第一题：

[要求]

给定一个整数数组和一个整数 k，判断数组中是否存在两个不同的索引 i 和 j，使得 nums [i] = nums [j]，并且i 和 j的差的绝对值至多为k。

[接口]

CPP:bool ContainsNearbyDuplicate(vector<int>& nums, int k)

C:bool ContainsNearbyDuplicate(int\* nums, int numsSize, int k)

[示例]

Array = 1,2,3,1

Key = 3

Output = true

--------------------------------------------------------------------------------------------

第二题：

[要求]

给定一个整数，返回它在 Excel 表中相对应的列名称，如果在Excel表中找不到该列，则返回""。

例如，

1 -> A

2 -> B

3 -> C

...

26 -> Z

27 -> AA

28 -> AB

...

[接口]

CPP:string ExcelSheetColumnTitle(int n)

C:char \* ExcelSheetColumnTitle(int n)

[示例]

Input = 1

Output = "A"

--------------------------------------------------------------------------------------------

第三题：

[要求]

编写一个程序判断给定的数是否为丑数。丑数就是只包含质因数 2, 3, 5 的正整数。

说明：

1. 1 是丑数。

2.输入不会超过 32 位有符号整数的范围: [−（2的31次方）, 2的31次方 − 1]。

[接口]

CPP:bool IsUgly(int num)

C:bool IsUgly(int num)

[示例]

Input = 6

Output = true

--------------------------------------------------------------------------------------------

第四题：

[要求]

判断一个整数是否是回文数。回文数是指正序（从左向右）和倒序（从右向左）读都是一样的整数。

[提高]

进阶:你能不将整数转为字符串来解决这个问题吗？

[接口]

CPP:bool IsPalindrome(int x)

C:bool IsPalindrome(int x)

[示例]

Input = 121

Output = true

--------------------------------------------------------------------------------------------

第五题：

[要求]

给定一个二叉树，找出其最小深度。

最小深度是从根节点到最近叶子节点的最短路径上的节点数量。

[接口]

CPP:int MinDepth(TreeNode\* root)

C:int MinDepth(struct TreeNode\* root)

struct TreeNode

{

int val;

struct TreeNode \*left;

struct TreeNode \*right;

};

[示例]

3

/ \

9 20

/ \

15 7

（这个图在测试样例中不需要画，只要下面的输入输出即可）

Input = 3,9,20,null,null,15,7

Output = 2

--------------------------------------------------------------------------------------------

第六题：

[要求]

给定一个整数数组，判断是否存在重复元素。如果任意一值在数组中出现至少两次，函数返回true 。如果数组中每个元素都不相同，则返回false 。

[接口]

CPP:bool ContainsDuplicate(vector<int>& nums)

C:bool ContainsDuplicate(int\* nums, int numsSize)

[示例]

Input = [1,2,3,1]

Output = true

--------------------------------------------------------------------------------------------

第七题：

[要求]

给定一个二叉树，找出其最大深度。

二叉树的深度为根节点到最远叶子节点的最长路径上的节点数。

[接口]

CPP:int MaxDepth(TreeNode\* root)

C:int MaxDepth(struct TreeNode\* root)

[示例]

3

/ \

9 20

/ \

15 7

Input = 3,9,20,null,null,15,7

Output = 3

--------------------------------------------------------------------------------------------

第八题：

[要求]

编写一个函数，输入是一个无符号整数，返回其二进制表达式中数字位数为‘1’的个数（也被称为汉明重量）。

[提高]

如果多次调用这个函数，你将如何优化你的算法

[接口]

CPP:int HammingWeight(unsigned int n)

C:int HammingWeight(uint32\_t n)

[示例]

Input = 3

Ouput = 2

--------------------------------------------------------------------------------------------

第九题：

[要求]

给你两个二进制字符串，返回它们的和（用二进制表示）。

输入为非空字符串且只包含数字 1 和 0。

提示：每个字符串仅由字符 '0' 或 '1' 组成。

1 <= a.length, b.length <= 10^4

字符串如果不是 "0" ，就都不含前导零。

[接口]

CPP:string AddBinary(string a, string b)

C:char \* AddBinary(char \* a, char \* b)

[示例]

Input1 = "11"

Input2 = "1"

Output = "100"

--------------------------------------------------------------------------------------------

第十题：

[要求]

给定一个二叉树，返回所有从根节点到叶子节点的路径。

struct TreeNode {

int val;

struct TreeNode \*left;

struct TreeNode \*right;

};

[接口]

CPP:vector<string> BinaryTreePaths(TreeNode\* root)

C:char \*\* BinaryTreePaths(struct TreeNode\* root, int\* returnSize)

[示例]

