字符串

字符型

字符赋值给整数：

```
#include <bits/stdc++.h>//1-1967-1  javacn
using namespace std;
int main()
{
	char a;
	cin>>a;
	int x = a;
	cout<<x;
	return 0;
}

#include<bits/stdc++.h>//1-1967-2  liuchunhui001
using namespace std;
int main()
{
	char a;
	cin >> a;
	cout << (int)a << endl;
}
```

整数赋值给字符：

```
#include <bits/stdc++.h>//2-1968  javacn
using namespace std;
int main()
{
	int x;
	cin>>x;
	char a = x;
	cout<<a;
	return 0;
}
```

字符可以直接比较大小，本质上是比较的 ascii 码：

```
#include <bits/stdc++.h>//3-1970  javacn
using namespace std;
int main()
{
	char c;
	cin>>c;
	//判断是什么字符
	if(c >= 'A' && c <= 'Z')
	{
		cout<<"upper";
	}
	else if(c >= 'a' && c <= 'z')
	{
		cout<<"lower";
	}
	else if(c >= '0' && c <= '9')
	{
		cout<<"digit";
	}
	return 0;
}
```

小写字母和对应的大写字母的 ascii 码的差值是 32：

```
#include <bits/stdc++.h>//4-1971  javacn
using namespace std;
int main()
{
	char c;
	cin>>c;
	//判断大小写，转换
	if(c >= 'A' && c <= 'Z'){	c = c + 32; }
	else			{	c = c - 32; }
	cout<<c;
	return 0;
}
```

判断字符的范围，如果是'a'~'y'，或者是'A'~'Y'，直接+1，如果是'z'或者是'Z'，则替换为'a'、'A'。

```
#include<bits/stdc++.h>//5-1969-1  javacn
using namespace std;
int main()
{
	char c;
	cin>>c;
	if(c>='A'&&c<='Y'||c>='a'&&c<='y')
	{
		c++;
		cout<<c;
	}
	else if(c == 'Z'){	cout<<'A';  }
	else if(c == 'z'){	cout<<'a';  }
	return 0;
}
```

```
#include<bits/stdc++.h>//5-1969-2  18963137059
using namespace std;
int main()
{
	char ch;cin>>ch;
	if(ch=='z'||ch=='Z')	{	ch=ch-25;	}
	else			{	ch=ch+1;	}
	cout<<ch;
	return 0;
}
```

```cpp
#include <bits/stdc++.h>//6-1093-1  javacn
using namespace std;
int main() {// 可以直接利用字符可以进行比较、运算的特性，将字符作为循环变量。
	char i;// 定义字符类型，字符可以直接比较和计算
	for(i = 'a';i <='z';i++)
	{
		cout<<i;
		if(i == 'm' || i == 'z'){	cout<<endl; }
	}
	for(i = 'z';i >= 'a';i--)
	{
		cout<<i;
		if(i == 'n'){	cout<<endl; }
	}
	return 0;
}
#include <bits/stdc++.h>//6-1093-2  dragoncatter    每 13 个输出换行
using namespace std;
int main(){
	int c=0;
	for(char i='a'; i<='z'; i++)
	{
		cout<<i;
		c++;
		if(c%13==0) cout<<endl;
	}
	c=0;
	for(char i='z'; i>='a'; i--)
	{
		cout<<i;
		c++;
		if(c%13==0) cout<<endl;
	}
	return 0;
}
```

先得到数字矩形，再将数字矩形中的数字换算成对应字母的 ascii 码。

```
#include <bits/stdc++.h>//7-1974  javacn
using namespace std;
int n;
int main()
{
	cin>>n;
	char c;
	for(int i = 1;i <= n;i++)
	{
		for(int j = 1;j <= n;j++)// 这道题可以自己写一个更好的解法
		{
			//1->'A' 2->'B'
			//'A':65
			//cout<<i;
			c = i + 64;
			cout<<c;
		}
		cout<<endl;
	}
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//8-1975  字母矩形 2  457587590
using namespace std;
int main()
{
	int i,j,n;
	cin>>n;
	char c;
	for(i=1;i<=n;i++)
	{
		for(j=1;j<=n;j++)
		{
			c=j+64;
			cout<<c;
		}
		cout<<endl;
	}
	return 0;
}
#include<iostream>//11-1094  字符图形 10- 字母三角  jiangyf70  仅供参考
using namespace std;
int main()
{
    int n;
    cin >> n;
    for(int i = 1; i <= n; i++)
    {
        for(int j = 1; j <= n + i - 1; j++)
        {
            if(j > n - i) cout << char('A' - 1 + i);
            else cout << " ";
        }
        cout << endl;
    }
    return 0;
}
```

字符图形 11- 字母正三角

先得到对应的数字矩形，再将数字换算成对应字母的 ascii 码。

```
#include <bits/stdc++.h>//12-1095  javacn
using namespace std;
int n;
int main()
{
	cin>>n;
	char c;
	for(int i = 1;i <= n;i++)
	{
		for(int j = 1;j <= n - i;j++)
		{
			cout<<" ";
		}

		for(int j = 1;j <= 2 * i - 1;j++)
		{
			c = j + 64;
			cout<<c;//cout<<j;
		}
		cout<<endl;
	}
	return 0;
}
```

数组查找及替换

```
#include<bits/stdc++.h>//13-1858-1 marsshu
using namespace std;
int a[110];
int main()
{
	// 定义与输入
	int n,b;
	cin>>n>>b;
	for(int i=0;i<n;i++)
	{
		cin>>a[i];
	}
	// 排序
	sort(a,a+n);
	// 找出不可被模除的数
	for(int i=0;i<n;i++)
	{
		if(a[i]%b!=0)
		{
			// 在字符编码 A~Z 中输出字母，否则输出数字
			if(a[i]>='A'&&a[i]<='Z')
			{
				cout<<char(a[i])<<" ";
			}
			else
			{
				cout<<a[i]<<" ";
			}
		}
	}
	return 0;
}
```

```cpp
#include <bits/stdc++.h>//13-1858-2  javacn
using namespace std;
int a[110];
int c[110];
int main()
{
	int n,b,m=0;
	cin>>n>>b;
	for(int i=0;i<n;i++)	{	cin>>a[i];	}
	// 从 a 数组中寻找 b 的倍数，如果不是 b 的倍数存入 c 数组
	for(int i=0;i<n;i++)
	{
		if(a[i]%b!=0)
		{
			c[m] = a[i];
			m++;
		}
	}

	sort(c,c+m);		// 默认从小到大排序

	for(int i=0;i<m;i++)
	{
		if(c[i]>=65&&c[i]<=90)
		{
			cout<<char(c[i])<<" ";
		}
		else
		{
			cout<<c[i]<<" ";
		}
	}
	return 0;
}
```

```cpp
#include <iostream>//13-1858-3  liuchunhui001 仅供参考
#include <stdlib.h>
#include <string>
#include <algorithm>
#include <vector>
using namespace std;
int main() {
	int n ,b;
	cin >> n >> b;
	int *p = (int *)malloc((n + 1) * 4);
	int *p1 =(int *)malloc((n + 1) * 4);
	int t = 0;
	for (int i = 1; i <= n; i++) {	cin >> *(p + i);  }
	int j = 1;
	for (int i = 1; i <= n; i++) {
		if (*(p + i) % b != 0) {*(p1 + j) = *(p + i);   j++;  t++;  }
	}
	int s;
	for   (int i = 1;i <= t - 1;i ++) {
		for (int k = 1;k <= t - i;k++) {
			if (*(p1 + k) > *(p1 + k + 1)) {
				s = *(p1 +  k + 1);
				*(p1 + k + 1) = *(p1 + k);
				*(p1 + k) = s;
			}
		}
	}
	for (int i = 1; i <= t; i++){
		if (*(p1 + i) >= 'A' && *(p1 + i) <= 'Z') {
			cout << (char)(*(p1 + i)) << ' ';
		}
		else {	cout << *(p1 + i) << ' ';	}
	}
}
```

```cpp
#include <bits/stdc++.h>//14-1972  13202828973
using namespace std;
int a[1010];
bool sushu(int n)
{     //判断素数函数
      bool f = true;
      for(int i = 2;i <= sqrt(n);i++)
      {
            if(n % i == 0)
            {
                  f = false;
                  break;
            }
      }
      //判断特殊情况
      if(n <= 1)    f = false;
      return f;
}
int main()
{
      for(int i=65;i<=90;i++)
      {
            if(sushu(i)==true)
            {
                  cout<<i<<" "<<(char)i<<endl;
            }
      }
      for(int i=97;i<=122;i++)
      {
            if(sushu(i)==true)
            {
                  cout<<i<<" "<<(char)i<<endl;
            }
      }
}
```

