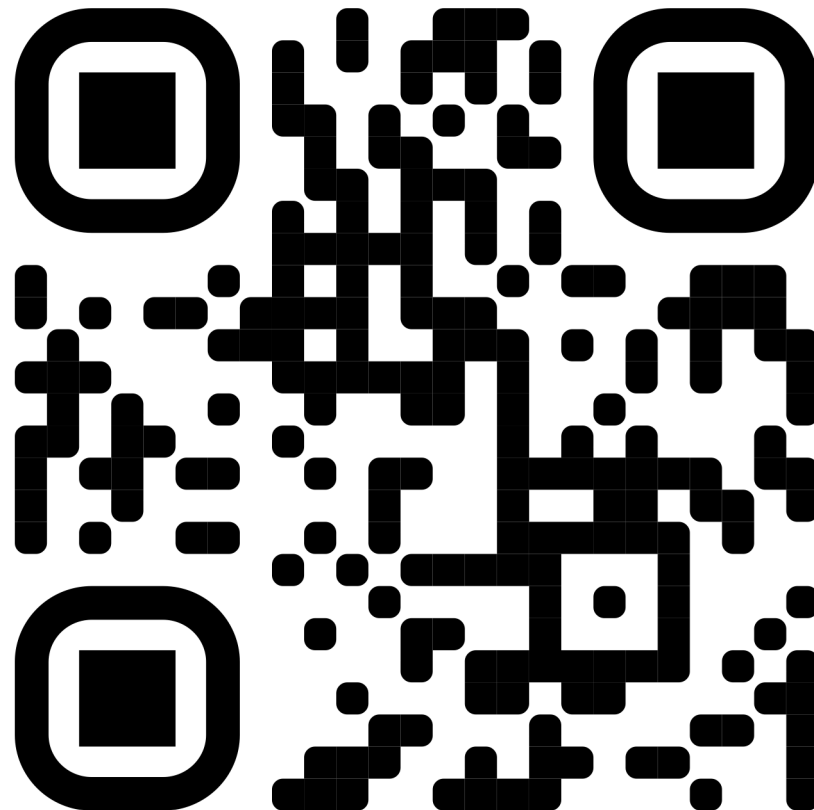


Абстракции -2

u.to/72D_Gg

Лекция 5, 05 марта, 2021



Лектор:

Дмитрий Северов, кафедра информатики 608 КПМ

dseverov@mail.mipt.ru

Обратная связь : **u.to/7Wn7Gg**

Класс

```
class complex { float Re,Im;
public:
complex() { Re = Im = 0; }
complex(float R, float I) { Re=R; Im=I; }
float abs() { return sqrt(Re*Re + Im*Im); }
complex operator+(complex a) {
    return complex(Re+a.Re, Im+a.Im); }

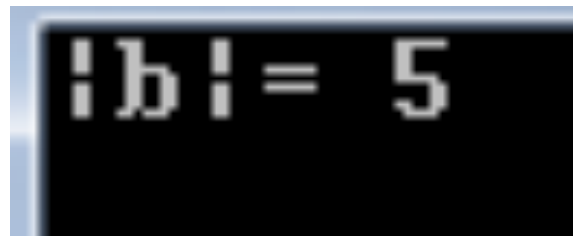
friend ostream& operator<<(
    ostream& a, const complex& b) {
    return a << '(' << b.Re << ',' << b.Im << ')'
    << endl; }
};
```

Шаблон класса

```
template<typename T> class complex {  
    T Re, Im;  
public:  
    complex(T R=0, T I=0) { Re=R; Im=I; }  
    T abs() { return sqrt(Re*Re + Im*Im); }  
    complex operator+(complex a) {  
        return complex(Re+a.Re, Im+a.Im); }  
  
    friend ostream& operator<<(ostream& a, const  
        complex& b) {  
        return a << '(' << b.Re << ',' << b.Im <<  
            ')'; }  
};
```

Использование шаблона класса

```
#include <iostream>
using namespace std;
#include "complex.h"
int main() {
    complex<double> a(1.,2.),b,c(2.,2.);
    b = a + c;
    cout << "|b|= " << b.abs() << endl;
    return 0; }
```



|b|= 5

Шаблон узла набора данных

```
template <typename T> class Item {  
    T node; Item* next;  
public:  
    Item(const T& elem, Item* n=0) {  
        node=elem; next=n;}  
    T& get_node() { return node; }  
    Item* &get_next() { return next; }  
};
```

Шаблон класса Stack

```
template <typename T> class Stack {  
    Item<T> *top; T rab;  
public:  
    Stack() { top = 0; }  
    void push(const T& elem) {  
        top=new Item<T>(elem,top); }  
    T& pop() {  
        if(!top) ERR("Stack::pop: empty stack");  
        rab = top->get_node(); Item<T> *p=top;  
        top = p->get_next(); delete p; return rab; }  
    bool empty() {return top == 0; }  
};
```

