1.

（1）数值传递

#include <iostream>

#include <stdio.h>

using namespace std;

int fun(int a, int b)

{

int c;

c = (a % 10) \* 1000 + (b % 10) \* 100 + (a / 10) \* 10 + b / 10;

return c;

}

int main()

{

int a,b ;

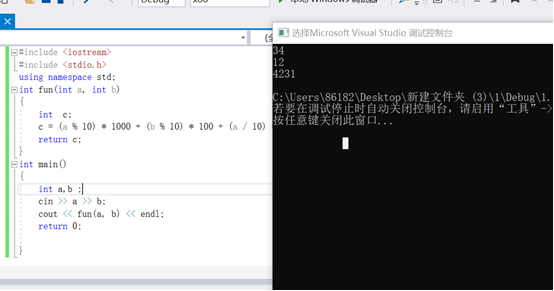
cin >> a >> b;

cout << fun(a, b) << endl;

return 0;

}

运行截图：



1. 引用作为函数参数

#include <iostream>

#include <stdio.h>

using namespace std;

int fun(int &a, int &b)

{

int c;

c = (a % 10) \* 1000 + (b % 10) \* 100 + (a / 10) \* 10 + b / 10;

return c;

}

int main()

{

int x,y;

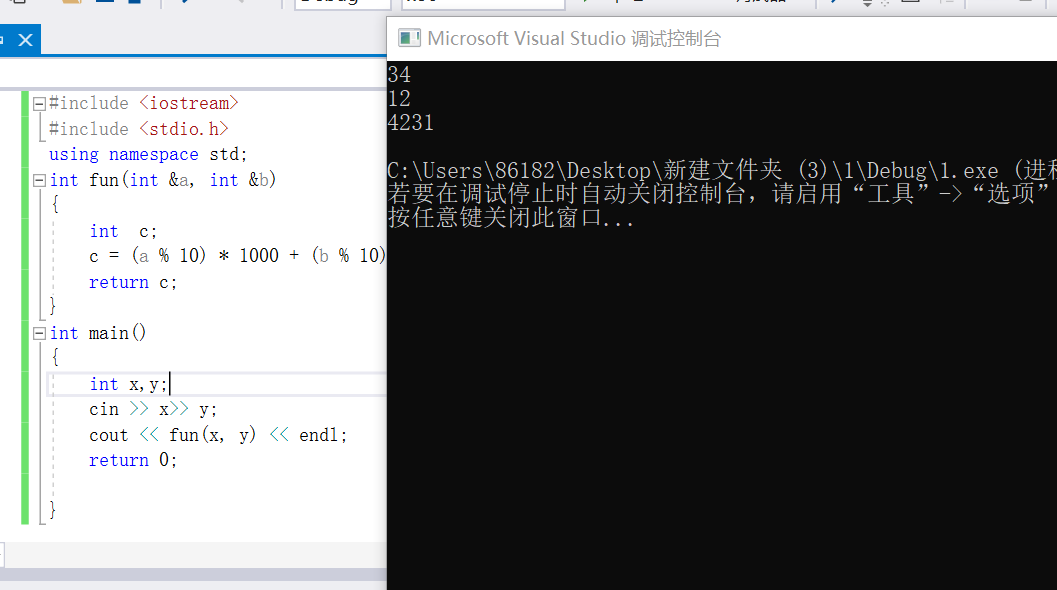
cin >> x>> y;

cout << fun(x, y) << endl;

return 0;

}

运行截图：



（3）指针传址

#include <iostream>

#include <stdio.h>

using namespace std;

int fun(int \*a, int \*b)

{

int c;

c = (\*a % 10) \* 1000 + (\*b % 10) \* 100 + (\*a / 10) \* 10 + \*b / 10;

return c;

}

int main()

{

int a, b;

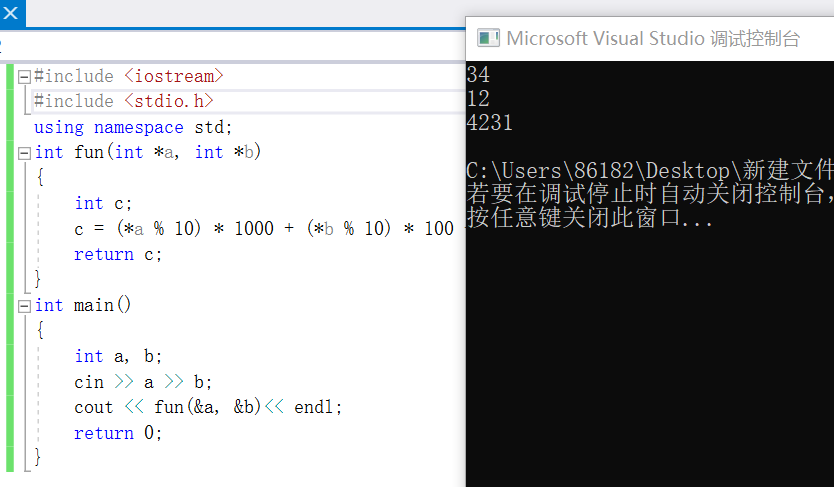
cin >> a >> b;

cout << fun(&a, &b)<< endl;

return 0;