思路：循环所有的大写字母，大写字母的 ascii 在 65~90 之间，一定是 2 位数，因此只要判断这个两位数是否是回文 。

```
#include <bits/stdc++.h>//15-1973  javacn
using namespace std;

/*
思路：循环所有的大写字母
大写字母的 ascii 在 65~90 之间
一定是 2 位数，因此只要判断这个两位数是否是回文
*/

int main()
{
	// 循环所有的大写字母
	for(char i = 'A';i <= 'Z';i++)
	{
		// 如果这个字母的 ascii 码是回文
		if(i/10==i%10)
		{
			cout<<i<<endl;
		}
	}
	return 0;
}
```

字符串基础

```cpp
#include <bits/stdc++.h>//1-1093-1  javacn  推荐学习
using namespace std;
int main() {
	char i;// 定义字符类型，字符可以直接比较和计算
	for(i = 'a';i <= 'z';i++)
	{
		cout<<i;
		if(i == 'm' || i == 'z')	{	cout<<endl; }
	}
	for(i = 'z';i >= 'a';i--)
	{
		cout<<i;
		if(i == 'n')	{	cout<<endl; }
	}
	return 0;
}
#include <bits/stdc++.h>//1-1093-2 dragoncatter
using namespace std;
int main(){
	int c=0;
	for(char i='a'; i<='z'; i++)
	{
		cout<<i;	c++;	if(c%13==0) cout<<endl;
	}
	c=0;
	for(char i='z'; i>='a'; i--)
	{
		cout<<i;
		c++;
		if(c%13==0) cout<<endl;
	}
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//2-1101-1 时间的差 liuchunhui001
using namespace std;
int main()
{
	string a,b;
	int c,d;
	cin >> a >> b;
	c =(a[0] * 10 + a[1]) * 3600 + (a[3] * 10 + a[4]) * 60 + a[6] * 10 + a[7];
	d =(b[0] * 10 + b[1]) * 3600 + (b[3] * 10 + b[4]) * 60 + b[6] * 10 + b[7];
	cout << c - d;
}
#include <bits/stdc++.h>//2-1101-2   dragoncatter
using namespace std;
int main()
{
	int h,m,s,time1,time2;
	char c;
	cin>>h>>c>>m>>c>>s;
	time1=h*3600+m*60+s;
	cin>>h>>c>>m>>c>>s;
	time2=h*3600+m*60+s;
	cout<<time1-time2;
	return 0;
}
#include<iostream>//2-1101-3   jiangyf70  仅供参考
using namespace std;
int main(){
	int a1, a2, a3, b1, b2, b3;
	scanf("%d:%d:%d", &a1, &a2, &a3);
	scanf("%d:%d:%d", &b1, &b2, &b3);
	int a = a1 * 3600 + a2 * 60 + a3;
	int b = b1 * 3600 + b2 * 60 + b3;
	cout << a - b;
	return 0;
}
```

从字符串中取出任意一位是字符类型，因此本题用 string 读入后，将每一位字符 -'0' 转换为对应的整数后，再求和。

```
#include <bits/stdc++.h>//3-1115-1  javacn  推荐学习
using namespace std;

int main()
{
	/*
	  本题读入的整数不超过 200 位，因此用 string 读入
	*/
	string s;
	int i,r = 0;

	// 读入一个很大的整数，不会有空格，因此可以用 cin
	cin>>s;

	// 循环每位
	for(i = 0;i < s.size();i++)
	{
		// 此处要将字符数字，转换为真实的数字
		//'0' -> 0  '1' -> 1
		//r = r + (s[i] - '0');
		r = r + (s[i] - 48);
	}

	cout<<r<<endl;
}
```

```cpp
#include <bits/stdc++.h>//3-1115-2  数字和  w2ql
using namespace std;
int main()
{
	char s[202];
	scanf("%s",s);
	int i,sum=0;
	for(i=0;i<strlen(s);i++)
		sum+=s[i]-'0';
	printf("%d",sum);
	return 0;
}
```

```cpp
#include<iostream>//3-1115-3  jiangyf70
using namespace std;
int main()
{
	char c;
	int sum = 0;
	while(cin >> c)
		sum += c - '0';
	cout << sum;
	return 0;
}
```

国王的魔镜思路：一个字符串可能是接触过镜面的字符串的标准：长度为偶数且回文字符串

解决方法：定义函数，判断一个字符串是否满足接触过镜面的标准

在 main 中使用 while 语句，当字符串满足标准，截取一半覆盖字符串，直到不满足标准

```
#include <bits/stdc++.h>//4-1134-1  javacn  推荐学习
using namespace std;
bool mojing(string s)// 判断字符串是否满足接触过镜面的标准
{
	bool f = true;// 假设满足
	if(s.size()%2==0)		// 循环字符串长度的一半，对称位置比较
	{
		for(int i = 0;i < s.size() / 2;i++)
		{   // 下标 0-> 对称下标 s.size()-1
			if(s[i] != s[s.size()-i-1])
			{
				f = false;
				break;
			}
		}
	}
	else  // 奇数长度不满足
	{
		f = false;
	}
	return f;
}
int main()
{
	string s;
	cin>>s;
	while(mojing(s))	// 当 s 满足标准
	{
		s = s.substr(0,s.size()/2);  //s 截取一半
	}
	cout<<s.size();
}
```

```cpp
#include<bits/stdc++.h>//4-1134-2    alvinP
using namespace std;
int tong[91];
int main(){
	int cnt = 0;
	string s;
	cin >> s;
	for(int i = 0; i < s.size(); i++)    {     tong[s[i]]++;     }
	for(int i = 65; i <= 90; i++)
	{
		if(tong[i] > 0)    {     cnt++;      }
	}
	cout << cnt;
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//4-1134-3  13202828973
using namespace std;
int main()
{
	string a;
	cin>>a;
	while(1)
	{
		if(a.size()%2!=0)        {     break;      }
		string b=a;
		reverse(b.begin(),b.end());    //reverse()  反转字符串函数
		if(a!=b)            {     break;            }
		a=a.substr(0,a.size()/2);     //.substr(0,n)  获取从 0 到 n 的长度
	}
	cout<<a.size();
	return 0;
}
```

```cpp
#include <bits/stdc++.h>//4-1134-4    liuchunhui001
using namespace std;
string fun1(string s);
int main()
{
	string s;
	getline(cin,s);
	while (s == fun1(s))
	{
		s = s.erase(s.size() / 2,s.size() / 2);
	}
	cout << s.size();
}
string fun1(string s)
{
	string b;
	for (int i = s.size() - 1;i >= 0;i--)
	{
		b.push_back(s[i]);
	}
	return b;
}
```

```cpp
#include<bits/stdc++.h>//4-1134-5  liuchunhui001
using namespace std;
int main()
{
	string s;
	string ts,ts1,ts2;
	cin >> s;
	int t = 0;
	ts = s.substr(s.size() / 2,s.size() / 2);
	for (int i = ts.size() - 1;i >= 0;i--)
	{
		ts1.push_back(ts[i]);
	}
	s = s.substr(0,s.size() / 2);
	//cout << s << endl;
	while (s == ts1)
	{
		t += 1;
		ts = s.substr(s.size() / 2,s.size() / 2);
		//cout << ts;
		ts1.clear();
		for (int i = ts.size() - 1;i >= 0;i--)
		{
			ts1.push_back(ts[i]);
		}
		ts2 = s;
		s = s.substr(0,s.size() / 2);
	}
	//cout << ts2 << endl;
	cout << ts2.size();
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//4-1134-6  jiangyf70
using namespace std;
bool pd(char a[], int l) //判断 0~l 之间的字符是不是回文
{
	if(l % 2 != 0)
		return 0;
	for(int i = 0; i < l / 2; i++)
	{
		if(a[i] != a[l - 1 - i])
			return 0;
	}
	return 1;
}
int main()
{
	char a[1000];
	cin >> a;
	int l = strlen(a);
	while(pd(a, l))
		l /= 2;
		//判断 0~l 之间的字符是不是回文，若是，再判断 0~l/2 之间是不是回文
	cout << l;
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//5-1387-1  javacn  推荐学习
using namespace std;//'A'~'E'，解密为'V'~'Z'，也可以直接加上差值。
string s;
int main()
{
	getline(cin,s);
	for(int i = 0;i < s.size();i++)
	{
		if(s[i] >= 'F' && s[i] <='Z') //如果是大写字母，则解密
		{
			s[i] = s[i] - 5;
		}
		else if(s[i] >= 'A' && s[i] <= 'E')
		{
			s[i] = s[i] + ('V' - 'A');
		}
	}
	cout<<s;
	return 0;
}
#include<iostream>//5-1387-2  简单加密 jiangyf70  仅供参考
using namespace std;
int main(){
	char c[210];
	cin.getline(c, 210);
	for(int i = 0; c[i]; i++)
	{
		if(c[i] >= 'A' && c[i] <= 'Z')
		{
			c[i] = (c[i] - 'A' - 5 + 26) % 26 + 'A';
		}
	}
	cout << c;
	return 0;
}
```

```cpp
#include <bits/stdc++.h>//5-1387-3  javacn  推荐学习
using namespace std;
/*
1. 将大写字母解密
2. 解密方法
F~Z，解密方法是：-5
A~E，对应 V~Z
*/
string s;
int main()
{
	getline(cin,s);
	for(int i = 0;i < s.size();i++)	//逐个解密
	{
		if(s[i] >= 'A' && s[i] <= 'Z')	//如果是大写字母
		{
			//解密
			if(s[i] >= 'F' && s[i] <= 'Z')
			{
				s[i] = s[i] - 5;
			}
			else if(s[i] == 'A'){	s[i] = 'V'; }
			else if(s[i] == 'B'){	s[i] = 'W'; }
			else if(s[i] == 'C'){	s[i] = 'X'; }
			else if(s[i] == 'D'){	s[i] = 'Y'; }
			else if(s[i] == 'E'){	s[i] = 'Z'; }
		}
	}
	cout<<s;
	return 0;
}
```

