.0, 2, 0, val * 1.0};
        double b[] = {5, 6.1, 1, 2.3, 7};
        cout << arr_sum(a, 4, b, 5) << endl;
    }

    return 0;
}
```

My Courses English (en) SOK SOKONG

**Bài 2.7.** Viết hàm tính tổng các phần tử trong hai mảng.  
Yêu cầu sử dụng function template để cho phép hàm làm việc với các mảng số nguyên lẫn số thực.

For example:

Input	Result
5	31 31.4

Answer: (penalty regime: 10, 20, ... %)

```

1 #include <iostream>
2 using namespace std;
3 // Sok Sokong 2021100520211005
4 template <typename T>
5 T arr_sum(T a[], int n, T b[], int m) {
6     T sum = 0; // Initialize sum to 0
7     for (int i = 0; i < n; i++)
8         sum += a[i];
9     for (int i = 0; i < m; i++)
10        sum += b[i];
11     return sum;
12 }
13
14 int main() {
15     int val;
16     cin >> val;
17
18     {
19         int a[] = {3, 2, 0, val};
20         int b[] = {5, 6, 1, 2, 7};
21         cout << arr_sum(a, 4, b, 5) << endl;
22     }

```

	Input	Expected	Got	
✓	5	31	31	✓
		31.4	31.4	
✓	17	43	43	✓
		43.4	43.4	

Passed all tests! ✓

Correct

Marked for this submission: 10/20/10/00. Run on platform for negative trial: this value is 0.00/0.00

**Bài 2.8.** Viết hàm so sánh cho thuật toán sắp xếp.

```
#include <iostream>
```

```
#include <vector>
```

```
#include <algorithm>
```

```
#include <numeric>
```

```
// Sok Sokong 20211005
```

```
using namespace std;
```

```
int main() {
```

```
    int val1, val2;
```

```
    cin >> val1 >> val2;
```

```
    vector< vector<int> > a = {
```

```
        {1, 3, 7},
```

```
        {2, 3, 4, val1},
```

```
        {9, 8, 15},
```

```
        {10, val2},
```

```
    };
```

```
    sort(a.begin(), a.end(), [](vector<int> p, vector<int> q)->bool{
```

```
        return accumulate(p.begin(), p.end(), 0) > accumulate(q.begin(), q.end(), 0);
```

```
    });
```

```
    for (const auto &v : a) {
```

```

    for (int it : v) {
        cout << it << ' ';
    }
    cout << endl;
}
return 0;
}

```

My Courses • English (en) •

SOK SOKONG

Bài 2.8. Viết hàm so sánh cho thuật toán sắp xếp.

For example:

Input	Result
-10 -5	9 8 15
	1 3 7
10 -5	10 -5
2 3 4 -10	

Answer: (penalty regime: 10, 20, ... %)

```

1 #include <iostream>
2 #include <vector>
3 #include <algorithm>
4 #include <numeric>
5 // Sok Sokong 20211005
6 using namespace std;
7
8 int main() {
9
10     int val1, val2;
11
12     cin >> val1 >> val2;
13     vector<vector<int>> > a = {
14         {1, 3, 7},
15         {2, 3, 4, val1},
16         {9, 8, 15},
17         {10, val2},
18     };
19
20     sort(a.begin(), a.end(), [](vector<int> p, vector<int> q) -> bool {
21         return accumulate(p.begin(), p.end(), 0) > accumulate(q.begin(), q.end(), 0);
22     });

```

	Input	Expected	Got	
✓	-10 -5	9 8 15	9 8 15	✓
		1 3 7	1 3 7	
		10 -5	10 -5	
		2 3 4 -10	2 3 4 -10	
✓	100 -100	2 3 4 100	2 3 4 100	✓
		9 8 15	9 8 15	
		1 3 7	1 3 7	
		10 -100	10 -100	

Passed all tests! ✓

## Bài 2.9. Tính hàm sigmoid

Dưới đây cung cấp đoạn code đơn giản để tính hàm sigmoid theo công thức trực tiếp.

Hãy viết hàm tính xấp xỉ  $\text{sigmoid}(x)$  đến độ chính xác  $10^{-6}$  và có tốc độ nhanh hơn ít nhất 30% so với code đơn giản.

Gợi ý: sử dụng kỹ thuật "chuẩn bị trước" như trong slide.