}

运行截图：



2.

#include <iostream>

using namespace std;

struct Monkey

{

int num;

struct Monkey \*next;

};

int main()

{

int m, n, i, j, king;

Monkey \*head, \*p1, \*p2;

cin >> m >> n;

if (n == 1)

{

king = m;

}

else

{

p1 = p2 = new Monkey;

head = p1;

p1->num = 1;

for (i = 1; i < m; i++)

{

p1 = new Monkey;

p1->num = i + 1;

p2->next = p1;

p2 = p1;

}

p2->next = head;

p1 = head;

for (i = 1; i < m; i++)

{

for (j = 1; j < n - 1; j++)

p1 = p1->next;

p2 = p1->next;

p1->next = p2->next; ”

p1 = p2->next;

delete p2;

}

king = p1->num;

delete p1;

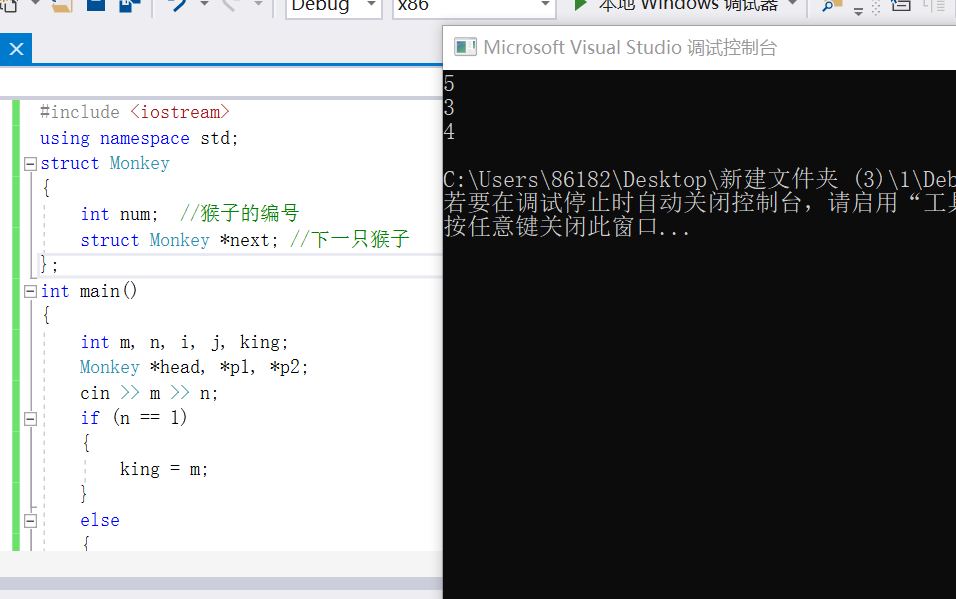
}

cout << king << endl;

return 0;

}

运行截图：



（2）数组解法：利用数组int n[M]，都初始化为1，淘汰标记为0，循环操作，剩下最后一个数组元素1的下标+1即为所求答案

#include <iostream>

using namespace std;

int whoIsMonkeyKing(int, int);

int main()

{

int m, n;

cout << whoIsMonkeyKing(5,3) << endl;

return 0;

}

int whoIsMonkeyKing(int m, int n)

{

if (m < 1 || n < 1)

{

cout << "输入参数错误" << endl;

return -1;

}

int \*p = new int[m];

int \*q = p, M = m;

int res;

for (int i = 0; i < m; i++)

{

p[i] = 1;

}

while (M != 1)

{

int i = 0;

while (i != n)

{

if (q == p + m)

{

q = p;

}

if (\*q++ == 1)

{

++i;

}

}

\*(q - 1) = 0;

--M;

}

for (int i = 0; i < m; i++)

{

if (\*(p + i) == 1)

{

res = i + 1;

break;

}

else

continue;