Шаблон класса Queue (1)

```
template <typename T> class Queue {  
    Item<T> *head,*tail; T rab;  
public:  
  
    Queue() { tail = head =0; }  
  
    bool empty() { return head == 0;}  
  
    void put(const T& elem) {  
        if(tail==0) tail = head = new Item<T>(elem);  
        else tail = (tail->get_next() =  
            new Item<T>(elem));}
```

Шаблон класса Queue (2)

```
T& get() {  
    if(!head) ERR("Queue::get: queue is empty");  
    Item<T> *p=head;  
    rab = head->get_node();  
    head=head->get_next();  
    delete p;  
    if(head==0) tail=0;  
    return rab;  
}  
}; // template <typename T> class Queue
```


Шаблон класса List (1)

```
template <typename T> class List {  
  
    Item<T> *front,*back; T rab;  
  
    Item<T>* find(Item<T>* &F, const T& k) {  
        if(front==NULL) return (F=NULL);  
        Item<T> *ptr=F=front;  
        if(front->get_node()==k) return 0;  
        while((F=ptr->get_next())!=NULL) {  
            if(F->get_node()==k) break; ptr=F; }  
        return ptr;  
    }  
}
```

Шаблон класса List (2)

public:

```
List() { front = back = 0; }
```

```
bool empty() { return front==0; }
```

```
void push_back(const T& elem) {  
    if(back==0) front = back = new Item<T>(elem);  
    else back = (back->get_next() = new  
Item<T>(elem)); }
```

Шаблон класса List (3)

```
T& pop_back() {  
    if(back==0)  
        ERR("List::pop_back: list is empty");  
    rab=back->get_node();  
    Item<T> *p=front;  
    if(front==back) front = back = 0;  
    else {  
        while(p->get_next()!=back) p=p->get_next();  
        back=p;  
        p=p->get_next();  
        back->get_next()=0;  
    }  
    delete p;  
    return rab; }
```

Шаблон класса List (4)

```
bool insert_after(const T& k, const T& after) {  
    Item<T> *c;  
    find(c, after);  
    if(c==0) return 0;  
    c->get_next() =  
        new Item<T>(k, c->get_next());  
    return 1;  
}
```

Шаблон класса List (5)

```
bool remove(const T& k) {  
    Item<T> *b, *c;  
    b=find(c,k);  
    if(c==NULL) return 0;  
    if(b==NULL) {  
        front=front->get_next(); delete (c); }  
    else {  
        b->get_next()=c->get_next(); delete(c); }  
    return 1;}  
  
void push_front(const T& elem) {  
    front = new Item<T>(elem,front);  
    if(back==0) back = front; }
```

Шаблон класса List (6)

```
T& pop_front() {  
    if(front==0)  
        ERR("List::pop_front: list is empty");  
    Item<T> *p=front; rab = p->get_node();  
    front=p->get_next(); delete p;  
    if(front==0) back=0;  
    return rab; }
```

Шаблон класса List (7)

```
void sort() { Item<T> *D=0;
    while(front!=0) {
        Item<T> *p=front; rab = front->get_node();
        while(p=p->get_next()) if(p->get_node()>rab)
            rab=p->get_node();
        D = new Item<T>(rab,D); remove(rab); }
    front = D; back = front;
    while(back->get_next()!=0) back=back-
    >get_next();
}
```

Шаблон класса List (8)

```
void revers() { Item<T> *D=0;
    while(front!=0) D = new
    Item<T>(pop_front(),D);
    front = D; back=front;
    while(back->get_next()!=0)back=back-
    >get_next();
}
```


Шаблон класса List (9)

```
T& operator[](int i) {  
    Item<T>* p=front;  
    while(i-- && p) p=p->get_next();  
    if(p) return p->get_node();  
    ERR("LIST::operator[]: end of List appear");}  
  
friend ostream& operator<<  
    (ostream& out, const List<T>& a) {  
    Item<T>* p=a.front;  
    while(p) {  
        out << p->get_node() << ' ';  
        p=p->get_next();  
    }  
    return out;  
}; // template <typename T> class List
```