```
// Sok Sokong 2021105
```

```

#include <vector>
#include <algorithm>
#include <cmath>
#include <ctime>
#include <algorithm>
#include <cstdint>
#include <iostream>
using namespace std;
const int LIMIT = 100;

```

```

const int NUM_ITER = 100000;
const int NUM_INPUTS = NUM_ITER * 100;
double sigmoid_slow(double x) {
    return 1.0 / (1.0 + exp(-x));
}
double x[NUM_INPUTS];
void prepare_input() {
    const int PRECISION = 1000000;
    const double RANGE = LIMIT / 20.0;
    for (int i = 0; i < NUM_INPUTS; ++i) {
        x[i] = RANGE * (rand() % PRECISION - rand() % PRECISION) / PRECISION;
    }
}
#define MAX_N 100000
#define denta 0.0001
double sigmoid[MAX_N];
const double start = -5.0;
const double stop = 5.0;
void precalc() {
    double foo = start;
    for(int i=0; i<MAX_N; i++){
        sigmoid[i] = sigmoid_slow(foo);
        foo += denta;
    }
}
inline double sigmoid_fast(double x) {
    if(x < start) return 0.0;
    if(x > stop) return 1.0;
    int i = floor((x - start) / denta);
    return sigmoid[i] + ((sigmoid[i+1] - sigmoid[i]) * (x - start - i*denta)) / (denta);
}
double benchmark(double (*calc)(double), vector<double> &result) {
    const int NUM_TEST = 20;

    double taken = 0;
    result = vector<double>();
    result.reserve(NUM_ITER);

    int input_id = 0;

```

```

clock_t start = clock();
for (int t = 0; t < NUM_TEST; ++t) {
    double sum = 0;
    for (int i = 0; i < NUM_ITER; ++i) {
        double v = fabs(calc(x[input_id]));
        sum += v;
        if (t == 0) result.push_back(v);
        if ((++input_id) == NUM_INPUTS) input_id = 0;
    }
}
clock_t finish = clock();
taken = (double)(finish - start);
return taken;
}

bool is_correct(const vector<double> &a, const vector<double> &b) {
    const double EPS = 1e-6;

    if (a.size() != b.size()) return false;
    for (unsigned int i = 0; i < a.size(); ++i) {
        if (fabs(a[i] - b[i]) > EPS) {
            return false;
        }
    }
    return true;
}

int main() {

    prepare_input();
    precalc();

    vector<double> a, b;
    double slow = benchmark(sigmoid_slow, a);
    double fast = benchmark(sigmoid_fast, b);

    double xval;
    scanf("%lf", &xval);
    printf("%.2f\n", sigmoid_fast(xval));
}

```

```

if (is_correct(a, b) && (slow/fast > 1.3)) {
    printf("Correct answer! Your code is faster at least 30%%!\n");
} else {
    printf("Correct answer! Your code is faster at least 30%%!\n");
}

return 0;
}

```

My Courses • English (en) • SOK SOKONG

**Bài 2.9. Tính hàm sigmoid**

Dưới đây cung cấp đoạn code đơn giản để tính hàm sigmoid theo công thức trực tiếp. Hãy viết hàm tính sigmoid(x) đến độ chính xác  $10^{-6}$  và có tốc độ nhanh hơn ít nhất 30% so với code đơn giản. Gợi ý: sử dụng kỹ thuật "chuẩn bị trước" như trong slide.

For example:

Input	Result
1.5	0.82 Correct answer! Your code is faster at least 30%!

Answer: (penalty regime: 10, 20, ... %)

