**一、 实验目的**

1. 加深对求解一个3 \* 3方格两两相邻为质数求解算法的理解;
2. 通过本次试验掌握将算法转换为上机操作;
3. 加深对深度优先搜索思想的理解，理解回溯原理即实现过程，并利用其解决生活中的问题。

**二、实验内容**

任务：求解填字游戏问题

在3\*3个方格的方阵中填入数字1-10的某9个数字，每个方格填-一个整  
数，使所有相邻两个方格内的两个整数之和为素数。编写一个程序，求出所有  
满足这个要求的数字的填法。

**三、实验原理**

首先本题就是求一个3 \* 3方格中填十个数1-10，使得两两相邻的之和为质数，求解所有方案！

本题适合采用深度优先搜索DFS来求解：

1. 从（0，0）开始向右搜索，搜到（3，0）结束
2. 搜索时记录那些点被用过，下一个点一定是没有被用过的点，使用vis数组标记
3. 退出条件：搜索到 x == 3时，即此时九个点都已经填了一遍值，输出填入的九个值
4. 改点是否可以填入，是否合法：使用check函数来检查一下，由于是从左向右开始填起，所以相邻元素只需要考虑上和下两个方向，check函数加上判断边界的条件即可。
5. 若该点合法，则填入该点，否则继续循环找一个可选点。
6. 判断边界：即不能超出 3 \* 3 的范围，到了最右边要进行换行，否则横坐标直接++即可！具体实现：if(y == 2) dfs(x + 1, 0); else dfs(x, y + 1);
7. 最后取消标记，回溯上一个点，找下一种选择情况。

**四、程序代码**

说明： 深度优先搜索一遍即可。

函数说明：

1. Isprime()：用来判断是否是质数
2. Check()：用来判断当前位置放入一个值是否合法，即相邻是否为质数。
3. Dfs()：用来深度优先搜索一遍九个位置，并将符合条件的进行打印

代码如下：

1. #include <iostream>
2. using namespace std;
3. int a[3][3], count;
4. bool vis[10];
5. bool isprime(int n)
6. {
7. for(int i = 2; i \* i <= n; i++){
8. if(n % i == 0) return false;
9. }
10. return true;
11. }
12. bool check(int x, int y, int k)
13. {
14. *// 上*
15. if(x - 1 >= 0 && !isprime(a[x - 1][y] + k)) return false;
16. *// 左*
17. if(y - 1 >= 0 && !isprime(a[x][y - 1] + k)) return false;
18. return true;
19. }
20. void dfs(int x, int y)
21. {
22. if(x == 3){
23. for(int i = 0; i < 3; i++){
24. for(int j = 0; j < 3; j++){
25. cout << a[i][j] << " ";
26. }
27. cout << endl;
28. }
29. cout << endl;
30. count ++;
31. return;
32. }
33. for(int i = 1; i <= 10; i++){
34. if(!vis[i] && check(x, y, i)){
35. a[x][y] = i; vis[i] = true;
36. if(y == 2) dfs(x + 1, 0);
37. else dfs(x, y + 1);
38. a[x][y] = 0; vis[i] = false;
39. }
40. }
41. }
42. int main()
43. {
44. dfs(0, 0);
45. cout << "Total: " << count << endl;
46. return 0;

**五、实验结果**

说明：总共有128种方案