Пример I

```
#include <iostream>
using namespace std;
#include "My_str.h"
#include "SQL.h"
#include <windows.h>

HANDLE hStdOut = GetStdHandle(STD_OUTPUT_HANDLE);

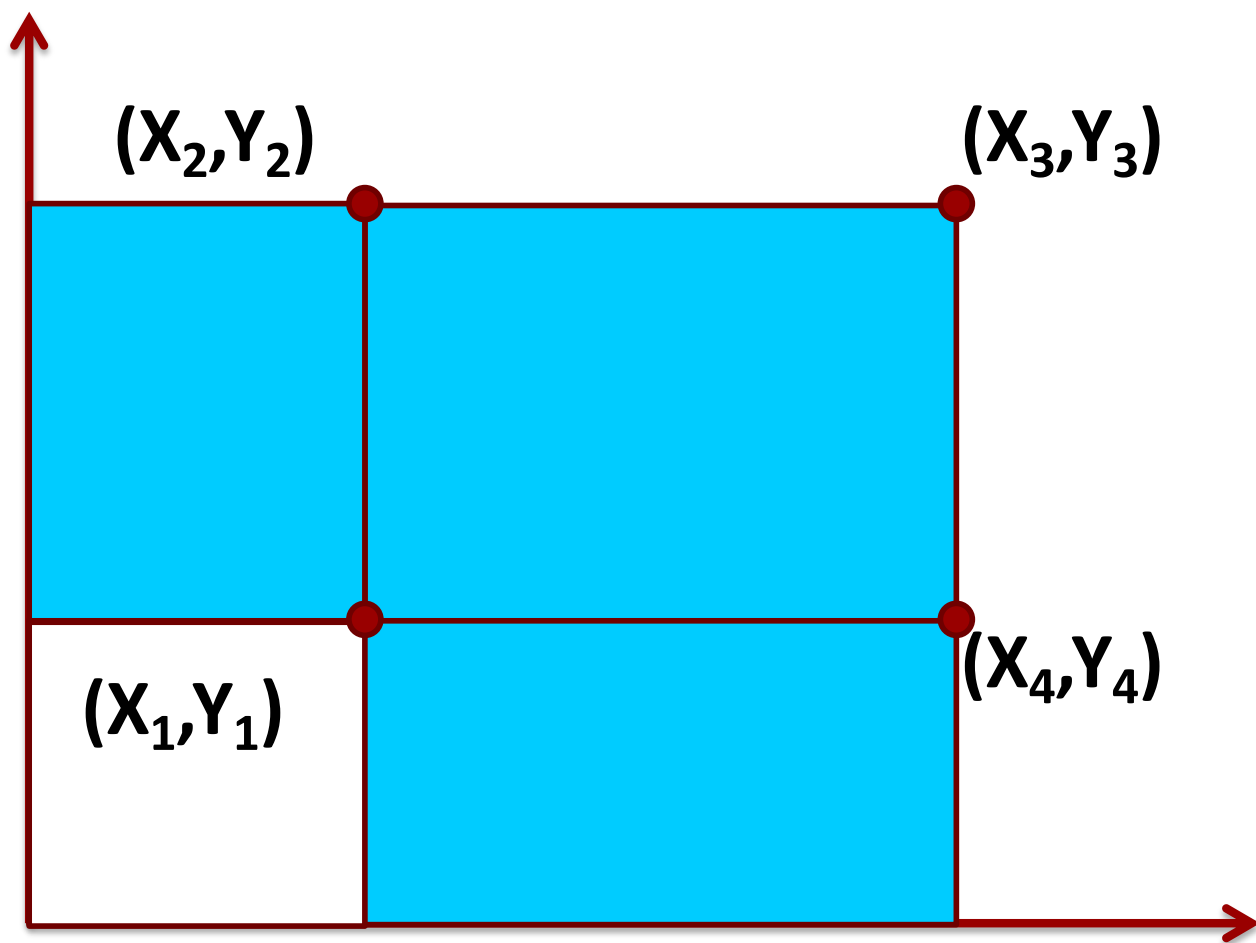
int main() {
    My_string x;
    Stack<My_string> a;
    Queue<My_string> b;
    List<My_string> c;
    while(cin >> x) { a.push(x); b.put(x); c.push_back(x); }
    while(!b.empty()) cout << b.get() << endl;
    SetConsoleTextAttribute(hStdOut, FOREGROUND_BLUE | FOREGROUND_INTENSITY);
    while(!a.empty()) cout << a.pop() << endl;
    SetConsoleTextAttribute(hStdOut, FOREGROUND_GREEN | FOREGROUND_INTENSITY);

    for(int i=3; i>0; i--) { x=c[i]; c.remove(x); }
    c.revers(); cout << c;
    SetConsoleTextAttribute(hStdOut, FOREGROUND_RED | FOREGROUND_INTENSITY);
    c.sort(); cout << c;
    c.push_front(x); c.insert_after(x,x);
    SetConsoleTextAttribute(hStdOut, FOREGROUND_BLUE | FOREGROUND_GREEN | FOREGROUND_RED | FOREGROUND_INTENSITY);
    cout << c;
    return 0;
}
```

Пример 2: вычисление площади прямоугольной фигуры, заданной произвольным перечислением координат вершин

```
#include <iostream>
#include <fstream>
#include "SQL.h" /* Файл с описанием шаблонов
    Стека, Очереди и Списка */
using namespace std;
```