找字典码最小的字符串

打擂台求最小：

```
#include <bits/stdc++.h>//6-1480  javacn
using namespace std;
int main()
{
	//s：代表字典码最小的字符串，x：代表运算结果
	string mi,x;
	int n,i;
	cin>>n;
	// 先读入一个字符串，作为被比较的对象
	// 假设它是字典码最小的字符串
	cin>>mi;

	// 再次读入 n-1 个字符串，并逐个和 mi 比较（打擂台）
	for(i = 2;i <= n;i++)
	{
		cin>>x;
		if(x < mi)
		{
			mi = x;
		}
	}

	cout<<mi;
}
```

字符串对比 1. 情况 1 和情况 2 很好判断，2. 如果不符合情况 1、情况 2，将两个字符串都转小写，如果相等，就是情况 3，否则是情况 4

```
#include <bits/stdc++.h>//7-1475-1   javacn 推荐学习
using namespace std;
string s1,s2;
int main()
{
	cin>>s1>>s2;
	// 情况 1
	if(s1.size() != s2.size())	{	cout<<1;	}
	else if(s1 == s2)		{	cout<<2;	}
	else
	{
		// 将两个字符串中的大写都改小写，如果是一样的，说明是情况 3
		for(int i = 0;i < s1.size();i++)
		{	//32 -> 'a' -  ‘A’
			if(s1[i] >= 'A' && s1[i] <= 'Z'){	s1[i] = s1[i] + 32;	}
			if(s2[i] >= 'A' && s2[i] <= 'Z'){	s2[i] = s2[i] + 32;	}
		}

		if(s1 == s2){	cout<<3;	}
		else		{	cout<<4;	}
	}
	return 0;
}
```

定义函数，将字符串转小写：

```
#include<bits/stdc++.h>//7-1475-2    javacn 推荐学习
using namespace std;
string s1,s2;
string tolower(string s) //将字符串转小写
{
	for(int i = 0;i < s.size();i++)
	{
		if(s[i] >= 'A' && s[i] <= 'Z')
		{
			s[i] = s[i] + 32;
		}
	}
	return s;
}

int main()
{
	cin>>s1>>s2;
	//情况 1：长度不等
	if(s1.size() != s2.size())          {    cout<<1;    }
	else if(s1 == s2)                   {    cout<<2;    } //情况 2：字符一样
	else if(tolower(s1) == tolower(s2)) {    cout<<3;    }
	else                                {    cout<<4;    }
	return 0;
}
```

出现次数最多的小写字母，数组计数法统计每个字母出现的次数，求出现次数最多的小写字母，相当于是求数组的最大数下标。

```
#include <bits/stdc++.h>//8-1478-1  javacn  推荐学习
using namespace std;
string s;
int a[200];
int main()
{
	cin>>s;
	for(int i = 0;i < s.size();i++)
	{
		a[s[i]]++;
	}

	char ma = 'a';// 出现次数最多的小写字母
	for(char i = 'a';i <= 'z';i++)
	{
		// 如果有多个最多，求编码最大的
		if(a[i] >= a[ma])
		{
			ma = i;
		}
	}
	cout<<ma;
}
```

```cpp
#include<bits/stdc++.h>//8-1478-2  marsshu
using namespace std;
int a[27];
int main()
{
	char c[110];
	int MAX=0,k=0;
	//定义与输入
	cin.getline(c,110);
	//对应字符落桶
	for(int i=0;i<strlen(c);i++)
	{
		a[c[i]-'a']++;
	}
	for(int i=0;i<26;i++)
	{
		if(a[i]>=MAX)
		{
			MAX=a[i];
			k=i;
		}
	}
	//输出最多的字符
	cout<<char(k+'a');
	return 0;
}
```

```cpp
#include <bits/stdc++.h>//9-1098-1  javacn  推荐学习
using namespace std;
/* 思路一：去掉 .reverse 之后和原来的字符串一样，就是回文
思路二：判断对称位置的字符有一个不等，就不是回文 */
string s;
int main()
{
	cin>>s;
	s.erase(s.size()-1,1);// 删除最后的点
	bool f = true;// 假设是回文
	for(int i = 0;i < s.size()/2;i++)
	{
		if(s[i] != s[s.size()-i-1])
		{
			f = false;
			break;
		}
	}
	if(f) cout<<"TRUE";
	else cout<<"FALSE";
	return 0;
}
#include<bits/stdc++.h>//9-1098-2 判断是否构成回文 hasome  推荐学习
using namespace std;
int main(){
	string s,s1;
	cin>>s;
	s=s.substr(0,s.size()-1);	// 去掉最后一个点
	s1=s;
	reverse(s.begin(),s.end());	// 反转
	// 判断反转后两个字符串是否相等
	if(s==s1)	cout<<"TRUE";
	else		cout<<"FALSE";
	return 0;
}
```

双指针用法

```cpp
#include<iostream>//9-1098-3  jiangyf70
#include<cstring>
using namespace std;
int main()
{
    char b[105];
    cin >> b;
    int l = strlen(b) - 2;
    bool flag = true;
    for(int i = 0, j = l; i < j; i++, j--)//双指针用法。
    {
        if(b[i] != b[j])
        {
            flag = false;
            break;
        }
    }
    if(flag) cout << "TRUE";
    else cout << "FALSE";
    return 0;
}
```

