DS_OOP Online Test(2017/05/08)
18:30~21:30 EC315, EC316

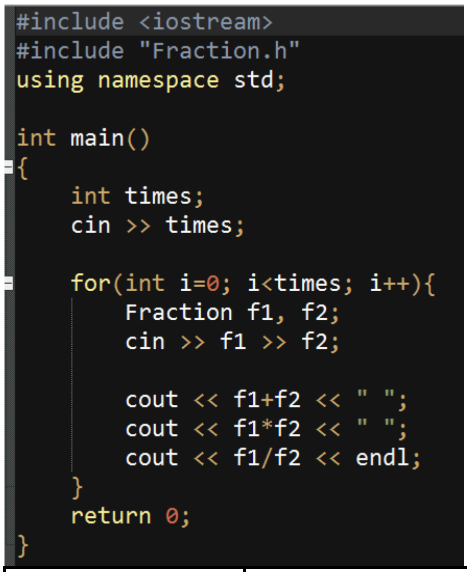
1.(15%)

Compute the fraction operations with operator overloading. (+ , * , / , >> , <<)

First line is the number of testing data.
From the second line, each line contains 4 numbers (A, B, C, D), and each number is separated by space.
These 4 numbers, A B C D, represent 2 fractions (A/B) and (C/D).
A, B, C, D are in the range of (0~2^32-1), but B and D will not be zero.

Please compute (a/b)+(c/d), (a/b)*(c/d), and (a/b)/ (c/d).

Please print the fraction in lowest term.
The output format of a fraction is numerator/denominator,
but if the denominator of the answer is 1 or numerator of the answer is 0, please do not print out the denominator.
TAs will use this main to run your code, you don't need to have main function in your code



```
#include <iostream>
#include "Fraction.h"
using namespace std;

int main()
{
    int times;
    cin >> times;

    for(int i=0; i<times; i++){
        Fraction f1, f2;
        cin >> f1 >> f2;

        cout << f1+f2 << " ";
        cout << f1*f2 << " ";
        cout << f1/f2 << endl;
    }
    return 0;
}
```

sample input:	3
	3 2 2 3
	0 2 1 2
	4 2 12 3

sample output:	13/6 1 9/4 1/2 0 0 6 8 1/2

2.(15%)

Given a series of big numbers, and for each big number x is in the range: 0 <= x <= 10^20000.

Please print "Computer" if x is multiples of 9,
print "Science" if x is mulltiples of 11,
print "ComputerScience" if x is multiples of 99,
and print "Bug" if otherwise.

Please use standard input and standard output to finish this question.

example:

input:	594 87 88 9
output:	ComputerScience Bug Science Computer

3.(20%)

First line is the input case number , which is less than 100.
Given a line of words, reverse the words position in the line.
Each word is separated by a space or more.
Each line will less than 1000000 characters.
The answer of each word is separated by a single space, each answer is in a line.

Please use standard input and standard output to finish this question.

example:

input:	3 You are smart long Time no seE TA’s ars some beautiful.
output:	smart are You seE no Time long beautiful. some ars TA’s

4.(20%)

Polynomial operation – please implement the following operator:

(a) Polynomial operator+() – Add 2 polynomials

(b) Polynomial operator-() – Subtract 2 polynomials

(c) Polynomial operator~() – Differential a polynomial
(d) istream & operator>>() – Read a polynomial from cin
(e) int get_value(int v) – compute the value of this polynomial with x-value v

Input – first line is number of testing data, and the following lines are consist of 2 polynomials and 2 integers

Each polynomials is ended with “!”

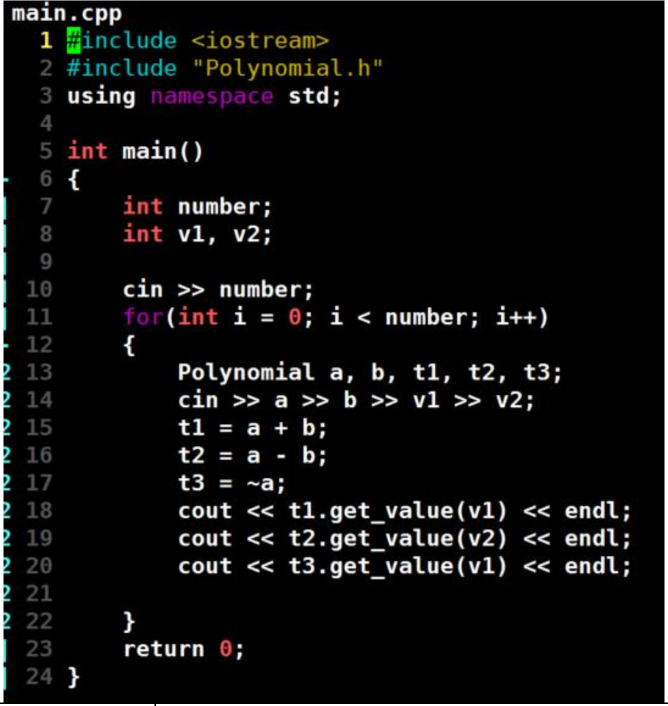
Each polynomial is represented as “coefficient degree coefficient degree...” and separated by space, eg, 1x^2+2x^1+1x^0 is represented as 1 2 2 1 1 0 !