Пример 2: вычисление площади прямоугольной фигуры, заданной произвольным перечислением координат вершин



$$\begin{aligned} S = & + x_3 * y_3 \\ & - x_4 * y_4 \\ & - x_2 * y_2 \\ & + x_1 * y_1 \end{aligned}$$

Вычисление площади фигуры

```

#include <iostream>
#include <fstream>
using namespace std;
#include "SQL.h"    /* Файл с описанием шаблонов Стекa, Очереди и Списка */

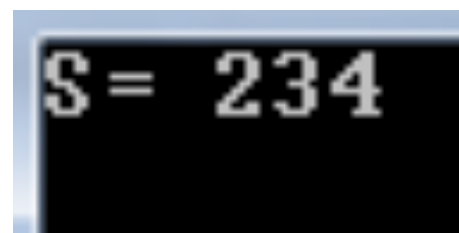
class point {
    int x,y;
public:
    point(int a=0,int b=0) { x=a; y=b; }
    friend ostream& operator<< ( ostream& a,const point& b) {
        return a << '(' << b.x << ',' << b.y << ')'; }
    friend istream& operator>> ( istream& a,point& b) {
        return a >> b.x >> b.y; }
    bool operator> (const point& b) {
        if (x==b.x) return y>b.y; else return x>b.x; }
    bool operator!= (const point& b) { return x!=b.x || y!=b.y; }
    bool operator== (const point& b) { return x==b.x && y==b.y; }
    point operator!() { return point(y,x); }
    int operator[](int i) { return i==1?x:y; }
};

```

```

int main() {
List<point> Sx,Sy;
point a,b; int i,k,s=0;
ifstream in("test_sq.dat");
    while(in>>a) { Sx.push_front(a); Sy.push_front(!a); }
    Sx.sort(); Sy.sort(); a=b=Sx[0];
    do { for(i=k=0;Sx[i][1]<a[1];i++);
        while(a[1]==Sx[i][1] && Sx[i][2]<a[2]) i++,k++;
        if(k%2) i--; else i++; a=!Sx[i]; s-=a[1]*a[2];
        for(i=k=0;Sy[i][1]<a[1];i++);
        while(a[1]==Sy[i][1] && Sy[i][2]<a[2]) i++,k++;
        if(k%2) i--; else i++; a=!Sy[i]; s+=a[1]*a[2];
    } while(a!=b);
    cout << "S= " << s << endl;
    return 0;
}

```



S= 234

5 10
5 20
6 11
19 11
32 18
19 12
32 20
19 13
34 7
19 14
34 14
6 18
21 11
35 5
8 5
21 12
35 15
8 7
21 13
8 12
21 14
22 7
8 15
22 11
9 13
22 14
9 14
22 18
15 1
23 5
15 5
23 10
15 20
23 15
15 24
23 20
16 23
24 2
16 18
24 7
16 7
24 18
16 2
24 23
17 5
25 1
17 10
25 5
17 15
25 20
17 20
25 24
18 7
31 11
18 11
31 12
18 14
32 10
18 18
32 13

Типы данных

- Базовые
- Указатели
- Составные
 - Массивы, строки,
 - Структуры, объединения, перечисления
- Абстрактные
 - ...<...>

ДЛИННЫЙ ФАКТОРИАЛ

$$n! = n \times (n - 1) \times \dots \times 3 \times 2 \times 1$$

$$10! = 10 \times 9 \times \dots \times 3 \times 2 \times 1 = 3628800,$$

сумма цифр числа $10!$ есть ...

$$3 + 6 + 2 + 8 + 8 + 0 + 0 = 27.$$

Найдите сумму цифр числа $N!$


```

#include <iostream>
using namespace std;

class number {
    unsigned char d[200]; int m;
public:
    number() { for(int i=0;i<200;i++) d[i]=0; d[1]=1; m=2; }
    void operator*(int k) {
        unsigned char z[200]; z[0]=0;
        int k2=k/10, k1=k%10, p=0;
        for(int i=1;i<=m;i++) { p=z[i-1]/10; z[i-1]=p%10; z[i]=d[i-1]*k2+d[i]*k1+p; }
        if(z[m]) if(p=z[m]/10) { z[m]=p%10; z[m+1]=p; }
        for(int i=0;i<=m;i++) d[i]=z[i]; }
    friend ostream& operator<<(ostream& a, const number& b) { int s=0;
        for(int i=b.m-1;i>0;i--) { a << char(b.d[i]+'0'); s+=b.d[i]; } a << '\t' << s << endl;
        return a; }
};

int main() {
    int N; number n;
    cin >> N;
    if(N==100) N=99;
    for(int i=2;i<=N;i++) n*=i;
    cout << "N=" << N << " N!=" << n << endl;

    return 0;
}

