1.将一个整数转换为字符串 void toString(char \*s, int num);

2. 将一个仅含有数字的字符串转换为整数 int str2Int(const char \*s);

3. 将一个含有数字和小数点的字符串转换为浮点型数据 double str2lf(const char \*s);

4. 求两个正整数的最小公倍数 int lcm(int m, int n);

5. 求两个正整数的最大公约数 int gcd(int m, int n);

6. 判断一个正整数是否为素数 bool isPrime(int num); 或 int isPrime(int num);

7. 判断一个整数是几位数 int digits(int num);

8. 将一个正整数转换为十六进制表示的字符串 void int2Hex(char \*hex, int num);

9. 将一个十六进制表示的字符串转换为十进制整数 int hex2Int(char \*hex);

10. 将一个十进制整数按进制表示法输出（0所在项不输出）

//如：输入 1204 输出 1204=1\*10^3+2\*10^2+4 void int2str(char \*s, int num);

//其他各种形式的输出 1024=1\*1000+2\*10+4\*1

11. 求一个整数所有数字和 int digitsSum(int num);

12. 求一个整数数字逆序后的整数，如输入：1204 输出：4021

13. 求两个整数的所有公因子，返回公因子的个数。int factors(int \*fts, int m, int n);

14. 求整数的所有因子和（不含1和它本身）int factorsSum(int m);

15. 输出一个正整数中包含数字k的个数(k为0~9中的任一数字)

// int numbers(int num, int k);

16.求字符串的长度。 int length(char \*s);

17. 将字符串s2复制到一个字符串s1中。 char \*stringcopy(char \*s1, const char \*s2);

18. 将字符串s2连接到字符串s1的后面。 char \*stringcat(char \*s1, const char \*s2);

19. 将字符串s中的所有大写字母转换为小写字母。 char \*stolwr(char \*s);

20. 将字符串s中的所有小写字母转换为大写字母。 char \*stoupr(char \*s);

21. 比较两个字符串s1和s2的大小，返回两个字符串中第一个不同字符的ASCII码的差，两个字符串相同，返回0，如："abcd"和"dddd"的结果为-3。

// int stringcmp(char \*s1, char \*s2);

22. 求字符串s中从pos开始，长度为len的子串。

//char \*substr(const char \*s, char \*subs, int pos, int len);

23. 求某字符在字符串中的位置，返回第一次出现的位置，不存在返回-1

24. 求字符串s中第一次出现字符串ss的位置， 如果不存在返回-1。

//int strAt(const char \*s, char \*ss);

25. 求字符串s中元音字母(不区分大小写)的个数。 int vowels(char \*s);

26. 求字符串s中出现字符c的个数。 int chars(const char \*s, char c);

27. 求一个字符串中包含的所有整数，并返回整数的个数。

// 如"123ab324xy3cumtb24xyz 17a"中包含5个整数，分别为123, 324, 3, 24, 17。

// int integers(const char \*s, int nums[]);

28. 将一个字符c插入到字符串s指定的位置上。 char \*insertchar(char \*s, char c, int pos);

29. 将一个字符串s2插入到字符串s1指定的位置上。

//char \*insertstr(char \*s1, char \*s2, int pos);

30.将一个字符串s逆置，如"abcd" 逆置后为"dcba"。 char \*reverse(char \*s);

31. 将一个字符串s在指定的位置pos分隔成两个字符串s1和s2。

void split(const char \*s, char \*s1, char \*s2, int pos);

32. 求一个字符串s中包含多少个单词（以空格分隔，可能有多个）。

//如"I Love C Language"，包含4个单词。 int words(char \*s);

//此题方法可用于求一个字符串s中包含多少个整数

33. 将一个字符串s中的所有字符old替换为另一个字符new。

//char \*replace(char \*s, char old, char new);

34. 二分查找：在一个长度为n的有序数组p中，查找元素x，返回其所在的位置，如果该元素不存在，返回-1。 int binary\_search(int \*p, int n, int x);