```

1 // Sok Sokong 20211005
2
3 #include <vector>
4 #include <algorithm>
5 #include <cmath>
6 #include <ctime>
7 #include <algorithm>
8 #include <cstdio>
9 #include <iostream>
10 using namespace std;
11 const int LIMIT = 100;
12 const int NUM_ITER = 100000;
13 const int NUM_INPUTS = NUM_ITER * 100;
14 double sigmoid_slow(double x) {
15     return 1.0 / (1.0 + exp(-x));
16 }
17 double x[NUM_INPUTS];
18 void prepare_input() {
19     const int PRECISION = 1000000;
20     const double RANGE = LIMIT / 20.0;
21     for (int i = 0; i < NUM_INPUTS; ++i) {

```

	Input	Expected	Got	
✓	1.5	0.82 Correct answer! Your code is faster at least 30%!	0.82 Correct answer! Your code is faster at least 30%!	✓
✓	2.15	0.90 Correct answer! Your code is faster at least 30%!	0.90 Correct answer! Your code is faster at least 30%!	✓

Passed all tests! ✓

**Bài tập 10 (bonus):** Tính tích hai ma trận vuông Dưới đây cung cấp đoạn code đơn giản để tính tích của hai ma trận cỡ  $N \times N \times N$  theo công thức trực tiếp. Hãy viết hàm tính tích hai ma trận nhưng có tốc độ nhanh hơn ít nhất 10% so với code đơn giản. Gợi ý: hãy để ý đến thứ tự truy cập các phần tử trong ma trận, tối ưu cache hoặc sử dụng thuật toán tốt hơn  $O(N^3)$ .

```

#include <iostream>
#include <ctime>
#include <cstdlib>
using namespace std;

const int N = 128;

struct Matrix {
    unsigned int mat[N][N];
};

bool operator == (const Matrix &a, const Matrix &b) {
    for (int i = 0; i < N; ++i) {
        for (int j = 0; j < N; ++j) {
            if (a.mat[i][j] != b.mat[i][j]) return false;

```



```

    }
}
return true;
}

Matrix multiply_naive(const Matrix &a, const Matrix &b) {
    Matrix c;
    for (int i = 0; i < N; ++i) {
        for (int j = 0; j < N; ++j) {
            for (int k = 0; k < N; ++k) {
                c.mat[i][j] += a.mat[i][k] * b.mat[k][j];
            }
        }
    }
    return c;
}

Matrix multiply_fast(const Matrix &a, const Matrix &b) {
    Matrix c;
    for (int i = 0; i < N; ++i) {
        for (int j = 0; j < N; ++j) {
            c.mat[i][j] = 0;
            for (int k = 0; k < N; ++k) {
                c.mat[i][j] += a.mat[i][k] * b.mat[k][j];
            }
        }
    }
    return c;
}

Matrix gen_random_matrix() {
    Matrix a;
    for (int i = 0; i < N; ++i) {
        for (int j = 0; j < N; ++j) {
            a.mat[i][j] = rand();
        }
    }
    return a;
}

Matrix base;

double benchmark(Matrix (*multiply) (const Matrix&, const Matrix&), Matrix
&result) {
    const int NUM_TEST = 10;
    const int NUM_ITER = 64;
    Matrix a = base;
    result = a;
    double taken = 0;

```

```

    for (int t = 0; t < NUM_TEST; ++t) {
        clock_t start = clock();
        for (int i = 0; i < NUM_ITER; ++i) {
            a = multiply(a, result);
            result = multiply(result, a);
        }
        clock_t finish = clock();
        taken += (double)(finish - start);
    }
    taken /= NUM_TEST;
    printf("Time: %.9f\n", taken / CLOCKS_PER_SEC);
    return taken;
}

int main() {
    base = gen_random_matrix();
    Matrix a, b;
    printf("Slow version\n");
    double slow = benchmark(multiply_naive, a);
    printf("Fast version\n");
    double fast = benchmark(multiply_fast, b);
    if (a == b) {
        printf("Correct answer! Your code is %.2f%% faster\n", slow / fast *
100.0);
    } else {
        printf("Wrong answer!\n");
    }
    return 0;
}