```

d_m	d_{m-1}	...	d_3	d_2	d_1	0
	$k_1 * d_{m-1}$...	$k_1 * d_3$	$k_1 * d_2$	$k_1 * d_1$	
$k_2 * d_{m-1}$...	$k_2 * d_3$	$k_2 * d_2$	$k_2 * d_1$		
$k_2 * d_{m-1} + k_1 * d_m$...		$k_2 * d_0 + k_1 * d_1$	

```

#include <iostream>
using namespace std;

class number {
    unsigned char d[200]; int m;
public:
    number() { for(int i=0;i<200;i++) d[i]=0; d[1]=1; m=2; }
    void operator*(int k) {
        unsigned char z[200]; z[0]=0;
        int k2=k/10, k1=k%10, p=0;
        for(int i=1;i<=m;i++) { p=z[i-1]/10; z[i-1]=p%10; z[i]=d[i-1]*k2+d[i]*k1+p; }
        if(z[m]) if(p=z[m]/10) { z[m-1]=p%10; z[m]=p; }
        for(int i=0;i<m;i++) d[i]=z[i]; }
    friend ostream& operator<<(ostream& a, const number& b) { int s=0;
        for(int i=b.m-1;i>0;i--) { a << char(b.d[i]+'0'); s+=b.d[i]; } a << '\t' << s << endl;
        return a; }
};

```

d_m	d_{m-1}	...	d_3	d_2	d_1	0
	$k_1 * d_{m-1}$...	$k_1 * d_3$	$k_1 * d_2$	$k_1 * d_1$	
$k_2 * d_{m-1}$...	$k_2 * d_3$	$k_2 * d_2$	$k_2 * d_1$		
$k_2 * d_{m-1} + k_1 * d_m$...		$k_2 * d_0 + k_1 * d_1$	

```

int main() {
    int N; number n;
    cin >> N;
    if(N==100) N=99;
    for(int i=2;i<=N;i++) n*=i;
    cout << "N=" << N << " N!= " << n << endl;
}

```

```
return 0;
```

```

100
N=99 N!= 93326215443944152681699238856266700490715968264381621468592963895217599
993229915608941463976156518286253697920827223758251185210916864000000000000000000
000000 648

```

ДЛИННОЕ 2^N

Какова сумма цифр числа 2^n ?

$$0 \leq n \leq 1000$$

$$2^{1000} \approx 10^k \quad k \approx 1000 * \lg 2 \quad k \approx 302$$

```

#include <iostream>
using namespace std;

class number {
    unsigned char d[400]; int m;
public:
    number() { for(int i=0;i<400;i++) d[i]=0; d[1]=1; m=2; }
    void operator*(int k) {
        unsigned char z[400]; z[0]=0;
        int k2=k/10, k1=k%10, p=0;
        for(int i=1;i<=m;i++) { p=z[i-1]/10; z[i-1]=p%10; z[i]=d[i-1]*k2+d[i]*k1+p; }
        if(z[m]) if(p=z[m]/10) { z[m]=p%10; m++; }
        for(int i=0;i<m;i++) d[i]=z[i]; }
    friend ostream& operator<<(ostream& a, const number& b) { int s=0;
        for(int i=b.m-1;i>0;i--) { a << char(b.d[i]+'0'); s+=b.d[i]; } a << '\t' << s << endl;
        return a; }
};

int main() {
    int M; number m;

    cin >> M;
    for(int i=1;i<=M;i++) m*=2;
    cout << "M=" << M << " 2**M= " << m << endl;

    return 0;
}

```

Конкретно

```
#include <iostream>
using namespace std;
```

```
class number {
    unsigned char d[400]; int m;
public:
    number() { for(int i=0;i<400;i++) d[i]=0; d[1]=1; m=2; }
    void operator*(int k) {
        unsigned char z[400]; z[0]=0;
        int k2=k/10, k1=k%10, p=0;
        for(int i=1;i<=m;i++) { p=z[i-1]/10; z[i-1]%=10; z[i]=d[i-1]*k2+d[i]*k1+p; }
        if(z[m]) if(p=z[m]/10) { z[m-1]%=10; z[m]=p; }
        for(int i=0;i<=m;i++) d[i]=z[i]; }
    friend ostream& operator<<(ostream& a, const number& b) { int s=0;
        for(int i=b.m-1;i>0;i--) { a << char(b.d[i]+'0'); s+=b.d[i]; } a << '\t' << s << endl;
        return a; }
};
```

```
int main() {
    int M; number m;
```

```
    cin >> M;
    for(int i=1;i<=M;i++) m*=2;
    cout << "M=" << M << " 2**M=" << m << endl;

    return 0;
}
```

Конкретно

```
1000
M=1000 2**M= 1071508607186267320948425049060001810561404811705533607443750388370
35105112493612249319837881569585812759467291755314682518714528569231404359845775
74698574803934567774824230985421074605062371141877954182153046474983581941267398
767559165543946077062914571196477686542167660429831652624386837205668069376
1366
```

```
#include <iostream>
using namespace std;
```

```
void ERR(const char* s) { cerr << s << endl; exit(1); }
```

