


CS153

A Mars rover, likely a Curiosity rover, is shown on a rocky, sandy surface under a clear blue sky. The rover is equipped with various scientific instruments and cameras. A long shadow is cast by the rover onto the ground. The entire image has a blue color overlay.

算法
建模

课件下载链接:

<https://pan.baidu.com/s/1htbqTfA>

密码: imfv

作业网站:

<http://120.132.18.213:8080/thrall-web/main#home>

打字测速

打开金山打字通
点击打字测试
课程选择：computer



金山打字通



金山打字通



返回 首页 > 打字测试

英文测试

to pro

to pro

led wi

led wi

man ra

man ra

computer

☐ 限时 分钟

测试成绩：**264字/分钟**

时间：1分27秒 字数：383字 速度：264字/分 正确率：100%

击败了全国 **99.74%** 的人



当前打字水平级别：**九级（共十级）**



打字速度比风快

查看进步曲线



ly fil

ly fil

the hu

the hu

时间 00:01:27

速度 264字/分

进度 100%

正确率 100%



用户反馈：



大数模型 高精度计算

数据类型： 范围

类型	范围	占用内存空间
char	-128~127	1
short	-32768~32767	2
int	-2147483648 ~ +2147483647	4
long long	-9223372036854775808 ~ +9223372036854775807	8
unsigned short	0 ~ 65536	2
unsigned int	0 ~ 4294967295	4
unsigned long long	0~18446744073709551615	8

x+y问题

输入正整数x和y，输出x+y

注意: $x, y \leq 10^{200}$

输入样例

123456789012345678901234567890
999999999999999999999999999999999999

输出样例

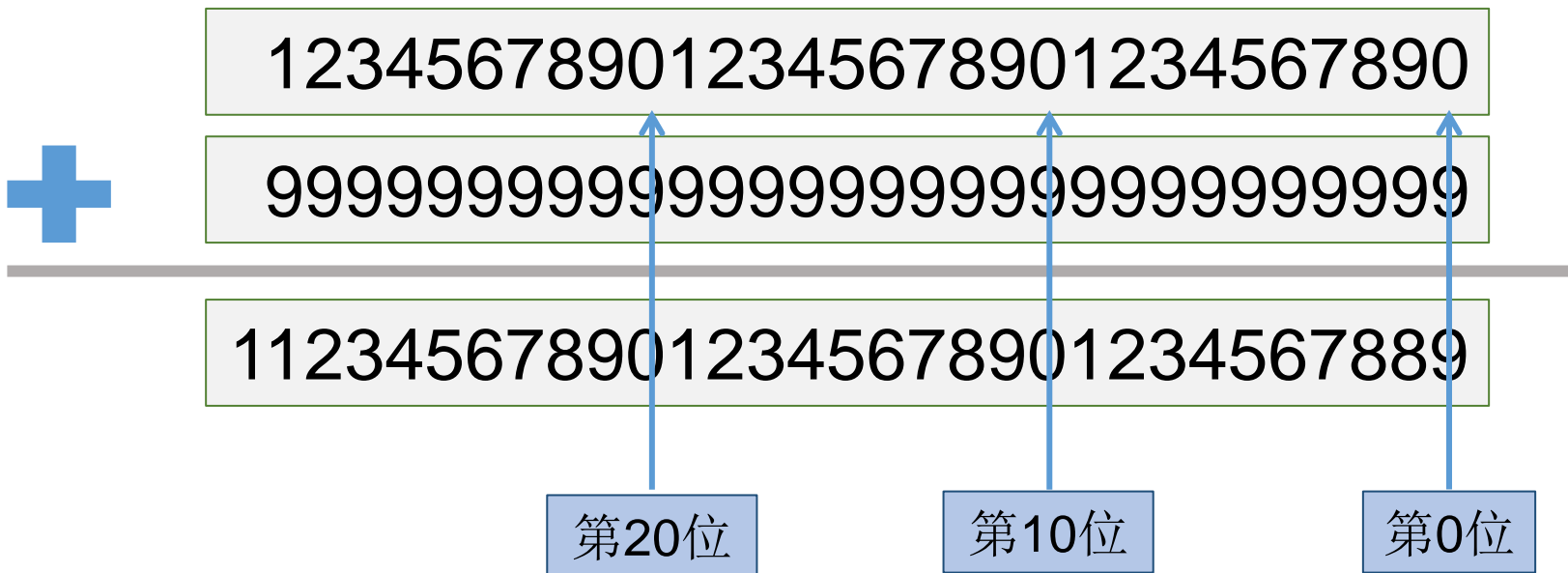
1123456789012345678901234567889

如何存储大数？

大数存储：用数组

```
#define SIZE 200
int x[SIZE],y[SIZE];
```

大数x,y有 200位高精度



大数的个位数 放在数组的第0号下标位置

x+y问题： 算法步骤

```
23 int x[SIZE],y[SIZE];
24 int main() {
25     string s1,s2;
26     cin>>s1>>s2;
27     converts(x,s1);
28     converts(y,s2);
29     add(x,y);
30     print(x);
31     return 0;
32 }
```