35. 在一个长度为n的数组p中，查找元素x第一次出现的位置，如果该元素不存在，返回-1。

36. 求字符串s中第一次出现元音字母的位置（不区分大小写），如果不存在返回-1。

// int find\_first\_vowel(const char \*s);

37. 求n个字符串的最长公共前缀，若不存在则返回空字符串。

// "flower","flow","flight"的最长公共前缀为“fl”，假设字符串的长度不超过20

//char\* lcs(char a[][21], char\* lcstring, int n);

//方法：先对所有字符串排序，再求第一个字符串和最后一个字符串的公共前缀即可

//此题可扩展为：求n个字符串的最长公共后缀

38. 在一个长度为n的有序数组p中插入元素x，使得该数组仍然有序。实现过程中不能使用任何排序算法。

//39. 判断一个字符串是否为回文串

//回文串是一个正读和反读都一样的字符串，比如“level”或者“noon”等等就是回文串。

//方法：判断该字符串是否和逆置后的字符串相同即可

//此题可扩展，求最大回文子串的长度，求所有的最大回文子串等

//此题可扩展为求回文数的问题

40 查找n个字符串中，是否存在字符串s，若存在返回第一次出现的位置，否则返回-1.

//若存在返回1，不存在返回0，稍加修改即可

41. 用冒泡法对一长度为n的整型数组p排序。

42. 用选择法对一长度为n的整型数组p排序。

43. 使用插入排序法对一长度为n的整型数组a排序

44. 将长度分别为m和n的有序整型数组p和q，合并到整型数组dest中，使得dest仍然有序。 //void merge(int \*p, int \*q, int\* dest, int m, int n);

45. 冒泡法对n个字符串排序。

46. 选择法对n个字符串排序。

47. 插入排序法对n个字符串排序。

48. 统计一个字符串中出现不同字符的个数，如字符串"ababc3x"中含有5个不同字符。

49. 写一个函数判断一个年份是否为闰年

50. 统计两个年份之间闰年的个数

扩展题目：

1. 括号匹配深度

一个合法的括号匹配序列有以下定义:

1. 空串""是一个合法的括号匹配序列
2. 如果"X"和"Y"都是合法的括号匹配序列,"XY"也是一个合法的括号匹配序列
3. 如果"X"是一个合法的括号匹配序列,那么"(X)"也是一个合法的括号匹配序列
4. 每个合法的括号序列都可以由以上规则生成。

例如: "","()","()()","((()))"都是合法的括号序列  
对于一个合法的括号序列我们又有以下定义它的深度:

1. 空串""的深度是0
2. 如果字符串"X"的深度是x,字符串"Y"的深度是y,那么字符串"XY"的深度为max(x,y)
3. 如果"X"的深度是x,那么字符串"(X)"的深度是x+1

例如: "()()()"的深度是1,"((()))"的深度是3。

现在给你一个合法的括号序列,需要你计算出其深度。

如：输入:(()) 输出:2

2. 求水仙花数，所谓 "水仙花数 "是指一个三位数，其各位数字立方和等于该数本身。例如：153是一个 "水仙花数 "，因为153=1的三次方＋5的三次方＋3的三次方。

//完全平方数、完全立方数等

3. 输入某年某月某日，判断这一天是这一年的第几天？

4. 分数数列求和问题

5. 统计整数的二进制表示中1的个数

6. 统计n(包含n)以内的自然数中1出现的次数

7. 矩阵的鞍点问题

//矩阵的运算、一次方程组的求解

8. 和为n的连续正整数序列的个数

//例如输入15，由于1+2+3+4+5=4+5+6=7+8=15，所以输出3

9. 删除字符串中的特定字符(可以是多个)：输入两个字符串，从第一个字符串中删除第二个字符串中的所有字符

//如从一个字符串:I Love China!中删除aeiou

10. 娱乐游戏中的概率问题：抽奖问题、骰子问题、扑克牌问题等