Абстрактно

```
template<int Size> class number {
    unsigned char d[Size]; int m;
```

```
public:
```

```
number(int k=0) { for(m=0; k; m++) { d[m]=k%10; k/=10; } for(int i=m--; i<Size; i++) d[i] = 0; }
```

```
number(const number& n) { for(int i=0; i<Size; i++) d[i] = n.d[i]; m = n.m; }
```

```
number operator+(const number& n) { int p=0; number a(n); a.m = max(m,n.m);
    for(int i=0; i<=a.m; i++) { p = (a.d[i] += d[i] + p) / 10; a.d[i] %= 10; }
    if(p) { if(a.m==Size-1) ERR("number::operator+: array overflow"); a.d[++a.m] = p; }
    return a; }
```

```
number operator*(int k) { int p=0; number a(*this);
    if(k<0 || k>9) ERR("number::operator*: parameter must be a decimal digit!");
    if(!k) return number(0);
    for(int i=0; i<=m; i++) { p = (a.d[i] = a.d[i]*k + p) / 10; a.d[i]%=10; }
    if(p) { if(a.m==Size-1) ERR("number::operator*: array overflow"); a.d[++a.m] = p; }
    return a; }
```

```
void operator~(void) {
    if(m==Size-1) ERR("number::operator~: array overflow");
    if(!m && !d[0]) return;
    for(int i=m++; i>=0; i--) d[i+1] = d[i]; d[0] = 0; }
```

```

number operator*(const number& k) { number s(0);
    for(int i=k.m;i>=0;i--) { ~s; s = s + (*this) * k.d[i]; }
    return s; }
int operator[](int i) { return (i>=0 && i<=m) ? d[i]: -1; }
friend ostream& operator<<(ostream& a, const number& b) {
    for(int i=b.m;i>=0;i--) a << char(b.d[i]+'0'); return a; }
};

```

Абстрактно

```

int main() { int N,s=0;
number<200> f(1),m;
    cin >> N;
    for(int i=1;i<=N;i++) { m = i; f = f * m; }
    for(int i=0;;i++) { int d; if((d=f[i]) != -1) s+=d; else break; }
    cout << N << "!= " << f << '\t' << s << endl;

```

```

number<310> p(1);
    cin >> N; s=0;
    for(int i=1;i<=N;i++) p = p * 2;
    for(int i=0;;i++) { int d; if((d=p[i]) != -1) s+=d; else break; }
    cout << "2**" << N << "= " << p << '\t' << s << endl;

```

```

number<300> fb1(1), fb2(1), fb; s=0;
    for(int i=3;i<=N;i++) { fb = fb2; fb2 = fb2 + fb1; fb1 = fb; }
    for(int i=0;;i++) { int d; if((d=fb2[i]) != -1) s+=d; else break; }
    cout << "Fibb(" << N << ")= " << fb2 << '\t' << s << endl;
    return 0;

```

```
100
100!= 933262154439441526816992388562667004907159682643816214685929638952175999932299156089414639761565182862536979208272
23758251185210916864000000000000000000000000    648
1000
2**1000= 107150860718626732094842504906000181056140481170553360744375038837035105112493612249319837881569585812759467291
755314682518714528569231404359845775746985748039345677748242309854210746050623711418779541821530464749835819412673987675
59165543946077062914571196477686542167660429831652624386837205668069376   1366
Fibb(1000)= 434665576869374564356885276750406258025646605173717804024817290895365554179490518904038798400792551692959225
93080322634775209689623239873322471161642996440906533187938298969649928516003704476137795166849228875   1005
```


АСТЕРОИДЫ

Три астероида летят в космическом пространстве. Каждый из них движется с постоянной скоростью и в постоянном направлении. На всех трех астероидах существуют разумные формы жизни. В обычное время обитатели астероидов понемногу развиваются, осваивают новые технологии и готовятся выйти в космическое пространство. Но все радикально меняется, когда все три космических тела находятся на одной прямой. Когда происходят такие "затмения", жители астероидов начинают чувствовать себя неуютно. Они видят только одно из двух привычных тел на небосводе, и это порождает чувство какой-то неуверенности. Гуманоиды не могут заниматься привычными делами, некоторые даже начинают поговаривать о приближающемся конце света. О полетах в космос в такие моменты можно забыть: на всех трех астероидах разом случается экономический кризис. Ваша задача - определить, сколько еще таких кризисов случится в будущем. Именно во время таких кризисов и приходится допускать к управлению биржами системы типа "SkyNet"

Замечание 1. Астероиды настолько малы, что их можно считать точками. Чтобы слишком не усложнять анализ, будем считать, что астероиды вполне могут оказываться одновременно в одной и той же точке пространства. При этом они не сталкиваются, а пролетают сквозь друг друга. Если угодно, можно думать, что на самом деле они просто проходят друг от друга на незначительном расстоянии.