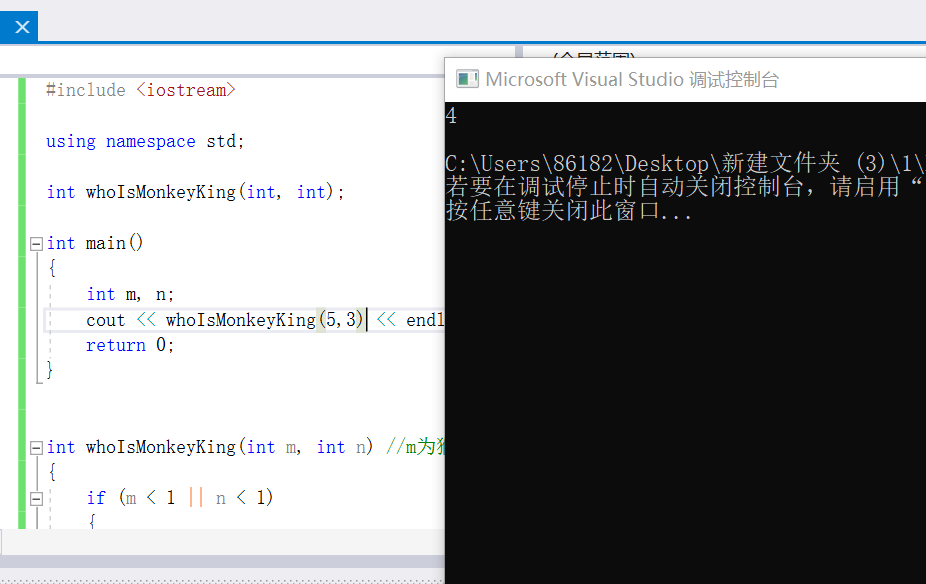
}

delete[] p;

return res;

}

运行截图：



3.

#include<iostream>

#include <algorithm>

using namespace std;

struct student

{

int ch, math, eng, sum, num;

}stu[10];

bool cmp(student stu1, student stu2)

{

if (stu1.sum != stu2.sum)

return stu1.sum > stu2.sum;

else if (stu1.ch != stu2.ch)

return stu1.ch > stu2.ch;

else return stu1.num < stu2.num;

}

int main() {

int i, n;

bool first = true;

while (cin >> n)

{

for (i = 0; i < n; i++) {

stu[i].num = i + 1;

cin >> stu[i].ch >> stu[i].math >> stu[i].eng;

stu[i].sum = stu[i].ch + stu[i].math + stu[i].eng;

}

if (first) first = false;

else cout << endl;

sort(stu, stu + n, cmp);

for (i = 0; i < 5; i++)

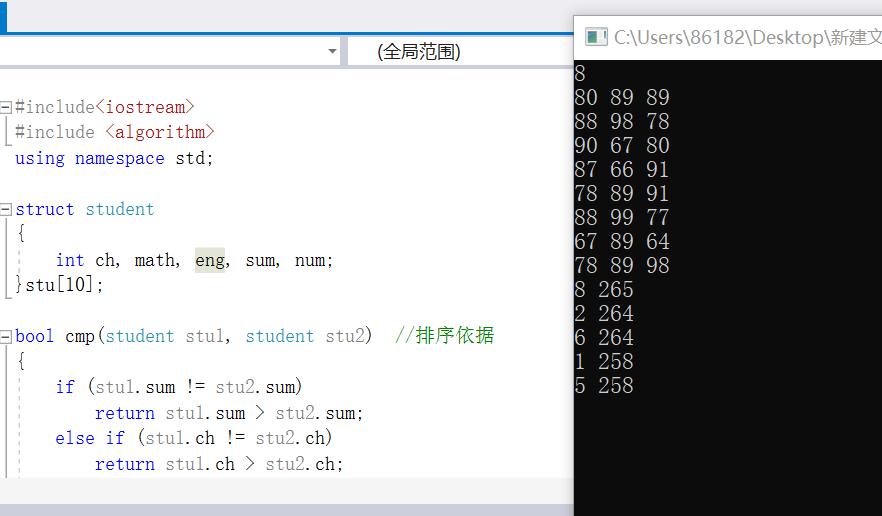
cout << stu[i].num <<" " << stu[i].sum << endl;

}

return 0;

}

运行截图：



4.

