1、回文

#### 分析

暴力枚举一下check回文,可以确定出最后答案的一半,就可以得到答案了。

#### 参考代码

1. #include <bits/stdc++.h>
2. using namespace std;
3. bool isPalindrome(string s) {
4. int i = 0, j = s.size() - 1;
5. while(i < j)
6. if(s[i++] != s[j--])
7. return false;
8. return true;
9. }
10. string s;
11. int main() {
12. cin >> s;
13. string add;
14. int i, j;
15. for(i = 0; i < s.size(); i++) {
16. if(isPalindrome(s.substr(i))) break;
17. }
18. for(int j = i - 1; j >= 0; j--) add += s[j];
19. s += add;
20. cout << s.size() << endl;
21. }

2、两个子串

#### 分析

暴力枚举计算字符串前缀后缀相等的最长长度,然后拼接一下就是结果。

#### 参考代码

1. #include <bits/stdc++.h>
2. using namespace std;
3. string s;
4. int main() {
5. cin >> s;
6. int l = s.size(), m = 0;
7. for(int i = 1; i < l; i++) {
8. if(s.substr(0, i) == s.substr(l - i, i))
9. m = i;
10. }
11. cout << s.substr(0, l - m) + s << endl;
12. return 0;
13. }

3、括号匹配方案

#### 分析

这个题字符串长度不长,可以直接爆搜。

有一种代码比较简单的解法,挨着累计'('的个数,遇到')'就完成一次匹配,把情况数乘进答案。本质是把题目所说的移除操作做了一个等价的变化。

#### 参考代码

1. #include <bits/stdc++.h>
2. using namespace std;
3. string s;
4. int main() {
5. cin >> s;
6. int ans = 1, cnt = 0;
7. for(int i = 0; i < s.size(); i++) {
8. if(s[i] == '(') {
9. cnt++;
10. } else {
11. ans \*= cnt;
12. cnt--;
13. }
14. }
15. cout << ans << endl;
16. }

4、疯狂的序列

#### 分析

范围比较大,考虑二分答案,等差数列求和来check。

#### 参考代码

1. #include <bits/stdc++.h>
2. using namespace std;
3. long long n;
4. int main() {
5. cin >> n;
6. long long l = 1, r = 2000000000LL, mid;
7. while(l < r) {
8. mid = (l + r) / 2;
9. if(mid \* (mid + 1) / 2 >= n)
10. r = mid;
11. else
12. l = mid + 1;
13. }
14. cout << l << endl;
15. return 0;
16. }

5、神奇数

#### 分析

注意到r - l不会特别大,直接枚举范围内的数,然后暴力check。

暴力check的时候注意到所有数位和最大只有81,所以会比较快。

#### 参考代码一

1. #include <bits/stdc++.h>
2. using namespace std;
3. bool check(int n) {
4. char s[11];
5. int cur = 0, t = 0;
6. while(n > 0) {
7. s[cur] = n % 10;
8. t += s[cur++];
9. n /= 10;
10. }
11. if(t % 2) return false;
12. t /= 2;
13. bool ok[42] = {0};
14. ok[s[0]] = true;
15. for(int i = 1; i < cur; i++) {
16. int v = s[i];
17. for(int j = 41; j >= 0; j--) {
18. if(ok[j] && j + v <= 41) {
19. ok[j + v] = true;
20. }
21. }
22. if(ok[t]) {
23. return true;
24. }
25. }
26. return false;
27. }
28. int l, r;
29. int main() {
30. int res = 0;
31. cin >> l >> r;
32. for(int i = l; i <= r; i++) {
33. if(check(i)) res++;
34. }
35. cout << res << endl;
36. return 0;
37. }

#### 参考代码二

1. import java.util.HashSet;
2. import java.util.Scanner;
3. import java.util.Set;
4. public class Main {
5. public static boolean check(int n){
6. int[] s = new int[11];
7. int cur = 0, t = 0;
8. while(n > 0) {
9. s[cur] = n % 10;
10. t += s[cur++];
11. n /= 10;
12. }
13. if( t % 2 == 1 ) return false;
14. t /= 2;
15. boolean[] ok = new boolean[42];
16. ok[s[0]] = true;
17. for(int i = 1; i < cur; i++) {
18. int v = s[i];
19. for(int j = 41; j >= 0; j--) {
20. if(ok[j] && j + v <= 41) {
21. ok[j + v] = true;
22. }
23. }
24. if(ok[t]) {
25. return true;
26. }
27. }
28. return false;
29. }
30. public static int max(int a, int b){
31. return (a>b) ? a : b;
32. }
33. public static void main(String[] args) {
34. Scanner in = new Scanner(System.in);
35. int res = 0;
36. int l = in.nextInt();
37. int r = in.nextInt();
38. for(int i = l; i <= r; i++) {
39. if(check(i)) res++;
40. }
41. System.out.println(res);
42. }
43. }