定义大数x,y高精度数组

输入字符串s1,s2

将字符串s1,s2
转换成大数x,y
高精度数组

算加法: $x+=y$

输出x

输入 高精度大数

1

先输入字符串变量

2

再把字符串变量
转成高精度数组

字符串转成高精度大数

`int *a` 表明`a`是数组

其实 `a`也是指针
`a`指向的数据是`int`类型

```
5 void converts(int *a, string s){  
6     int i, len=s.size();  
7     for(i=0; i<len; i++) a[i]=s[len-i-1]-'0';  
8     for(; i<SIZE; i++) a[i]=0;  
9 }
```

个位数 放在数组的第0号下标位置

十位数 放在数组的第1号下标位置

。 。 。 。 。 。

空的位置全部用0填充

输出 高精度大数

```
18 void print(int *a){  
19     int i;  
20     for(i=SIZE-1;i>0;i--) if(a[i]>0) break;  
21     for(;i>=0;i--) cout<<a[i]; cout<<endl;  
22 }
```

先找到数组中非0位置的最高位

再逐位输出
直到数组的第0号下标位置
也就是个位数

高精度+高精度

```
10 void add(int *a, int *b, int len=SIZE){ // a+=b
11     // 逐位相加
12     for(int i=0; i<len; i++)
13     {
14         if((a[i]+=b[i])>=10){ // 进位
15             a[i+1]++;
16             a[i]-=10;
17         }
18     }
```

从个位开始，逐位相加

如果当前位置的数 ≥ 10 ，就进位

变种：高精度+int

```
10 void add(int *a, int b){ //a+=b
11     //逐位相加，循环条件：b有剩余 或者 a[i]结果需进位
12     for(int i=0; b || a[i]>=10; b/=10, i++)
13     {
14         if((a[i]+=b%10)>=10) { //进位
15             a[i+1]++;
16             a[i]-=10;
17         }
18     }
19 }
```

挑战

请用**3**分钟时间仔细观察“加法：高精度+高精度”代码

合上电脑，在白纸上用笔完成同一个程序

由同班同学互相找bug

x-y问题

输入正整数x和y，输出x-y

注意： $x, y \leq 10^{200}$

输入样例

666666

88888

输出样例

577778

输入样例

123456789012345678901234567890

99999999999999999999999999999999

输出样例

-876543210987654321098765432109

若 $y > x$ 时该
如何存处理？

如何判断
是否 $y > x$ ？

x-y问题： 算法步骤

```
29 int x[SIZE],y[SIZE];
30 int main() {
31     string s1,s2;
32     cin>>s1>>s2;
33     converts(x,s1);
34     converts(y,s2);
35     if(le(y,x)) {
36         sub(x,y);
37         print(x);
38     }
39     else {
40         sub(y,x);
41         cout<<"-"; print(y);
42     }
43     return 0;
44 }
```

le: less than or equal to

如果 $y \leq x$

算减法: $x -= y$

输出 x

比较大小

```
11 bool le(int *a, int *b, int len=SIZE) {  
12     int i;  
13     for(i=len-1; i>=0&&a[i]==b[i]; i--) ;  
14     return i<0 || a[i]<b[i];  
15 }
```

从最高位开始，逐位比较

两种循环
结束条件

1

i<0: 已查看完所有数字

2

找到了不同数字

减法：高精度-高精度

```
16 //高精度-高精度，相当于a-=b，注意确保a>=b
17 void sub(int *a, int *b, int len=SIZE) {
18     for(int i=0;i<len;i++)
19         if((a[i]-=b[i])<0){//借位
20             a[i+1]--;
21             a[i]+=10;
22         }
23 }
```

从个位（最低位）开始，逐位相减

若发现负数，就需要借位

变种：高精度-int

```
10 //高精度-int, 相当于a-=b, 注意确保a>=b
11 void sub(int *a, int b) {
12     //逐位相减, 循环条件: b有剩余 或者 a[i]结果需借位
13     for(int i=0; b || a[i]<0; b/=10, i++) {
14         if((a[i]-=b%10)<0) { //借位
15             a[i+1]--;
16             a[i]+=10;
17         }
18     }
19 }
```

一组高精度大数

```
#define SIZE 200  
#define N 20  
int f[N][SIZE];
```

定义数组f
里面有20个高精度大数

```
29 int main() {  
30     converts(f[0], "12345678901234567890");  
31     for(int i=1; i<N; i++){  
32         copy(f[i], f[i-1]); ←  
33         add(f[i], 1);  
34     }  
35     for(int i=0; i<N; i++)  
36         print(f[i]);  
37     return 0;  
38 }
```

模块化编程

把完整的程序划分成一块一块的模块

每个模块功能相对独立

高精度大数 综合练习