字符串中的空格移位

```
#include<iostream>//10-1102-1  jiangyf70 推荐学习
using namespace std;
int main()
{
	string s, k = "";
	getline(cin, s);
	for(int i = 0; i < s.size(); i++)
	{
		if(s[i] == ' ') k = s[i] + k;
		else k = k + s[i];
	}
	cout << k;
}
```

```
#include<bits/stdc++.h>//10-1102-2 zzb
using namespace std;
int main()
{
	char a[100];
	int i,s=0;
	fgets(a);
	for(i=0;a[i]!='\0';i++)
	{
	  if(a[i]==' '){    s++;      }
	}

	for(i=1;i<=s;i++){       cout<<' ';    }

	for(i=0;a[i]!='\0';i++)
	{
		if(a[i]!=' '){     cout<<a[i];    }
	}
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//10-1102-3 hasome
using namespace std;
int main(){
	string s,s1;
	int c=0;
	getline(cin,s);
// 遍历字符串的每个字符，如果是空格，记住空格数量，否则存放到另外一个字符串中
	for(int i=0;i<s.size();i++)
	{
		if(s[i]==' ') c++;
		else s1=s1+s[i];
	}
	for(int i=1;i<=c;i++){  cout<<" ";  }	// 输出字符串中空格的数量
	cout<<s1;	// 输出无空格的字符串
	return 0;
}
#include<bits/stdc++.h>//10-1102-4  hasome  仅供参考
using namespace std;
int main(){
	string s;
	int i,j,n=0,c=0;//c 表示空格的数量
	getline(cin,s);
	while(n!=s.size())		// 遍历字符串
	{
		if(s[n]==' ')
		{
			i=n;
			c++;  // 空格向前移动到 c-1 的位置
			for(j=i;j>=c;j--){		swap(s[j-1],s[j]);		}
		}
		n++;
	}
	cout<<s<<endl;
	return 0;
}
```

删除字符串中间的*

```
#include<bits/stdc++.h>//11-1125  laohuqiu
using namespace std;
int main()
{
	string l="",r="",s,a="";
	getline(cin,s);
	int i=0,j=s.length()-1;
	while(s[i]=='*')// 前一段*保存在 l 数组
	{
		l=l+'*';
		i++;
	}
	while(s[j]=='*')// 后一段*保存在 r 数组
	{
		r=r+'*';
		j--;
	}
	for(int k=i;k<=j;k++)// 中间一段不是*的保存在 a 数组
	{
		if(s[k]!='*')
		{
			a=a+s[k];
		}
	}
	cout<<l<<a<<r;
	return 0;
}
```

字符串的反码，求出每个字母在 26 个英文字母中的对称字母。

```
#include<bits/stdc++.h>//12-1133
using namespace std;
int main()
{
	string s;
	cin>>s;
	for(int i=0;i<s.size();i++)
	{
		//对字母求反码
		if(isupper(s[i]))
		{
			s[i] = 'Z' - s[i] + 'A';
		}
		else if(islower(s[i]))
		{
			s[i] = 'z' - s[i] + 'a';
		}
	}
	cout<<s;
	return 0;
}
```

```cpp
#include<iostream>//13-1312    看完动漫要几天？ jiangyf70
#include<cmath>
using namespace std;
int main()
{
    int a, b1, b2, c1, c2;
    scanf("%d", &a);
    scanf("%d:%d", &b1, &b2);
    scanf("%d:%d", &c1, &c2);
    int d = c1 * 60 + c2 - b1 * 60 - b2;  //转成分钟然后相减
    int e = ceil(a * 1.0 / d);  //向上取整
    printf("%d", e);
    return 0;
}
```

时钟旋转（2）

```cpp
#include<iostream>//14-1321  jiangyf70
using namespace std;

int main()
{
    int a1, a2, b1, b2;
    scanf("%d:%d", &a1, &a2);
    scanf("%d:%d", &b1, &b2);
    double a = a1 + a2 / 60.0;
    double b = b1 + b2 / 60.0;
    printf("%.1lf", (b - a) * 30);
    return 0;
}
```

逐个字符处理加密过程：

```
#include<bits/stdc++.h>//15-1402-1  字符串加密？ javacn  推荐学习
using namespace std;
string s;
int main()
{
	getline(cin,s);
	for(int i = 0;i < s.size();i++)
	{
		// 字符 +1 的情况
		if(s[i]>='a'&&s[i]<='y'||s[i]>='A'&&s[i]<='Y')  {    s[i]++;    }
		else if(s[i] == 'z')    {    s[i] = 'a'; }
		else if(s[i] == 'Z')    {    s[i] = 'A'; }
	}
	cout<<s;
	return 0;
}
```

字符串加密？

```
#include<iostream>//15-1402-2  jiangyf70
using namespace std;
int main()
{
    char s[100];
    cin.getline(s, 100);
    for(int i = 0; s[i]; i++)
    {
        if(s[i] >= 'A' && s[i] <= 'Z') s[i] = (s[i] - 'A' + 1) % 26 + 'A';
        if(s[i] >= 'a' && s[i] <= 'z') s[i] = (s[i] - 'a' + 1) % 26 + 'a';
    }
    cout << s;
    return 0;
}
```

字符串进阶

我是第几个单词

```
#include<bits/stdc++.h>//1-1012-1  javacn 推荐学习
using namespace std;
string s,w,f;//w 代表每个单词，f 代表要找的单词
int c = 0;
int r = 0;// 统计字符数量
int main() {
	getline(cin,s);
	cin>>f;
	// 遍历每个字符，分解单词 // 以 . 结束，. 可以不循环
	for(int i = 0;i < s.size() - 1;i++)
	{
		if(isalpha(s[i]))		// 如果是字母
		{
			w = w + s[i];
			if(!isalpha(s[i+1]))	// 如果单词结束
			{
				c++;
				r = r + w.size();// 求字符数量
				//cout<<w<<" "<<c<<endl;
				if(w == f)
				{
					cout<<c;
					return 0;
				}
				w = "";
			}
		}
	}

	cout<<r;
	return 0;
}
```

1. 分解并输出每个单词，并统计每个单词是第几个单词

This 1

is 2

a 3

Book 4

2. 判断每个单词是否是要找的单词，如果是，输出单词是第几个单词

3. 判断如果没有找到这个单词，输出字符总数量

```cpp
#include<bits/stdc++.h>//1-1012-2  remedy1314
using namespace std;
string s,s1,s2;
map<string,int> mp;
int k,p;
int main()
{
	getline(cin,s);
	cin>>s1;
	for(int i=0;i<s.size()-1;i++)
	{
		s2="";
		k++;
		while(i!=s.size()-1&&s[i]!=' ')
			s2+=s[i++];
		if(!mp.count(s2))
			mp[s2]=k;
		p+=s2.size();
	}
	if(!mp.count(s1))
		cout<<p;
	else
		cout<<mp[s1];
	return 0;
}
```

```cpp
#include<iostream>//1-1012-3  jiangyf70
#include<cstring>
using namespace std;

int main()
{
    char a[260][260];
    int i = 0;
    while(cin >> a[i]) i++;

    a[i-2][strlen(a[i-2]) - 1] = '\0';// 处理’.’；将’.’改写成’\0’；

    int maxl = 0;
    bool flag = 1;
    for(int j = 0; j < i - 1; j++)
    {
        if(strcmp(a[j],a[i - 1]) == 0)  //strcmp(a, b) 比较函数， a== b 返回 0; a > b 返回 > 0; a< b 返回 <0;
        {
            cout << j + 1;
            flag = 0;
            break;
        }
        maxl += strlen(a[j]);
    }
    if(flag) cout << maxl;
    return 0;
}
```

```cpp
#include<iostream>//1-1012-4  liangls
#include<algorithm>
#include<string>
using namespace std;
int main()
{
	string w,d;
	int cnt=0,sum=0;
	getline(cin,w);
	getline(cin,d);
	int len=w.size();
	w.erase(len-1);
	w = ''+w+' ';
	d = ' '+d+' ';
	len=w.size();
	if(w.find(d)==-1)
	{
		for(int i=0; i<=len-1; ++i)
		{
			if(w[i]!=' ')
			{
				sum++;
			}
		}
		cout<<sum;
		return 0;
	}
	int pos=w.find(d);
	for(int i=0; i<=pos; ++i)
	{
		if(w[i]==' ')	{	cnt++;	}
	}
	cout<<cnt;
	return 0;
}
```