#include <stdio.h>

#include <iostream>

using namespace std;

#define max 100

int main()

{

int M, N, b;

int i, j, m, n, d = 0;

char str[max][max] = { 0 };

cout << "输入地图的行数和列数：";

cin >> M >> N;

printf("输入感染信号\n");

for (i = 0; i < N; i++)

{

for (j = 0; j < M; j++)

{

cin >> str[i][j];

}

}

cout << "输入周期数：";

cin >> b;

do

{

for (i = 0; i < N; i++)

{

for (j = 0; j < M; j++)

{

if (str[i][j] == 'X')

{

{ if (str[i - 1][j] = 'O')

str[i - 1][j] = 'm';

if (str[i + 1][j] = 'O')

str[i + 1][j] = 'm';

if (str[i][j + 1] = 'O')

str[i][j + 1] = 'm';

if (str[i][j - 1] = 'O')

str[i][j - 1] = 'm';

}

}

}

}

d++;

for (m = 0; m < N; m++)

{

for (n = 0; n < M; n++)

{

if (str[m][n] == 'm')

str[m][n] = 'X';

}

}

} while (d < b);

for (i = 0; i < N; i++)

{

for (j = 0; j < M; j++)

{

cout << str[i][j];

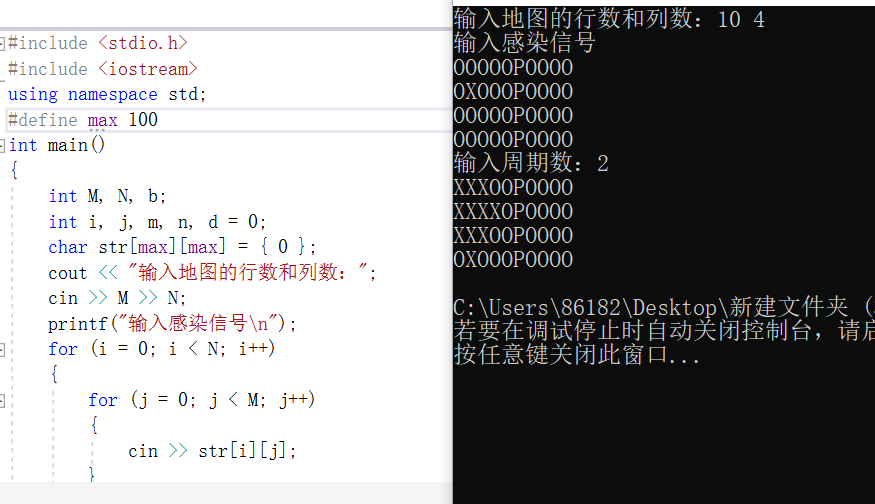
}

cout << endl;

}

}

运行截图：



5.

#include<iostream>

#include<string>

using namespace std;

template<typename T>

class Node

{ public:

T element;

Node \*next;

Node(T element) {

this->element = element;

next = NULL;

}

};

template<typename T>

class Stack

{

private:

Node<T> \*top;

int size;

public:

Stack() {

Node<T> \*newNode = new Node<T>(0);

top = newNode;

size = 0;

}

int stackLength() {

return size;

}

bool stackEmpty() {

if (0 == size)

return true;

else

return false;

}

void Push(T e) {

Node<T> \*newNode = new Node<T>(e);

newNode->next = top->next;

top->next = newNode;

size++;

}

T Pop() {

Node<T> \*current = top->next;

top->next = current->next;

T f = current->element;

delete current;

size--;

return f;

}

T getTop() {

return top->next->element;

}

};

char Priority(char ch1, char ch2) {

int a;

int b;

switch (ch1) {

case '=': a = 0;

break;

case '(': a = 1;

break;

case '+': a = 3;

break;

case '-': a = 3;

break;

case '\*': a = 5;

break;

case '/': a = 5;

break;

case '%': a = 5;

break;

case '^': a = 7;

break;

case ')': a = 8;

break;

}

switch (ch2) {

case '=': b = 0;

break;

case '(': b = 8;

break;

case '+': b = 2;

break;

case '-': b = 2;

break;

case '\*': b = 4;

break;

case '/': b = 4;

break;

case '%': b = 4;

break;

case '^': b = 6;

break;

case ')': b = 1;

break;

}

if (a < b)

return '<';

else if (a == b)

return '=';

else

return '>';

}

int Compute(int a, int b, char sign) {

int result;

switch (sign) {

case '+': result = a + b;

break;

case '-': result = a - b;

break;

case '\*': result = a \* b;

break;

case '/': result = a / b;

break;

case '%': result = a % b;

break;

case '^': result = a ^ b;

break;

}

return result;

}

int main()

{

Stack<int> number;

Stack<char> character;

character.Push('=');

cout << "输入例：1+(2\*3+4)={ }" << endl;