1

/ \

2 3

\

5

Input = 1,2,3,null,5

Output = "1->2->5", "1->3"

--------------------------------------------------------------------------------------------

第十一题：

[要求]

你和你的朋友，两个人一起玩 Nim 游戏：桌子上有一堆石头，每次你们轮流拿掉1-3块石头。 拿掉最后一块石头的人就是获胜者。你作为先手。

你们是聪明人，每一步都是最优解。 编写一个函数，来判断你是否可以在给定石头数量的情况下赢得游戏。

[接口]

CPP:bool CanWinNim(int n)

C:bool CanWinNim(int n)

[示例]

Input = 4

Output = false

--------------------------------------------------------------------------------------------

第十二题：

[要求]

给定一个只包括 '('，')'，'{'，'}'，'['，']' 的字符串，判断字符串是否有效。

有效字符串需满足：

左括号必须用相同类型的右括号闭合。

左括号必须以正确的顺序闭合。

注意空字符串可被认为是有效字符串。

[接口]

CPP:bool IsValid(string s)

C:bool IsValid(char \* s)

[示例]

Input = "()"

Output = true

--------------------------------------------------------------------------------------------

第十三题：

[要求]

请你来实现一个 atoi 函数，使其能将字符串转换成整数。

首先，该函数会根据需要丢弃无用的开头空格字符，直到寻找到第一个非空格的字符为止。接下来的转化规则如下：

如果第一个非空字符为正或者负号时，则将该符号与之后面尽可能多的连续数字字符组合起来，形成一个有符号整数。

假如第一个非空字符是数字，则直接将其与之后连续的数字字符组合起来，形成一个整数。

该字符串在有效的整数部分之后也可能会存在多余的字符，那么这些字符可以被忽略，它们对函数不应该造成影响。

提示：本题中的空白字符只包括空格字符 ' ' 。

假设我们的环境只能存储 32 位大小的有符号整数，那么其数值范围为 [−231, 231 − 1]。如果数值超过这个范围，请返回 INT\_MAX (231 − 1) 或 INT\_MIN (−231) 。

[说明]

假如该字符串中的第一个非空格字符不是一个有效整数字符、字符串为空或字符串仅包含空白字符时，则你的函数不需要进行转换，即无法进行有效转换。

在任何情况下，若函数不能进行有效的转换时，请返回 0 。

[示例]

示例 1:

输入: "42"

输出: 42

示例 2:

输入: " -42"

输出: -42

解释: 第一个非空白字符为 '-', 它是一个负号。

我们尽可能将负号与后面所有连续出现的数字组合起来，最后得到 -42 。

示例 3:

输入: "4193 with words"

输出: 4193

解释: 转换截止于数字 '3' ，因为它的下一个字符不为数字。

示例 4:

输入: "words and 987"

输出: 0

解释: 第一个非空字符是 'w', 但它不是数字或正、负号。

因此无法执行有效的转换。

示例 5:

输入: "-91283472332"

输出: -2147483648

解释: 数字 "-91283472332" 超过 32 位有符号整数范围。

因此返回 INT\_MIN (−231) 。

[测试样例]

"42"

[接口]

CPP: int MyAtoi(string s)

C:int MyAtoi(char \* s)

--------------------------------------------------------------------------------------------

第十四题：

[要求]

给定一个非空整数数组，除了某个元素只出现一次以外，其余每个元素均出现两次。找出那个只出现了一次的元素。

[提高]

你的算法应该具有线性时间复杂度。 你可以不使用额外空间来实现吗？

[等级]

简单

[接口]

CPP:int SingleNumber(vector<int>& nums)

C:int SingleNumber(int\* nums, int numsSize)

[示例]

Input = 2,2,1

Output = 1

--------------------------------------------------------------------------------------------

第十五题：

[要求]

给定一种规律pattern和一个字符串str，判断str是否遵循相同的规律。

这里的遵循指完全匹配，例如：pattern 里的每个字母和字符串 str 中的每个非空单词之间存在着双向连接的对应规律。

[等级]

简单

[接口]

CPP:bool WordPattern(string pattern, string s)

C:bool WordPattern(char \* pattern, char \* s)

[示例]

Pattern = "abba"

Str = "dog cat cat dog"

Output = true

Pattern = "abba"

Str = "dog cat cat fish"

Output = false

--------------------------------------------------------------------------------------------

第十六题：

[要求]

颠倒给定的 32 位无符号整数的二进制位。

[提高]

如果多次调用这个函数，你将如何优化你的算法？

[等级]

简单

[接口]

CPP: uint32\_t ReverseBits(uint32\_t n)

C:uint32\_t ReverseBits(uint32\_t n)

[示例]

Input = 43261596

Output = 964176192

--------------------------------------------------------------------------------------------