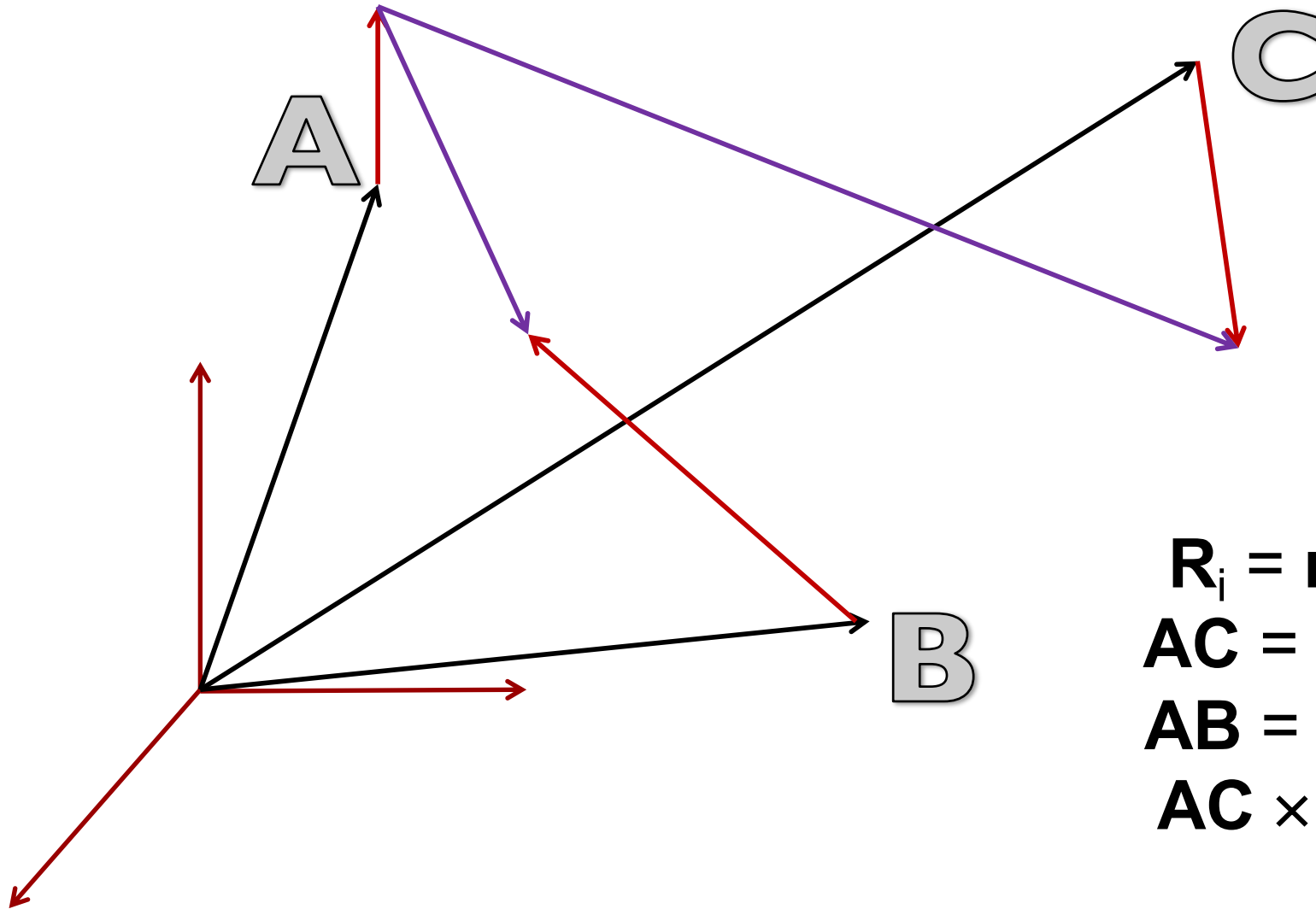
Замечание 2. Под "будущим" понимаются все моменты времени начиная с начала наблюдения. Само начало наблюдения тоже считаем частью "будущего".

Вход. Входной файл содержит три строки, по одной на каждый астероид. Каждая строка содержит 6 чисел X , Y , Z , V_x , V_y , V_z . Первые три числа задают координаты астероида в момент начала наблюдения, следующие три числа - координаты его вектора скорости. Все числа целые и не больше 20.