调换位置，求出逗号的位置，将逗号之后的内容和逗号之前的内容分别拷贝到另一个数组。

```
#include<bits/stdc++.h>//2-1116-1  javacn  推荐学习 1 和 3
using namespace std;
char a[110],b[110];
int main()
{
	cin.getline(a,101);
	// 第 2 个参数要比题目要求读入的最大长度 +1，多留一个位置给空字符 '\0'
	int p = 0;
	for(int i = 0;i < strlen(a);i++)
	{
		if(a[i] == ',')
		{
			p = i;
			break;
		}
	}

	// 将逗号之后的内容拷贝到 b 数组
	int k = 0;// 表示 b 数组长度
	for(int i = p + 1;i < strlen(a);i++)
	{
		b[k] = a[i];//b 下标从 0 开始用
		k++;
	}
	b[k++] = ',';

	for(int i = 0;i < p;i++)
	{
		b[k++] = a[i];
	}

	cout<<b;
	return 0;
}
```

```cpp
#include<iostream>//2-1116-2  jiangyf70
using namespace std;
int main()
{
    char a;
    char b[1000], c[1000];
    int i = 0;
    while(cin >> a, a != ',') b[i++] = a;
    b[i] ='\0';
    i = 0;
    while(cin >> a) c[i++] = a;
    c[i] ='\0';
    cout << c << ',' << b << endl;
    return 0;
}
```

将逗号之后的单词截取（substr）出来，将逗号之前的单词截取，交换位置输出。

```cpp
#include <bits/stdc++.h>//2-1116-3 javacn  推荐学习
using namespace std;

int main()
{
      string s,s1,s2;
      cin>>s;
      int p = s.find(",");
      s1 = s.substr(0,p);
      s2 = s.substr(p+1);
      cout<<s2<<","<<s1;
}
```

字符数组参考解法：

```
#include <bits/stdc++.h>//3-1129-1  javacn  推荐学习 1 和 2、4
using namespace std;
int main()
{
	// 本题在 ab 在 int 范围，那么 ab 的长度最多 10+10 位，加上乘和等号，再多 2 位
	// 此处多开一些
	char s[30];
	int p = 0;// 记录 * 的位置
	cin >> s;

	// 将 * 之前和之后的整数分解出来
	// 虽然 ab 在 int 范围内，要注意乘可以超 int，因此定义 long long
	long long a = 0, b = 0;
	//* 之前的整数
	for(int i = 0; s[i] != '*'; i++)
	{
		a = a * 10 + (s[i] - '0');
		if(s[i + 1] == '*') p = i + 1;
	}

	//* 之后的整数
	for(int i = p + 1; s[i] != '='; i++)
	{
		b = b * 10 + (s[i] - '0');
	}

	cout << a * b;
	return 0;
}
```

string 参考解法：

```
#include <bits/stdc++.h>//3-1129-2 javacn  推荐学习
using namespace std;

string s;

int main()
{
    cin>>s;
    int p = s.find("*");// 找 * 的下标
    string s1 = s.substr(0,p);// 截取 * 之前的数字
    int len = s.size();
    string s2 = s.substr(p+1,len - p - 1);

    cout<<stoll(s1) * stoll(s2);// 转 long long 后相乘
    return 0;
}
```

```
#include<bits/stdc++.h>//3-1129-3   hasome
using namespace std;

string s,s1,s2;
int main()
{
      int a,b;
      cin>>s;
      s1=s.substr(0,s.find("*"));
      s2=s.substr(s.find("*")+1,s.size()-s.find("*")-2);
      a=stoi(s1);
      b=stoi(s2);
      cout<<a*b;
      return 0;
}
```

简单 a*b

```
#include<iostream>//3-1129-4 jiangyf70   推荐学习
#include<cstring>
using namespace std;
int main()
{
    long long a, b;
    scanf("%ld*%ld=", &a, &b);
    cout << a * b;
    return 0;
}
```

简单 a+b

```
#include<iostream>//4-1130-1 jiangyf70
#include<cstring>
using namespace std;
int main()
{
    char c;
    int a = 0, b = 0;
    while(cin >> c, c != '+') a = a * 10 + (c - '0');
    while(cin >> c, c != '=') b = b * 10 + (c - '0');
    cout << a + b;
    return 0;
}
```

解法 2：

```
#include<iostream>//4-1130-2  jiangyf70
using namespace std;
int main()
{
    int a, b;
    scanf("%d+%d=", &a, &b);
    cout << a + b;
    return 0;
}
```

解法一：根据 + 和 = 的位置拆出 2 个数字字符串，转整数后求和

```
#include <bits/stdc++.h>//4-1130-3 javacn  推荐学习 3 和 4
using namespace std;
string s,s1,s2;
int x,y;
int main()
{
	cin>>s;
	//123+45=
	int p = s.find("+");
	s1 = s.substr(0,p);
	s2 = s.substr(p+1,s.size()-1-p-1);
	//    cout<<s1<<endl<<s2;

	// 转换整数
	x = atoi(s1.c_str());
	y = atoi(s2.c_str());
	cout<<x+y;
	return 0;
}
```

解法二：直接用整数的形式读入 2 个整数后求和。

```
#include <bits/stdc++.h>//4-1130-4   javacn
using namespace std;
int x,y;
char c;
int main()
{
	// 注意读入格式
	cin>>x>>c>>y>>c;//c 读掉了字符 '+' 和 '='
	cout<<x+y;
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//5-1100-1 词组缩写  marsshu  仅做参考
using namespace std;
char daxie(char k)
{
	if(k>='a'&&k<='z')
	{
		k=char(k-32);
	}
	return k;
}
int main()
{
	string s;
	getline(cin,s);
	for(int i=0;i<s.size();i++)
	{
		if(i==0)//1. 首字母是字符
		{
			if(s[i]>='a'&&s[i]<='z'||s[i]>='A'&&s[i]<='Z')
			{
				cout<<daxie(s[i]);
			}
		}
		else//2. 非首字符且前一个是空格
		{
			if(s[i-1]==' ')
			{
				if(s[i]>='a'&&s[i]<='z'||s[i]>='A'&&s[i]<='Z')
				{
					cout<<daxie(s[i]);
				}
			}
		}
	}
}
```

词组缩写

```
#include<iostream>//5-1100-2  jiangyf70  推荐学习
using namespace std;
int main()
{
    char a[15];
    while(cin >> a)
    {
        if(a[0] >= 'a' && a[0] <='z' ) cout << char(a[0] - 32);
        else cout << a[0];
    }
    return 0;
}
```

第一步：找出每个单词的首字母　　首字母特征：(1) 如果是第一个字符且不是空格 || (不是第一个字符 && s[i] 当前字符不是空格 && s[i-1] 上一个字符是空格)

(2) s =" " + s; 从第 2 个字符开始循环，如果当前字符不是空格 && 上一个字符是空格，则是首字母！ 第二步：如果是小写转大写

```
#include<bits/stdc++.h>//5-1100-3  javacn  推荐学习
using namespace std;
int main()
{
      string s;
      int i;
      getline(cin,s);     // 读入字符串
      s = " " + s;
      // 其余字符，如果当前字符不是空格且上个字符是空格是首字母
      for(i = 1;i < s.size();i++)
      {
            if(s[i] !=' '&& s[i - 1] ==' ')          // 如果是首字母
            {
                  cout<<(char)toupper(s[i]);
            }
      }
      return 0;
}
```

思路：采用数数的思想，数出连续相同的字符有几个！

默认 c=0 每遇到一个字符，c++，并判断连续的字符是否结束，如果结束，输出字符及出现的次数，并清空计数器！

if(到了最后一个字符 || 当前字符 s[i] != 下一个字符 s[i+1]){ }

```
#include <bits/stdc++.h>//6-1103-1  javacn
using namespace std;
int main()
{
	string s;
	int i,c = 0;
	cin>>s;

	for(i = 0;i < s.size();i++)
	{
		c++;
		// 如果连续相同字符结束了
		if(i == s.size() - 1 || s[i] != s[i + 1])
		{
			if(c != 1)
			{
				cout<<c<<s[i];
			}
			else
			{
				cout<<s[i];
			}
			c = 0;
		}
	}
}
```