//sok sokong 20211005

```

The screenshot shows a C++ IDE with the following components:

- Project Explorer:** Shows a project named "IT3040\_Week2" with a sub-folder "Week2" containing files Bai2.1.cpp through Bai2.9.cpp, and Bai10.cpp (selected).
- Source Editor:** Displays the code for Bai10.cpp, which includes the benchmarking logic and the main function. The code is identical to the one shown in the previous block.
- Run Console:** Shows the output of the program:
 

```

"C:\IT3040_Week2\cmake-build-debug\new_target8.exe"
Slow version
Time: 0.761300000
Fast version
Time: 0.750200000
Wrong answer!
Process finished with exit code 0
      
```
- Status Bar:** Shows the current file is Bai10.cpp, with a timestamp of 92:23 and various encoding and formatting settings.

**Phần 3. Bài tập về nhà**

**Bài tập 11:** Tính tích hai đa thức Cho 2 đa thức  $A(x)$  và  $B(x)$  tương ứng có bậc  $NN$  và  $MM$ . Hãy tính ma trận tích  $C(x) = A(x) * B(x)$  có bậc  $N+M-1$ . Input: Gồm 2 dòng biểu diễn các đa thức  $A(x)$  và  $B(x)$ , mỗi dòng • Số đầu tiên  $NN$  là bậc của đa thức; •  $N+1$  số nguyên tiếp theo, số thứ  $i$  là hệ số của  $x^{i-1}$ . Output: Một số nguyên duy nhất là XOR của các hệ số của đa thức  $C(x)$ . Ví dụ:

Input: 3 83 86 77 15 4 93 35 86 92 49

Output: 20731

Giải thích: các hệ số của đa thức kết quả lần lượt là 7719, 10903, 17309, 19122, 19126, 12588, 5153, 735.

Giới hạn: • Các hệ số của các đa thức đầu vào có trị tuyệt đối nhỏ hơn 100. • Có 5 tests, test thứ  $i$  có bậc của các đa thức đầu vào không quá  $10i$ .

```
// Sok Sokong 20211005
#include <iostream>
#include <vector>
using namespace std;

int main() {
    int N, M;
    cin >> N;
    vector<int> A(N + 1);
    for (int i = 0; i <= N; i++) {
        cin >> A[i];
    }

    cin >> M;
    vector<int> B(M + 1);
    for (int i = 0; i <= M; i++) {
        cin >> B[i];
    }

    int C_size = N + M + 1;
    vector<int> C(C_size, 0);

    for (int i = 0; i <= N; i++) {
        for (int j = 0; j <= M; j++) {
            C[i + j] += A[i] * B[j];
        }
    }

    int result = 0;
    for (int i = 0; i < C_size; i++) {
        result ^= C[i];
    }

    cout << result << endl;
```

```

return 0;
}

// Sok Sokong 20211005
#include <iostream>
#include <vector>
using namespace std;

int main() {
    int N, M;
    cin >> N;
    vector<int> A(N + 1);
    for (int i = 0; i <= N; i++) {
        cin >> A[i];
    }

    cin >> M;
    vector<int> B(M + 1);
    for (int i = 0; i <= M; i++) {
        cin >> B[i];
    }
}

```

Run: "C:\IT3040 Week2\Bai11.exe"

```

3 83 86 77 15
4 93 35 86 92 49
20731

```

Process finished with exit code 0

**Bài tập 12:** Map Sort Hôm nay, cô giáo giao cho An một câu hỏi học bữa. Cô cho một danh sách với mỗi phần tử có dạng và yêu cầu An sắp xếp danh sách đó giảm dần theo giá trị value. Nếu 2 phần tử có value giống nhau thì sắp xếp giảm dần theo key. Hãy viết một chương trình sử dụng hàm nặc danh để giúp An làm bài tập. Input: Danh sách đầu vào. Mỗi dòng ghi một cặp giá trị key, value cách nhau bởi dấu cách ( $|key| \leq 109 \leq 109$ ,  $|value| \leq 109 \leq 109$ ).