Выход. Одно целое число, равное количеству экономических кризисов начиная с момента наблюдения. Если количество "кризисных моментов" бесконечно, выведите число -1.

$$(\mathbf{r}_C - \mathbf{r}_A)(\mathbf{r}_B - \mathbf{r}_A) + t^*[(\mathbf{r}_C - \mathbf{r}_A)(\mathbf{v}_B - \mathbf{v}_A) + (\mathbf{v}_C - \mathbf{v}_A)(\mathbf{r}_B - \mathbf{r}_A)] + t^{2*}(\mathbf{v}_C - \mathbf{v}_A)(\mathbf{v}_B - \mathbf{v}_A) = 0$$

$$\mathbf{X} + t^* \mathbf{Y} + t^{2*} \mathbf{Z} = 0$$



$$\begin{aligned} \mathbf{R}_i &= \mathbf{r}_i + t^* \mathbf{v}_i \\ \mathbf{AC} &= \mathbf{R}_C - \mathbf{R}_A \\ \mathbf{AB} &= \mathbf{R}_B - \mathbf{R}_A \\ \mathbf{AC} \times \mathbf{AB} &= 0 \end{aligned}$$

```

#include <iostream>
#include <cmath>

using namespace std;

template <typename T> class vector {
    T v[3];
public:
    vector(T a=0, T b=0, T c=0) { v[0]=a; v[1]=b; v[2]=c; }
    vector(const vector& a) { v[0]=a.v[0]; v[1]=a.v[1]; v[2]=a.v[2]; }
    T& operator[](int i) { return v[i]; }
    vector operator+(vector b) { return vector(v[0]+b[0],v[1]+b[1],v[2]+b[2]); }
    vector operator-(vector b) { return vector(v[0]-b[0],v[1]-b[1],v[2]-b[2]); }
    vector operator*(vector b) { return vector(v[1]*b[2]-v[2]*b[1],v[2]*b[0]-v[0]*b[2],v[0]*b[1]-v[1]*b[0]); }
    int operator==(vector b) { return v[0]==b[0] && v[1]==b[1] && v[2]==b[2]; }
    friend ostream& operator<<(ostream& a, vector& b) { return a << '(' << b[0] << ',' << b[1] << ',' << b[2] << ')'; }
    friend istream& operator>>(istream& a, vector& b) { return a >> b[0] >> b[1] >> b[2]; }
};

```

```

int main() {
int k; double b,d;
vector<int> n,r1, v1, r2, v2, r3, v3, ab, ac, AB, AC,X,Y,Z;
    cin >> r1 >> v1 >> r2 >> v2 >> r3 >> v3;
    ab = r2 - r1; ac = r3 - r1;
    AB = v2 - v1; AC = v3 - v1;
    X = ab*ac; Y = AB*ac + ab*AC; Z = AB*AC;

    cout << X << '\t' << Y << '\t' << Z << endl;

    if( (X==n) && (Y==n) && (Z==n) ) k=-1;
    else if( Z==n ) { if( Y==n ) k=0;
        else for(int i=0;i<3;i++) if(Y[i]) { if( -(double)X[i]/Y[i] >= 0. ) k=1; else k=0; break; } }
        else for(int i=0;i<3;i++) if(Z[i]) { b=Y[i]/Z[i]; if( (d=b*b-4.*X[i]/Z[i]) >= 0 && sqrt(d)-b>=0) k=1; else k=0; break;}

    cout << k << endl;

    return 0;
}

```

```

int main() {
int k; double b,d;
vector<int> n,r1, v1, r2, v2, r3, v3, ab, ac, AB, AC,X,Y,Z;
    cin >> r1 >> v1 >> r2 >> v2 >> r3 >> v3;
    ab = r2 - r1; ac = r3 - r1;
    AB = v2 - v1; AC = v3 - v1;
    X = ab*ac; Y = AB*ac + ab*AC; Z = AB*AC;

    cout << X << '\t' << Y << '\t' << Z << endl;

    if( (X==n) && (Y==n) && (Z==n) ) k=-1;
    else if( Z==n ) { if( Y==n ) k=0;
        else for(int i=0;i<3;i++) if(Y[i]) { if( -(double)X[i]/Y[i] >= 0. ) k=1; else k=0; break; } }
        else for(int i=0;i<3;i++) if(Z[i]) { b=Y[i]/Z[i]; if( (d=b*b-4.*X[i]/Z[i]) >= 0 && sqrt(d)-b>=0) k=1; else k=0; break;}

    cout << k << endl;

    return 0;
}

```

```

D:\Rab_c\Контрольная\2015\Первая>v
-1 0 0 0 0 0
0 -1 0 0 1 0
1 1 0 0 -1 0
<0,0,3> <0,0,-3> <0,0,0>
1

```

```

D:\Rab_c\Контрольная\2015\Первая>v
1 0 0 1 2 3
0 1 0 1 2 3
1 1 0 1 2 3
<0,0,-1> <0,0,0> <0,0,0>
0

```

Основы анализа алгоритмов

Алгоритм - точное предписание, задающее процесс преобразования исходных данных в результаты.

■ Свойства:

- *Результативность*
- *Конечность*
- *Однозначность*
- *Массовость*
- *Детерминированность*
- *Эффективность*