23.排序！！！！
自定义排序准则(修改cmp)！！！！！！
31.几个容器：
unordered_set:
set：
vector:
queue：
stack:

1.当一个项对应多个分支时，可以考虑变成**数组**的方式，而不是多个if-else或者Switch。（例如：1002的从0-9）

```
char pinyin[][11] = {"ling","yi","er","san","si","wu","liu","qi","ba","jiu"};

cout << pinyin[i] << " ";
```

2.当输入/出一个超级大的数时，可以把它看做一个**字符串数组**，然后根据**字符串与数字的ASCII码转化**来继续操作。（例如：i-48或i-“0”）

3.使用strlen（）时要加头文件

4.C++可以使用字符串string。

```
string n;
string nm[11];
for(int i=0;i<num;i++)
{
n=nm[i];
}
```

5.取某个字符串中的字符进行比较，可以用 **for(auto iter : nter)**

```
        for (auto x : n)   //x代表字符串n中的第i个字符
        {
            if(x=="A") cout<<"true";
        }
```

6.C++中**单引号**和**双引号**代表意义不同！！！

```
‘P’代表一个字符
“P”代表指向字符串的一个指针
```

7.出现统计一个对象有多个属性时，可以用类（公共属性）

```
class stu {
public:
    string name;   //不超过10个字符
    string number;
    int grade;
};        //注意这里的“；”不能忘记
```

8.题目给例如，3n+1,又规定n<=100.则设置的时候要考虑n=100,3n+1超过100的**越界**情况！

9.对于数字**个十百**的计算，有时只要按最大位数算就行，不需要分情况。

```
    //if (n > 100) {
        g = n % 10;
        s = (n / 10) % 10;
        b = (n / 10) / 10;   //这样就行
    //}
    //else if (n > 10) {
    //  s = n / 10;
    //  g = n % 10;
    //}
    //else {
    //  g = n;
    //}
```

10.不超过n！！！

意思是**<=n**.

11. 对于素数的问题：

```
1.判断的时候，合数一定是一个小的*一个大的，所以当小的和大的接近时，就无限趋向于 根号n。
2.所以优化思想就是 从2到根号n 循环就行。
int sushu(int a) {//判断素数 1是0非
    for (int i =2; i <= sqrt(a); i++) { //一定是小于等于
        if (a % i == 0) {
            return 0;
        }
    }
    return 1;
}
//2是最小的素数
```

12.数学公式会用到头文件math。

```
#include<cmath>    //注意这里有个C
```

13.for循环

```
for (int i = N - M - j-1; i > 0; i= i - M) {  //这里的第三个条件是一条语句，i++实际是i=i+1
            num[i + M] = num[i];
        }
```

14.循环右移问题：

如果 **右移位数**m>**数组长度**n，就得考虑**m=m%n**，来作为新的移动位数，优化代码。

15.对字符串的输入：

```
//普通输入
string a;
cin>>a;
//如果字符串含空格 例如:Hello World Here I Come
#include<string>
string a;
getline(cin,a);
```

16.C++循环输入

```
1.不需要让循环停止的情况
int a，b;
while(cin>>a>>b){
//语句
}
2.可能需要判断循环的停止
int a，b;
char c;
while(cin>>a>>b){
c=getchar();
//语句
if(c=='\n')  //如果有回车，则跳出循环
    break;
}
```

17.整形，长整型的问题

```
int  //32位
long int  //32位或64位  尽量不用这个
long long int //64位
```

18.数组全部附为0

```
int a[5]{ 0 };
```

19.保留n位小数的输出

```
        float a3, f3;
        a3 = a[3]; f3 = f[3];
        printf("%.2f", (a3 / f3)); //.2f保留2位小数
```

20.计算时强制类型转换，只需要转一个就行

```
2/3 == 0
转：(double)2 / 3 == 0.666...
或：2 / (double) 3 == 0.666...
```

21.对于数字运算最后一位的舍入

```
//int ,long int ,long long int
int x;    //x定义为整形
double m;  //接受一个浮点数m  例如45.236、 326.255、 78.891
x=m+0.5;   //加0.5就可以表示舍入后的整数部分！！！
//x=m*100+0.5;
```

22.把 整形int 转成 字符型（进行ASCII码之间转化！！！）

```
char a;
int m=5;
a=m+‘0’ // 整数+0就可以表示相应字符的数字了
```