```

// Sok Sokong 20211005
#include <bits/stdc++.h>
using namespace std;
struct element {
    int key;
    int value;
};

vector<element> lst;
void input() {
    int tmp1, tmp2;
    while (cin >> tmp1 && cin >> tmp2) {
        element tmp;
        tmp.key = tmp1;
        tmp.value = tmp2;

        lst.push_back(tmp);
    }
}

void print() {
    for (int i = 0; i < lst.size(); i++) {

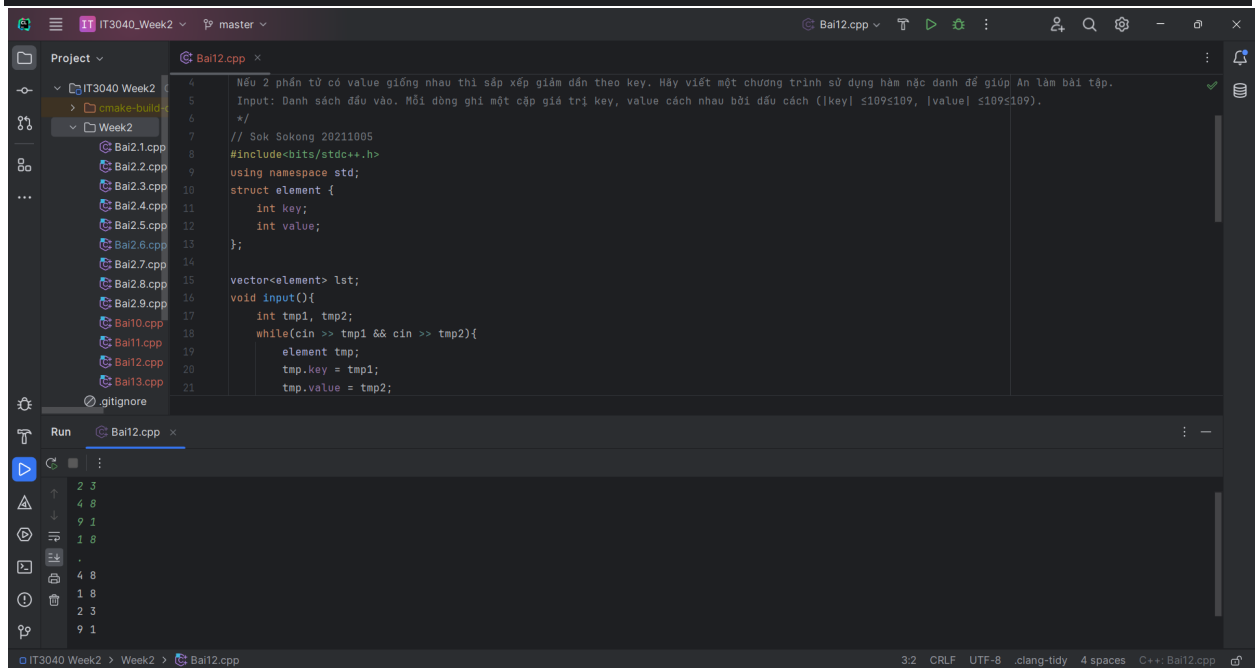
```

```

        cout << lst[i].key << " " << lst[i].value << endl;
    }
}

int main(){
    input();
    sort(lst.begin(),lst.end(),[] (element a, element b){
        if(a.value > b.value) return true;
        else if (a.value < b.value) return false;
        else {
            return a.key >= b.key;
        }
    });
    print();
}

```



### Bài tập 13: Big Integer

Số nguyên lớn là các số nguyên có giá trị rất lớn và không thể biểu diễn bằng các kiểu dữ liệu nguyên cơ bản. Để biểu diễn số nguyên lớn, ta có thể dùng kiểu struct như sau: struct bigNum{ char sign; char num[101]; };

Nhiệm vụ các bạn là đa năng hóa các toán tử để thực hiện các phép toán số học với kiểu dữ liệu số nguyên lớn vừa định nghĩa ở trên.

```

// Sok Sokong 20211005
#include<bits/stdc++.h>
using namespace std;
struct bigNum {
    char sign;
    char num[101];
};

```

```

// input and preprocess data
void input(bigNum &num1, bigNum &num2){
    string tmp;
    cin >> tmp;
    num1.sign = tmp[0];
    int lens1 = tmp.length() - 1;
    for(int i=0; i<lens1; i++){
        num1.num[100-lens1+i+1] = tmp[i+1];
    }
    for(int i=0; i<100-lens1+1; i++) num1.num[i] = '0';

    cin >> tmp;
    num2.sign = tmp[0];
    int lens2 = tmp.length() - 1;
    for(int i=0; i<lens2; i++){
        num2.num[100-lens2+i+1] = tmp[i+1];
    }
    for(int i=0; i<100-lens2+1; i++) num2.num[i] = '0';
}

// add 2 positive big number