第十七题：

[要求]

给定一个非空字符串s和一个包含非空单词列表的字典wordDict，判定s是否可以被空格拆分为一个或多个在字典中出现的单词

[接口]

CPP:bool WordBreak(string s, vector<string>& wordDict)

C:bool WordBreak(char \* s, char \*\* wordDict, int wordDictSize)

[示例]

Input1 = "cppcode"

Input2 = cpp, code

Output = true

--------------------------------------------------------------------------------------------

第十八题：

[要求]

给定一个由整数组成的非空数组所表示的非负整数，在该数的基础上加一。

最高位数字存放在数组的首位， 数组中每个元素只存储单个数字。

你可以假设除了整数 0 之外，这个整数不会以零开头。

[接口]

CPP:vector<int> PlusOne(vector<int>& digits)

C:int\* PlusOne(int\* digits, int digitsSize, int\* returnSize)

[示例]

Input = 1,2,3

Output = 1,2,4

--------------------------------------------------------------------------------------------

第十九题：

[要求]

实现 int sqrt(int x) 函数，计算并返回 x 的平方根，其中 x 是非负整数。由于返回类型是整数，结果只保留整数的部分，小数部分将被舍去。

[提高]

要求不使用现成的函数

[接口]

CPP:int MySqrt(int x)

C:int MySqrt(int x)

[示例]

Input = 8

Output = 2

--------------------------------------------------------------------------------------------

第二十题：

[要求]

给定一个数组 nums，编写一个函数将所有 0 移动到数组的末尾，同时保持非零元素的相对顺序。

1. 必须在原数组上操作，不能拷贝额外的数组。

2. 尽量减少操作次数。

[等级]

简单

[接口]

CPP:void MoveZeroes(vector<int>& nums)

C:void MoveZeroes(int\* nums, int numsSize)

[示例]

Input = 0,1,0,3,12

Output = 1,3,12,0,0

--------------------------------------------------------------------------------------------

第二十一题:

[要求]

给出一个32位的有符号整数，你需要将这个整数中每位上的数字进行反转。

假设我们的环境只能存储得下 32 位的有符号整数，则其数值范围为 [−231, 231 − 1]。请根据这个假设，如果反转后整数溢出那么就返回 0。

[等级]

简单

[接口]

CPP: int Reverse(int x)

C:int Reverse(int x)

[示例]

Input = 123

Output = 321

--------------------------------------------------------------------------------------------

第二十二题：

[要求]

给定一个数组，将数组中的元素向右移动k个位置，其中k是非负数。

[提高]

尽可能想出更多的解决方案，至少有三种不同的方法可以解决这个问题。

要求使用空间复杂度为 O(1) 的 原地 算法。

[等级]

简单

[接口]

CPP:void Rotate(vector<int>& nums, int k)

C:void Rotate(int\* nums, int numsSize, int k)

[示例]

Input = 1,2,3,4,5,6,7

k = 3

Output = 5,6,7,1,2,3,4

--------------------------------------------------------------------------------------------

第二十三题：

[要求]

给定一个只包含数字的字符串，复原它并返回所有可能的IP地址格式。

有效的IP地址正好由四个整数（每个整数位于0到255之间组成），整数之间用 '.' 分隔。

[接口]

CPP:vector<string> RestoreIpAddresses(string s)

C:char \*\* RestoreIpAddresses(char \* s, int\* returnSize)

[示例]

Input = 25525511135

Output = "255.255.11.135", "255.255.111.35"

--------------------------------------------------------------------------------------------

第二十四题：

[要求]

给定一个字符串，请你找出其中不含有重复字符的最长子串的长度。

[接口]

CPP:int LengthOfLongestSubstring(string s)

C:int LengthOfLongestSubstring(char \* s)

[示例]

Input = abcabcbb

Output = 3

;说明：无重复字符的最长子串是 "abc"

Input = pwwkew

Output = 3

;说明：无重复字符的最长子串是 "wke"

--------------------------------------------------------------------------------------------

第二十五题：

[要求]

给定一个链表和一个特定值 x，对链表进行分隔，使得所有小于 x 的节点都在大于或等于 x 的节点之前。你应当保留两个分区中每个节点的初始相对位置。

[接口]

CPP:ListNode\* Partition(ListNode\* head, int x)

struct ListNode

{

int val;

ListNode \*next;

ListNode(int x) : val(x), next(NULL) {}

};

C:struct ListNode\* Partition(struct ListNode\* head, int x)

struct ListNode

{

int val;

struct ListNode \*next;

};

[示例]

Input1 = 1,4,3,2,5,2

Input2 = 3

Output = 1,2,2,4,3,5

--------------------------------------------------------------------------------------------