6、求幂

#### 分析

裸暴力好像只能拿20%的分

我们考虑去枚举n范围内的所有i,然后处理出i的幂那些数。

考虑对于i ^ x, 我们需要计算满足 (i ^ x) ^ c = (i ^ y) ^ d的数量,其中i ^ x, i ^ y <= n. 这些我们可以通过预处理出来。

然后对于(i ^ x) ^ c = (i ^ y) ^ d 其实意味着x c = y d, 意味着(x / y) = (d / c), 其中x, y我们可以在预处理之后枚举出来,于是我们就可以借此计算出n范围内有多少不同这种c和d去满足等式。

其实就等于 n / max(x / gcd(x, y), y / gcd(x, y)),然后都累加进答案。gcd()表示最大公约数。

中间可能产生重复枚举,我们用一个set或者hash容器标记一下就好。

以上枚举对于2~sqrt(n)。最后对于大于sqrt(n)的部分,每个的贡献都是n。

#### 参考代码一

1. #include <bits/stdc++.h>
2. using namespace std;
3. const int mod = 1e9 + 7;
4. set<int> S;
5. int n;
6. int main() {
7. cin >> n;
8. long long res = 1LL \* n \* n % mod;
9. for(int i = 2; i \* i <= n; i++) {
10. if(S.find(i) != S.end()) continue;
11. long long tmp = i;
12. int cnt = 0;
13. while(tmp <= n) {
14. S.insert(tmp);
15. tmp = tmp \* i;
16. cnt++;
17. }
18. for(int x = 1; x <= cnt; x++) {
19. for(int y = 1; y <= cnt; y++) {
20. int g = \_\_gcd(x, y);
21. int tmpx = x / g;
22. int tmpy = y / g;
23. res = (res + n / max(tmpx, tmpy)) % mod;
24. }
25. }
26. }
27. res += 1LL \* (n - S.size() - 1) \* n;
28. res %= mod;
29. cout << res << endl;
30. }

#### 参考代码二

1. import java.util.HashSet;
2. import java.util.Scanner;
3. import java.util.Set;
4. public class Main {
5. public final static long MOD = 1000000000 + 7;
6. public static int max(int a, int b){
7. return (a>b) ? a : b;
8. }
9. public static long gcd(long a,long b){
10. return (a % b == 0) ? b : gcd(b,a%b);
11. }
12. public static void main(String[] args) {
13. Scanner in = new Scanner(System.in);
14. long n = in.nextInt();
15. long ans = (long)1\*n\*(n\*2-1) % MOD;
16. Set<Integer> set = new HashSet<>();
17. for (int i = 2; i\*i <= n; i++){
18. if ( set.contains(i)) continue;
19. long tmp = i;
20. int cnt = 0;
21. while(tmp <= n) {
22. set.add((int)tmp);
23. tmp = tmp \* i;
24. cnt++;
25. }
26. for(int k = 1; k <= cnt; k++) {
27. for(int j = k + 1; j <= cnt; j++) {
28. ans = (ans + n / (j / gcd(k, j) ) \* (long)2 ) % MOD;
29. }
30. }
31. }
32. System.out.println(ans);
33. }
34. }

7、购物车

1. function add(items) {
2. var tbody = document.getElementById('jsTrolley').getElementsByTagName('tbody')[0];
3. (items || []).forEach(function (item) {
4. var tr = document.createElement('tr');
5. tr.innerHTML = '<td>' + item.name
7. + '</td><td>' + item.price.toFixed(2) + '</td><td><a href="javascript:void(0);">删除</a></td>';
8. tbody.appendChild(tr);
9. });
10. update();
11. }
12. function bind() {
13. var table = document.getElementById('jsTrolley');
14. table.addEventListener('click', function (event) {
15. var el = event.target;
16. if (el.tagName.toLowerCase() === 'a') {
17. tr = el.parentNode.parentNode;
18. tr.parentNode.removeChild(tr);
19. update();
20. }
21. });
22. }
23. function update() {
24. var table = document.getElementById('jsTrolley');
25. var tbody = table.getElementsByTagName('tbody')[0];
26. var tfoot = table.getElementsByTagName('tfoot')[0];
27. var tr = [].slice.call(tbody.getElementsByTagName('tr'), 0);
28. var total = 0;
29. tr.forEach(function (tr) {
30. total += +(tr.getElementsByTagName('td')[1].innerHTML.trim());
31. });
32. tfoot.getElementsByTagName('td')[0].innerHTML = total.toFixed(2) + '(' + tr.length + '件商品)';