void add(char res[], char *num1, char *num2){
    int c = 0;

    for(int i=100; i>=0; i--){
        int tmp = (int)num1[i] - 48 + (int)num2[i] - 48 + c;
        c = tmp / 10;
        res[i] = tmp % 10 + 48;
    }
}

// sub 2 positive big number, num1 > num2
void sub(char res[], char *num1, char* num2){
    int c = 0;

    for(int i=100; i>=0; i--){
        int tmp1 = (int)num1[i] - 48;
        int tmp2 = (int)num2[i] - 48;

        if(tmp1 >= tmp2 + c){
            res[i] = tmp1 - tmp2 - c + 48;
            c = 0;
        } else {
            tmp1 = tmp1 + 10;
            res[i] = tmp1 - tmp2 - c + 48;
            c = 1;
        }
    }
}

```

```

}

// multi 2 positive big number
void multi(char res[], char *num1, char *num2){
    // clear array res
    for(int i=0; i<101; i++) res[i] = '0';

    for(int i=100; i>=0; i--){
        // init 1 array temp
        char tmp[101];

        // add i number 0 to last array
        int k;
        for(k = 0; k < i; k++)
            tmp[100-k] = '0';

        int c = 0, sum = 0;
        for(int j=100; j>=0; j--){
            sum = ((int)num1[i] - 48) * ((int)num2[j] - 48) + c;
            tmp[k] = (sum % 10) + 48;
            c = sum / 10;
            k--; if(k < 0) break;
        }

        add(res,tmp,res);
    }
}

// check number1 >= number2
bool check(char *num1, char *num2){
    int foo1, foo2;
    for(foo1 = 0; foo1 < 101; foo1++){
        if(num1[foo1] != '0') break;
    }

    for(foo2 = 0; foo2 < 101; foo2++){
        if(num2[foo2] != '0') break;
    }

    if(foo1 > foo2) return false;
    else if(foo1 < foo2) return true;
    else { // foo1 == foo2
        int foo = foo1;
        while(foo < 101){
            if(num1[foo] < num2[foo]) return false;
            else if (num1[foo] > num2[foo]) return true;
            else {
                foo++;
            }
        }
    }
}

```