注意：

||：或者的特征是如果第一条件为 true，第二条件不判断结果直接为 true！

&&：并且的特征是如果第一条件为 false，第二条件不判断结果直接 false！

本题在判断时，要先判断到了最后一个字符 || 当前字符 != 下一个字符，防止越界。

```cpp
#include <bits/stdc++.h>//6-1103-2 字符串压缩 remedy1314
using namespace std;
string s; int main() {
	cin>>s;
	for(int i=0;i<s.size();i++)
	{
		int n=0;
		char x=s[i];
		while(s[i]==x)
		{
			n++;
			i++;
		}
		i--;
		if(n>1)
			cout<<n;
		cout<<x;
	}
	return 0;
}
#include<iostream>//6-1103-3 字符串压缩 jiangyf70
using namespace std;
int main(){
    char a[260];
    cin >> a;
    for(int i = 0, j = 0; a[i]; i = j)// 双指针算法
    {
        int k = 1; // 计数器
        j = i + 1;
        while(a[i] == a[j])      {    k++;      j++;       }
        if(k == 1)       cout << a[i];
        else             cout << k << a[i];
    }
    return 0;
}
```

```cpp
#include <bits/stdc++.h>//7-1104-1 字符串解压 javacn  推荐学习
using namespace std;
/*     12ab10c2ax
1. 分解出每个连续的数字，从而获得每个字母出现的次数
12 a       1 b       10 c       2 a       1 x
2. 循环对应的次数，输出对应的字符      */
string s,w = "";//w 存放每个连续的数字
int cnt;
int main()
{
      cin>>s;
      for(int i = 0;i < s.size();i++)
      {
            if(isdigit(s[i]))// 如果是数字
            {
                  w = w + s[i];
            }
            else
            {
                  // 本题连续的数字后面一定会有字母
                  // 如果是字母，输出字母出现了几次，并清空数字
                  //cout<<w<<" "<<s[i]<<endl;
                  if(w == "")       cout<<s[i];
                  else
                  {
                        // 转整数
                        cnt = atoi(w.c_str());
                        for(int j = 1;j <= cnt;j++)
                              cout<<s[i];
                  }
                  w = "";
            }
      }
      return 0;
}
```

双指针

```
#include<bits/stdc++.h>//7-1104-2  jiangyf70
using namespace std;
int main()
{
	string s;
	cin >> s;
	for(int i = 0; i < s.size(); i++)
	{
		int k = 0, j = i;
		while(s[j] >= '0' && s[j] <= '9')
		{
			k = k * 10 + s[j] - '0';
			j++;
		}
		if(k == 0)
			cout << s[j];
		else
		while(k--)
			cout << s[j];
		i = j;
	}
	return 0;
}
```

3 当只出现一次时，没有数字，所以需要特判！

```
#include <bits/stdc++.h>//7-1104-3  dragoncatter
using namespace std;
int main()
{
	int sum=0;
	string s;
	cin>>s;
	for(int i=0; i<s.size(); i++)
	{
		if(s[i]>='0'&&s[i]<'9')
		{
			sum=sum*10+s[i]-'0';
		}
		else
		{
			if(sum==0)
			{
				cout<<s[i];
			}
			while(sum--)
			{
				cout<<s[i];
			}
			sum=0;
		}
	}
	return 0;
}
```

先将两个字符串连成一个字符串，再逐个字符讨论，如果当前字符在 r 中不存在，则存入 r。

```
#include<bits/stdc++.h>//8-1105-1  javacn  推荐学习
using namespace std;
string s1,s2,s,r;
int main()
{
	cin>>s1>>s2;
	s=s1+s2;
	int i;
	for(i=0;i<s.size();i++)
	{
		if(r.find(s[i])==-1)	//r 中没有当前字符，则将当前字符存入 r
			r=r+s[i];
	}
	cout<<r;
	return 0;
}
#include<bits/stdc++.h>//8-1105-2 字符串连接 Nancy_001
using namespace std;
bool f[150];
int main()
{
	string s,s1;
	cin>>s>>s1;
	s+=s1;
	for(int i=0;i<s.size();i++)
	{
		if(f[s[i]]==0) //该字母未出现过
		{
			cout<<s[i];
			f[s[i]]=1;//标记为已出现
		}
	}
	return 0;
}
```

统计单词个数

数单词的开始（第一个字母）或者结束（最后一个字母）。

如果是单词的开始，可以判断：当前字符是字母 && 上一个字符不是字母。注意第一个字符如果是字母可能不满足条件，因此可以在字符串前面补一个空格。

字符数组参考解法：该解法没有在第 1 个字符前补空格。

```
#include <bits/stdc++.h>//9-1106-1    javacn
using namespace std;
int main(){
	char s[110];
	int num = 0;
	cin.getline(s,101);
	for(int i = 0;i < strlen(s);i++)
	{
		//判断单词的结尾
		if(s[i] != ' ' && (i == 0 || s[i + 1] == ' ')){    num++;    }
	}
	cout<<num<<endl;
	return 0;
}
```

string 参考解法：

```
#include<bits/stdc++.h>//9-1106-2    javacn    推荐学习
using namespace std;
int c=0;
string s;
int main() {
	getline(cin,s);
	//在前面补一个空格，保证所有的单词的第一个字符都满足：
	//当前字符是字母，上一个字符不是字母
	s = " " + s;
	for(int i = 1; i < s.size(); i++) {
		if(isalpha(s[i]) && !isalpha(s[i-1])) c++;//判断单词的开始
	}
	cout<<c;
	return 0;
}
```

统计单词个数

```
#include<iostream>//9-1106-3  jiangyf70   推荐学习
#include<cstring>
using namespace std;
int main()
{
	char a[100];
	int i = 0;
	while(cin >> a)
		i++;
	cout << i;
	return 0;
}
```

```
#include<bits/stdc++.h>//9-1106-4   tc_ljp6 仅供参考
using namespace std;
int main()
{
	string s;
	int c=0;
	while(cin>>s)
	{
		c++;
		if(getchar()=='\n')// 判断结束为换行符
		{
			break;
		}
	}
	cout<<c;
	return 0;
}
```

```
#include <bits/stdc++.h>//10-1107-1    javacn  推荐学习
using namespace std;
/*
1. 分解并输出每个单词
in
which
four
coins

2. 打擂台，求最长的单词
*/
string s,w = "",ma = "";
int main()
{
	getline(cin,s);
	// 循环每个字符
	for(int i = 0;i < s.size();i++)
	{
		// 如果是字母，一定是单词的一部分
		if(isalpha(s[i]))
		{
			w = w + s[i];
			// 如果单词结束
			if(i==s.size()-1||!isalpha(s[i+1]))
			{
				//cout<<w<<endl;
				if(w.size()>ma.size())
					ma = w;
				w = "";// 清空单词存放下一个单词
			}
		}
	}
	cout<<ma;
	return 0;
}
```

```cpp
#include <string>//10-1107-2    求英文句子中的最长单词 liuchunhui001
#include <iostream>
#include <vector>
using namespace std;
vector<string> subStrToVec(string str, char sep);
int main() {
	string str; //char a;
	getline(cin, str);
	vector<string> vecArr = subStrToVec(str, '');    // 转化
	int max_length = vecArr[0].length();       // 打印
	int index = 0;
	for (int i = 0; i < vecArr.size(); i++)
	{
		if (vecArr[i].length() > max_length)
		{
			max_length = vecArr[i].length();
			index = i;
		}
	}
	cout << vecArr[index];
	return 0;    //getchar();
}
vector<string> subStrToVec(string str, char sep) {
	vector<string> vecArr;
	int flagSub = 0;
	for (int i = 0; i < str.length() + 1; i++)
	{
		if (str[i] == sep or str[i] == '\0') {
			string temp = str.substr(flagSub, i - flagSub);
			vecArr.push_back(temp);
			flagSub = i + 1;
		}
	}
	return vecArr;
}
```