23.排序！！！！

在C++中使用sort()函数需要使用`#include<algorithm>`头文件。

```
//用法
sort(begin, end, cmp)
//begin为指向待sort()的数组的第一个元素的指针，
//end为指向待sort()的数组的最后一个元素的下一个位置的指针
//cmp参数为排序准则，cmp参数可以不写，如果不写的话，默认从小到大进行排序。
 可以用a.begin()  a.end()函数
```

自定义排序准则(修改cmp)！！！！！！

```
链接：
https://blog.csdn.net/qq_41575507/article/details/105936466
```

24.**insert（）函数的使用**

```
1.在字符串某个位置加入字符串
string n="hello";
//string m;
n.insert(2,"kk");//在n的下标为2的位置插入字符串‘kk’
n.insert(2, m)//在n的下标为2的位置插入字符串m

2.
    string str1="hello";
    char c='w';
    str1.insert(4,5,c);//在原串下标为4的字符o前插入5个字符c

3.
    string str2="hello";
    string s2="weakhaha";
    str2.insert(0,s2,1,3);//将字符串s2从下标为1的e开始数3个字符，分别是eak，插入原串的下标为0的字符h前
```

25. reverse（）函数

```
//反转string
string N;
cin>>N;
reverse(N.begin(), N.end());//begin，和end；
```

```
//反转字符数组
char s[101];
cin.getline(s,sizeof(s));       //也可以不用cin.getline
int m=strlen(s);
reverse(s,s+m);
puts(s);

//反转整型数组
int a[100];
reverse(a,a+10);
```

26.为避免栈溢出，用new来实现

```
  int *data = new int[100005];
  int  *next = new int[100005]; //这两个用了new,不然运行出现栈溢出。
  int list[100005]; //数组list用来按顺序存放地址

  记得delete（data）；
```

27.反转链表（数组方式反转）

//做一个有顺序的链，链里面放节点的地址，那么链的顺序就是链接的顺序。

28.一种输入输出格式

```
printf("%05d %d -1", list[sum - 1], data[list[sum - 1]]);
//输出5位 向右对齐  不足5位前面补0
scanf_s("%d/%d/%d",&p[i].yy, &p[i].mm, &p[i].dd);
//含‘/’号等
```

29.获取最大公约数（递归的方法）

```
long long max(long long a, long long b) {       //获取最大公约数
    if (a < 0)a = -a;
    if (b < 0)b = -b;
    return b == 0 ? a : max(b, a % b);
};
```

30.C++类的构造方法

```
class Stu{   //首字母大写
    private:
        string name;
        int num;
    public:
        Stu(string a,int b){       //没有返回类型，函数名与类名相同
            name=a;  num=b;
        }
}
```

31.几个容器：

unordered_set:

无序集 //是基于哈希表存储，顺序是随机的（但也不重复）。其他与set类似

```
//头文件<unordered_set>
//空间换时间，比set快，但要求的存储空间更大。
```

set:

有序集 //会自动排序，自动不含重复元素

```
//头文件<set>
set<s> mp;
添加元素：a.insert(s);   // 对s操作，插入或删除时，可能不在最后位置
删除元素：a.erase(s);
查找元素：a.find(s);   //如果找到则返回位置，否则返回a.end()的位置      ！！！是s.end()不是s.size()
                        例如：if(st.find(9)!=st.end())     //用此类方法查元素是否在
                                cout<<"9 is present\n";
                            else
                                cout<<"9 is not present\n";
清空：   a.clear();
判断是否非空：  a.empty();
输出：
        for (auto it = n.begin(); it != n.end(); ++it) {
        cout << *it << endl;
```

vector:

一般的存储**容器**

```
//头文件<vector>
添加元素：a.push_back(); //  在最后进行操作
删除元素：a.pop_back();  //  类似于压栈和出栈
判空：  a.empty();
元素个数：a.size();
```

queue:

//**队列** ； 先进先出； 只允许访问队头和队尾元素！！！

```
//头文件<queue>
元素个数：a.size();  //同上
队尾加入一个元素：a.push(k);
队头取出一个元素：a.pop();
访问队头：a.front()
访问队尾：a.back()
```

stack:

栈//后进先出； 只允许访问栈顶元素

```
//头文件<stack>
压栈：a.push(k);
出栈：a.pop();
查看栈顶元素：a.top();
```