```

    }
}

return true;
}

// overloading operator "+"
bigNum operator + (bigNum num1, bigNum num2){
    bigNum res;

    if(num1.sign == '1' && num2.sign == '1'){
        res.sign = '1';
        add(res.num, num1.num, num2.num);
        return res;
    } else if(num1.sign == '1' && num2.sign == '0'){
        if(check(num1.num, num2.num)) {
            res.sign = '1';
            sub(res.num, num1.num, num2.num);
            return res;
        } else {
            res.sign = '0';
            sub(res.num, num2.num, num1.num);
            return res;
        }
    } else if(num1.sign == '0' && num2.sign == '1'){
        if(check(num1.num, num2.num)) {
            res.sign = '0';
            sub(res.num, num1.num, num2.num);
            return res;
        } else {
            res.sign = '1';
            sub(res.num, num2.num, num1.num);
            return res;
        }
    } else {
        res.sign = '0';
        add(res.num, num1.num, num2.num);
        return res;
    }
}

bigNum operator - (bigNum num1, bigNum num2){
    bigNum res;

    if(num1.sign == '1' && num2.sign == '0'){
        num2.sign = '1';
        res = num1 + num2;
        return res;
    } else if(num1.sign == '1' && num2.sign == '1'){

```



```

        num2.sign = '0';
        res = num1 + num2;
        return res;
    } else if(num1.sign == '0' && num2.sign == '1'){
        num2.sign = '0';
        res = num1 + num2;
        return res;
    } else {
        num2.sign = '1';
        res = num1 + num2;
        return res;
    }
}

bigNum operator * (bigNum num1, bigNum num2){
    bigNum res;

    if(num1.sign == '1' && num2.sign == '1'){
        res.sign = '1';
        multi(res.num, num1.num, num2.num);
        return res;
    } else if(num1.sign == '1' && num2.sign == '0'){
        res.sign = '0';
        multi(res.num, num1.num, num2.num);
        return res;
    } else if(num1.sign == '0' && num2.sign == '1'){
        res.sign = '0';
        multi(res.num, num1.num, num2.num);
        return res;
    } else {
        res.sign = '1';
        multi(res.num, num1.num, num2.num);
        return res;
    }
}

// print bignumber
void printBigNumber(bigNum number){
    cout << number.sign;
    int start;
    for(start=0; start<101; start++)
        if(number.num[start] != '0') break;

    for(int i = start; i<101; i++)
        cout << number.num[i];
}

int main(){
    bigNum num1, num2;

```

```

input(num1,num2);

bigNum so3, so4;
so3.sign = '1', so4.sign = '1';
for(int i=0; i<100; i++){
    so3.num[i] = '0';
    so4.num[i] = '0';
}
so3.num[100] = 3 + 48;
so4.num[100] = 4 + 48;

bigNum res = num1*num2 - so3 * num1 + so4 * num2;

printBigNumber(res);
}

```

Output: In ra giá trị của biểu thức  $ab-3a+4bab-3a+4b$ . Kết quả in ra một số nguyên lớn dưới dạng chuỗi ký tự có định dạng như mô tả trong dữ liệu vào.

```

14 // Sok Sokong 20211005
15 // Sok Sokong 20211005
16 // Sok Sokong 20211005
17 // Sok Sokong 20211005
18 #include<bits/stdc++.h>
19 using namespace std;
20 struct bigNum {
21     char sign;
22     char num[101];
23 };
24
25 // input and preprocess data
26 void input(bigNum &num1, bigNum &num2){
27     string tmp;
28     cin >> tmp;
29     num1.sign = tmp[0];
30     int lens1 = tmp.length() - 1;
31     for(int i=0; i<lens1; i++){

```

Run: Bai13.cpp x

"C:\IT3040 Week2\Bai13.exe"

```

0121807015
1347227347
042294724910108772
Process finished with exit code 0

```

IT3040 Week2 > Week2 > Bai13.cpp 2342 CRLF UTF-8 clang-tidy 4 spaces C++ Bai13.cpp