求英文句子中的最长单词

```
#include<iostream>//10-1107-3  jiangyf70
#include<cstring>
using namespace std;
int main()
{
    char a[260], b[260];
    int maxl = 0;
    while(cin >> a)
    {
        int l = strlen(a);
        if(maxl < l)
        {
            maxl = l;
            strcpy(b, a);
        }
    }
    cout << b;
    return 0;
}
```

除 2 取余，结果倒过来连成字符串：

```
#include<bits/stdc++.h>//11-1108-1   javacn   学完进制转换再做这题
using namespace std;
//10 进制转 2 进制
int n;
string r = "";// 存放转换结果
int main()
{
	cin>>n;
	// 当 n!=0 循环
	while(n != 0)
	{
		//cout<<n%2;// 取余
		// 将 n%2 的结果转换为字符 + 到 r 前面
		r = char(n%2+'0') + r;
		n=n/2;// 除 2
	}
	// 特判输入为 0 的情况
	if(r == "") cout<<0;
	else cout<<r;
	return 0;
}
```

正整数 N 转换成一个二进制数

```
#include<iostream>//11-1108-2  jiangyf70
using namespace std;
string i2b(int n, string s)
{
    if(n)
    {
        s = char(n % 2 + '0') + s;
        return i2b(n / 2, s);
    }
    return s;

}

int main()
{
    int n;
    cin >> n;
    if(n == 0) cout << 0;
    else cout << i2b(n, "");
    return 0;
}
```

查找子串并替换

要注意题目加粗描述的特殊情况，每次要从上一次找到的子串替换后的结束位置向后找。

```
#include <bits/stdc++.h>//12-1112 javacn
using namespace std;
int main()
{
	string s,s1,s2;
	getline(cin,s);
	getline(cin,s1);
	getline(cin,s2);

	int p=s.find(s1);	// 查找字串 s1 的下标
	while(p!=-1)
	{

		s.replace(p,s1.size(),s2);		// 替换

		p=s.find(s1,p+s2.size());  // 继续查找剩余 s1 的位置
	}
	cout<<s<<endl;
	return 0;
}
```

string 的解法：

```
#include<bits/stdc++.h>//13-1111-1    javacn    推荐学习
using namespace std;
/*
1. 分解并输出每个单词
2. 找到含有’a’的最长的单词
3. 判断没有这样的单词输出 NO
*/
string s,w,ma = "";
int main()
{
	getline(cin,s);
	for(int i = 0;i < s.size();i++)
	{
		if(isalpha(s[i]))// 连续的字母组成单词
		{
			w = w + s[i];
			if(i==s.size()-1||!isalpha(s[i+1])) // 判断单词是否结束
			{
				//cout<<w<<endl;
				if(w.find("a")!=-1&&w.size()>ma.size())
				{
					ma = w;
				}

				w ="";// 清空 w 存放下一个单词
			}
		}
	}

	if(ma == "") cout<<"NO";
	else cout<<ma;
	return 0;
}
```

找最长单词字符数组的解法：

```
#include<bits/stdc++.h>//13-1111-2  javacn
using namespace std;
bool has(char s[]) {// 判断字符数组中合适否有’a’
	bool r = false;
	for(int i=0; s[i]!='\0'; i++) {
		if(s[i]=='a') {   r=true;     break;     }
	}
	return r;
}
char s[310],w[110],r[110] = {'\0'};
int main() {
	int i,k,f=0;//f：假设没有答案
	cin.getline(s,301);// 读入字符串
	k = 0;
	for(i = 0; s[i]!='.'; i++)
	{    if(s[i] != ' ')
		{    w[k] = s[i];// 分解每个单词
			k++;
			if(s[i + 1] == ' ' || s[i + 1] == '.') // 到了单词末尾
			{
				w[k] = '\0';// 补结束字符
				if(has(w)==true) // 如果单词有 a
				{
					f=1;// 找到答案 // 单词比答案长 // 拷贝答案
					if(strlen(w)>strlen(r)) {    strcpy(r,w);     }
				}
				k=0;
			}
		}
	}
	if(f==1) {  cout<<r<<endl;          }     // 如果找到答案
	else cout<<"NO";
	return 0;
}
```

```cpp
#include<iostream>//13-1111-3    remedy1314
#include<cstdio>
#include<algorithm>
#include<cmath>
using namespace std;

string s,ss;
int n;

int main()
{
	while(cin>>s)
	{
		if(s[s.size()-1]=='.')
			s.erase(s.size()-1);
		if(s.size()>n&&(s.find('a')!=-1))
		{
			ss=s;
			n=s.size();
		}
	}
	if(ss=="")
		printf("NO");
	else cout<<ss;
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//13-1111-4    hasome
using namespace std;
int main()
{
    string s,s1,s2;    //s1 存放含有 a 的单词
    bool f=false;      //f 标记单词是否含 a
    //c 单词的长度，ma 已经保存起来含 a 单词的长度
    int i=0,ma=INT_MIN,c=0;
    getline(cin,s);
    while(s[i]!='.')
    {
        if(s[i]!=' ')
        {
            c++;  // 计算单词的长度
            s1=s1+s[i]; // 碰到空格前将字母组合成单词
            if(s[i]=='a'){    f=true;     }      // 判断是否含有字母 a
        }
        else
        {
    // 如果含有字母 a，而且长度比之前记的长，则将新找到的比较长的单词记住。
            if(f && c>ma)
            {
                s2=s1;
                ma=c;
                f=false;
            }
            s1="";
            c=0;
        }
        i++;
    }
    if(s2=="") cout<<"NO";  // 输出结果
    else cout<<s2;
    return 0;
}
```

/*
思路：
1、遍历字符串，
2、将每一个单词存放到变量 s1 中，并计算单词的长度 c 和记住是否含有字母 a
3、如果含有字母 a，而且单词长度比之前的长，则存放到 s2 中
4、如果没找到含字母 a 的单词，输出 NO。否则输出 S2
*/

```cpp
#include<iostream>//13-1111-5  jiangyf70
#include<cstring>
using namespace std;
bool f(char a[])
{
    for(int i = 0; a[i]; i++)
    {
        if(a[i] == 'a') return 1;
    }
    return 0;
}

int main()
{
    char a[100], b[100];
    int i = 0, maxl = 0;
    bool flag = 0;
    while(cin >> a)
    {
        if(f(a))
        {
            flag = 1;
            if(a[strlen(a) - 1] == '.') a[strlen(a) - 1] = '\0';
            if(maxl < strlen(a))
            {
                maxl = strlen(a);
                strcpy(b, a);
            }
        }
    }
    if(flag) cout << b;
    else cout << "NO";
    return 0;
}
```

隐藏的最大整数

注意：数字太大，不能转整数打擂台，要用字符串比较。

```
#include<bits/stdc++.h>//14-1113-1   javacn
using namespace std;
string s,w,ma;
int ans = 0;
int main()
{
	cin>>s;
	//  2
	//a12b456
	for(int i = 0;i < s.size();i++)      // 循环每个字符
	{
		if(isdigit(s[i])) // 如果是数字
		{
			w = w + s[i];// 则是连续数字的一部分
			// 如果连续数字结束了
			if(i==s.size() || !isdigit(s[i+1]))
			{
				// 数字，数字的开始位置的下标 + 1
				//cout<<w<<" "<<i - w.size() + 1 + 1<<endl;
				// 长的一定大，一样长，字典码大的一定大
				if(w.size()>ma.size() || (w.size()==ma.size()&&w>ma))
				{
					ma = w;
					ans = i - w.size() + 1 + 1;
				}
				w = "";
			}
		}
	}

	cout<<ans;
	return 0;
}
```