The range of coefficient is -2^31 ~ 2^31

The range of degree is 0~20000, and the degree is in descending order

The number of terms will be less than 100

TA will use this main.cpp to test your code, please make sure your code can run with this main function.(Do not write any main function in your hand in code)



main.cpp

```cpp
#include <iostream>
#include "Polynomial.h"
using namespace std;

int main()
{
    int number;
    int v1, v2;

    cin >> number;
    for(int i = 0; i < number; i++)
    {
        Polynomial a, b, t1, t2, t3;
        cin >> a >> b >> v1 >> v2;
        t1 = a + b;
        t2 = a - b;
        t3 = ~a;
        cout << t1.get_value(v1) << endl;
        cout << t2.get_value(v2) << endl;
        cout << t3.get_value(v1) << endl;

    }
    return 0;
}
```

Input.txt	1 1 2 2 1 1 0 ! -1 2 1 1 ! 2 3
command	./a.out < input.txt
cout	7 22 6

5.(20%)

Implement a circular deque which have following function:

1. push_front() : Push an element to the front of the deque if the queue is NOT full, o.w., print "queue is full".
2. push_back() : Push an element to the end of the deque if the queue is NOT full, o.w.,

print "queue is full".

3. pop_front() : Pop an element from the front of the deque if the queue is NOT empty, o.w., print "queue is empty" and return 0.

4. pop_back() : Pop an element from the end of the deque if the queue is NOT empty, o.w., print "queue is empty" and return 0.

5. operator<< : Print the elements in queue from front to rear and separate each element by a space.

The class declaration is below:

```
#ifndef CircularDeque
#define CircularDeque
using namespace std;
class CirDeque{

        private:
                int data[10];
                int front, rear;

   public:
                CirDeque(){rear = front = 9;};
                void push_front(int);  // push an element to the front of the deque
                void push_back(int); // push an element to the end of the deque
                int pop_front();      // pop an element from the front of the deque
                int pop_back();       // pop an element from the end of the deque
                friend ostream& operator<<(ostream&, const CirDeque&);
                // print the elements in queue from front to rear and separate each element by
a space.
};

// you may write your code below this line

// you may write your code above this line
#endif
```

Please write your code in the file we gave.
You only need to update the .h file of the class definition.
We'll use our main function to test your code.
Please make sure your code is runnable. (we will run the command "g++ main.cpp)

example:

main.cpp

```
#include <iostream>
#include "CirDeque.h"  // change the header to  0XXXXXX_5.h ( use your id )
```

```
int main()
{
	CirDeque queue;
	queue.pop_front();
	queue.push_back(1);
	queue.push_back(2);
	queue.push_back(3);
	queue.push_back(4);
	queue.push_back(5);
	cout<<queue<<endl;
	queue.push_front(6);
	queue.push_front(7);
	queue.push_front(8);
	queue.push_front(9);
	queue.push_front(10);
	cout<<queue<<endl;
	queue.pop_back();
	cout<<queue<<endl;
	return 0;
}
```

output:

```
	queue is empty
	1 2 3 4 5
	queue is full
	9 8 7 6 1 2 3 4 5
	9 8 7 6 1 2 3 4
```

6.(10%)

Following the previous question, implement class template of circular deque which have the function in question 5.

The class declaration is below:

```
#ifndef CircularDequeTemplate
#define CircularDequeTemplate
using namespace std;

template<class T> class CirDequeTemplate;
template<class T> std::ostream& operator<< (std::ostream& o, const
CirDequeTemplate<T>& q);


template <class T>
class CirDequeTemplate{

            private:
                        T data[10];
                        int front, rear;

    public:
                        CirDequeTemplate(){rear = front = 9;};
                        void push_front(T);  // push an element to the front of the deque
                        void push_back(T);   // push an element to the end of the deque
                        T pop_front();       // pop an element from the front of the deque
                        T pop_back();        // pop an element from the end of the deque
                        friend ostream& operator<< <>(ostream&, const CirDequeTemplate&);
                        // print the elements in queue from front to rear and separate each element by
a space.
};
// you may write your code below this line

// you may write your code above this line
#endif
```

Please write your code in the file we gave.
You only need to update the .h file of the class definition.
We'll use our main function to test your code.
Please make sure your code is runnable.

Submission:

1. submit <studentID_questionNO.>.h, <studentID_questionNO.>.cpp if you have any
   ex: 0516000_1.h, 0516000_2.h, ………
2. upload to e3, we will open it at 9:10~9:20, upload the corresponding file to each question, and you won’t compress it.