//string k="22";

//k[0]=1;

//k[0]=55;

//cout<<(k[0]-0);

string s;

getline(cin, s);

string k = "000000000000000000000000000000000000";

int sum = 0;

int i = 0;

int j = 0;

while (s[i] != '=')

{

if (s[i] >= 48 && s[i] <= 57)

{

//cout<<(s[i]-48);

sum = sum \* 10 + (s[i] - 48);

//cout<<sum;

k[j] = sum;

//cout<<(k[j]-0);

i++;

// cout<<i;

//i++;

if (s[i] < 48 || s[i]>57)

j++;

}

else

{

// k[]

//s[j]=s[i];

sum = 0;

k[j] = s[i];

j++;

i++;

//j++;

}

}

k[j] = '=';

// cout<<(k[0]-0);

//cout<<k[1];

//cout<<(k[2]-0);

//cout<<k[3];

int kk = 0;

int ch = k[kk++];

//cout<<ch;

char b;

b = static\_cast<char>(ch);

//cout<<(b-0);

while (b != '=' || character.getTop() != '=') {

if (ch == '+' || ch == '=' || ch == '-' || ch == '\*' || ch == '/' || ch == '(' || ch == ')') {

switch (Priority(character.getTop(), b)) {

case'<':

character.Push(b);

ch = k[kk++];

b = static\_cast<char>(ch);

break;

case'=':

character.Pop();

//character.Push(b);

ch = k[kk++];

b = static\_cast<char>(ch);

break;

case'>':

number.Push(Compute(number.Pop(), number.Pop(), character.Pop()));

break;

}

}

else {

number.Push(ch);

ch = k[kk++];

b = static\_cast<char>(ch);

}

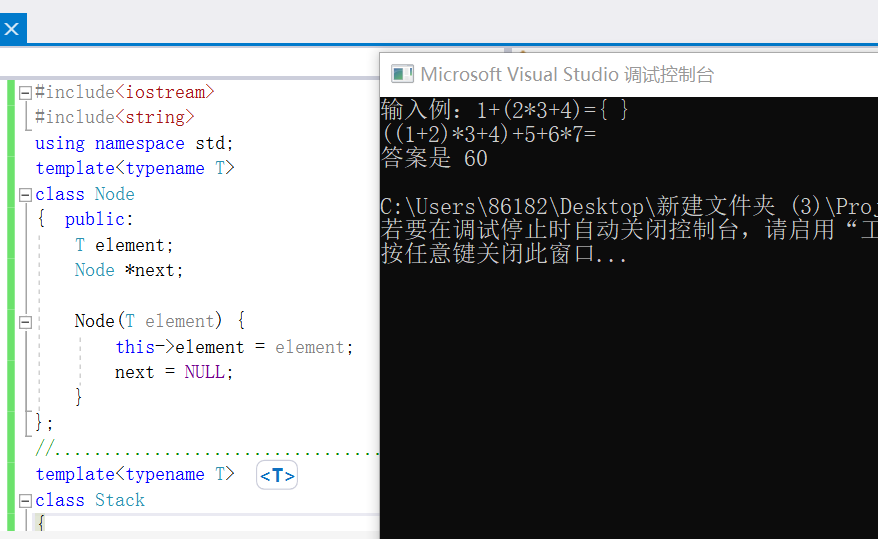
}

cout << "答案是" << " " << number.getTop() << '\n';

return 0;

}

运行截图：



6.

#include<iostream>

using namespace std;

class vehicle

{

protected:

int wheels;

int weight;

public:

vehicle(int w1, int w2)

{

wheels = w1;

weight = w2;

}

void disp()

{

cout << "wheels=" << wheels << ",weight=" << weight << endl;

}

};

class car :private vehicle

{

int passenger\_load;

public:

car(int w1, int w2, int pa) :vehicle(w1, w2)

{

passenger\_load = pa;

}

void disp()

{

vehicle::disp();

cout << "passenger\_load=" << passenger\_load << endl;

}

};

class truck :private vehicle

{

int passenger\_load;

int payload;

public:

truck(int w1, int w2, int pa, int pay) :vehicle(w1, w2)

{

passenger\_load = pa;

payload = pay;

}

void disp()

{

vehicle::disp();

cout << "passenger\_load=" << passenger\_load << endl;

cout<< "payload=" << payload << endl;

}

};

void main()

{

vehicle v(200, 180);

v.disp();

car c(100, 80, 60);

c.disp();

truck t(50, 35, 30, 15);

t.disp();

system("pause");

}

运行截图：