求：最大整数从第几个字符开始

思路：

1. 分解并输出连续的整数；
2. 求出每个整数的开始位置；
3. 打擂台求最大的整数及开始位置；

*/

双指针算法

数值超过 20 位了，long long 也不够，所以要用字符串比较，s.size() 大的数字大，一样长度的在比较字符大小 if(k.size() > maxn.size() || k.size() == maxn.size() && k > maxn)

```
#include<bits/stdc++.h>//14-1113-2  jiangyf70

using namespace std;

int main()
{
    string s;
    cin >> s;
    string maxn = "";
    int pos;
    for(int i = 0; i < s.size(); i++)
    {
        int j = i;
        string  k ="";
        while(s[j] >= '0' && s[j] <= '9')
        {
            k = k + s[j];
            j++;
        }
        if(k.size() > maxn.size() || k.size() == maxn.size() && k > maxn)
        {
            maxn = k;
            pos = i + 1;
        }
        i = j;
    }
    cout << pos;
    return 0;
}
```

```cpp
#include <bits/stdc++.h>//14-1113-3  hyb
using namespace std;

/**
 * 思路：遍历字符串
 * 对数字字符拼接字符串
 * 利用打擂台留下最大的数字字符串
 * 找到最大数字字符串的下标
 * 注意：数字字符串有可能超过整型范围，所以不应该用整型存放结果
 * @return 最大数字字符串的下标+1
 */
int main()
{
	string s,w="";
	string ma = "";
	cin>>s;
	for(int i=0; i<s.size(); i++)
	{
		if(isdigit(s[i]))
		{
			w = w + s[i];
			if(i==s.size()-1 || !isdigit(s[i+1]))
			{
				if(w.size()>ma.size() || w.size()==ma.size() && w>ma)
				{
					ma = w;
				}
				w = "";
			}
		}
	}
	cout<<s.find(ma)+1;
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//14-1113-4  hasome
using namespace std;
/*
	思路：	1、遍历字符串，找到是数字的，并将数字拼起来，并记住位数
	2、如果碰到不是数字的，则判断组成的数字是否比之前的大，如果大的话替换，并根据位数找出位数
	3、输出结果
*/
int main()
{
	string s;
	int i,p,r=0,ma=INT_MIN,c=0;
	cin>>s;
	for(i=0;i<s.size();i++)
	{
		if(isdigit(s[i]))
		{
			r=r*10+s[i]-'0';
			c++;
		}
		else
		{
			if(r>ma)
			{
				ma=r;
				r=0;
				// 根据位数和 i 的值找到这个数字的起始位置
				p=i-c+1;
			}
			c=0;
		}
	}
	cout<<p<<endl;
	return 0;
}
```

string 求解：

```
#include <bits/stdc++.h>//15-1114-1    javacn
using namespace std;
string s,s1,s2,t1,t2;
int main()
{
	cin>>s;
	int p = s.find("=");
	s1 = s.substr(0,p);// 截取 = 左右的子串
	s2 = s.substr(p+1);
	//devcpp 记得配置 c++11 支持才能用 stoi
	int z = stoi(s2);

	bool f = false;// 假设无解
	//s1 如果是 5 位数，可以加 4 次 + 号
	for(int i = 0;i < p - 1;i++)
	{
		t1 = s1.substr(0,i+1);
		t2 = s1.substr(i+1);
		//cout<<t1<<" "<<t2<<endl;
		int x = stoi(t1);
		int y = stoi(t2);

		if(x + y == z)
		{
			cout<<x<<"+"<<y<<"="<<z<<endl;
			f = true;
		}
	}

	if(f == false)
	{
		cout<<"Impossible!"<<endl;
	}
}
```

趣味填空。思路：根据=将左右字符串拆分出来，将左侧所有可能加加号的位置枚举出来，依次判断，添加加号是否成立。　　字符数组求解：

```
#include <bits/stdc++.h>//15-1114-2    javacn
using namespace std;
char s[20];
int n1,n2;// 左右两侧的数字
int p = -1;// 记录=的位置，相当于是=的左侧有多少位的数字
int main(){
	cin>>s;// 没有空格用 cin
	for(int i = 0;s[i] != '\0';i++)
	{
		if(s[i] =='=')	{	p = i;	continue;	}
		if(p == -1)	n1 = n1 * 10 + (s[i] - '0');	// 没有遇到过=
		else	n2 = n2 * 10 + (s[i] - '0');
	}
	// cout<<n1<<" "<<n2<<endl;// 输出测试
	// 尝试 n1 所有的拆分可能
	//p 位数有 p-1 种拆分的可能
	int t = 10,x1,x2;
	bool f = false;
	for(int i = 1;i <= p - 1;i++)
	{
		x1 = n1 / t;
		x2 = n1 % t;
		t = t * 10;
		//	cout<<x1<<" "<<x2<<endl;// 测试
		if(x1 + x2 == n2)
		{
			cout<<x1<<"+"<<x2<<"="<<n2<<endl;
			f = true;
		}
	}
	if(f == false)	cout<<"Impossible!";
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//15-1114-3    w2016010182
using namespace std;
long long a,b,c,sum;
int main()
{
	string s;
	getline(cin,s);
	int p1=s.find("=");
	c=stoll(s.substr(p1+1));
	for(int i=1;i<p1;i++)
	{
		a=stoll(s.substr(0,i));
		b=stoll(s.substr(i));
		sum=a+b;
		if(sum==c)
		{
			s.insert(p1-1,"+");
			cout<<s;
			return 0;
		}
	}
	cout<<"Impossible!";
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//15-1114-4  hasome
using namespace std;
int main(){
	string s,s1,s2;
	int a=0,b=0,y=0;
	cin>>s;
	s1=s.substr(0,s.find("="));    //取出＝号前后的两个数；
	s2=s.substr(s.find("=")+1);
	y=stoi(s2);          //将等号后面的数转化为整数
	for(int i=0;i<s1.size()-1;i++)
	{
		a=stoi(s.substr(0,i+1));
		b=stoi(s.substr(i+1));
		if(a+b==y){
			cout<<a<<"+"<<b<<"="<<y;
			return 0;
		}
	}
	cout<<"Impossible!"<<endl;
	return 0;
}
#include<bits/stdc++.h>//15-1114-5  jiangyf70
using namespace std;
int main(){
    int a, b, c = 0;
    scanf("%d=%d", &a, &b);
    while(a)
    {
        c =c * 10 + a % 10;
        a /= 10;
        if(a + c == b) break;
    }
    printf("%d+%d=%d", a, c, b);
    return 0;
}
```

```cpp
#include <bits/stdc++.h>//16-1132 -1    javacn
using namespace std;
string s, r;
int main()
{
	getline(cin, s);
	for(int i = 0; i < s.size(); i++)
	{
		if(isdigit(s[i]))    //如果是数字，直接拼上去
		{
			r = r + s[i];
		}
		else
		{
			//连续的非数字，只要把第1个或者最后一个改成*
			if(i == 0 || isdigit(s[i - 1]))
			{
				r = r + '*';
			}
		}
	}
	cout << r;
	return 0;
}
```

保留整数

```
#include<iostream>//16-1132-2  jiangyf70
using namespace std;
int main()
{
    string a, s = "";
    getline(cin, a);
    for(int i = 0; i < a.size(); i++)
    {
        if(a[i] >= '0' && a[i] <= '9')
        {
            s += a[i];
        }
        else if(s[s.size()-1] != '*')
        {
            s += '*';
        }

    }
    cout << s;
    return 0;
}
```

```cpp
#include<bits/stdc++.h>//16-1132-3    jiangyf70
using namespace std;
int main()
{
    string s, s1;
    getline(cin, s);
    for(int i = 0; i < s.size(); i++)
    {
        string s2 = "";
        int j = i;
        while(s[j] < '0' || s[j] > '9')
        {
            s2 = "*";
            j++;
        }
        s1 += s2;
        if(s[j] >= '0' && s[j] <= '9')
            s1 += s[j];
        i = j;
    }
    cout << s1;
    return 0;
}
```

```cpp
#include<bits/stdc++.h>//16-1132-4  hasome
using namespace std;
string s,s1;
int n,c;
int main()
{
	//读入字符串
	getline(cin,s);
	//思路：碰到第一个字符拼接*，碰到数字拼接数字。
	for(int i=0;i<s.size();i++)
	{
		if(isdigit(s[i]))
		{
			s1=s1+s[i];
			c=0;
		}
		else if(c==0)
		{
			s1=s1+'*';
			c++;
		}
	}
	cout<<s1<<endl;
	return 0;
}
```