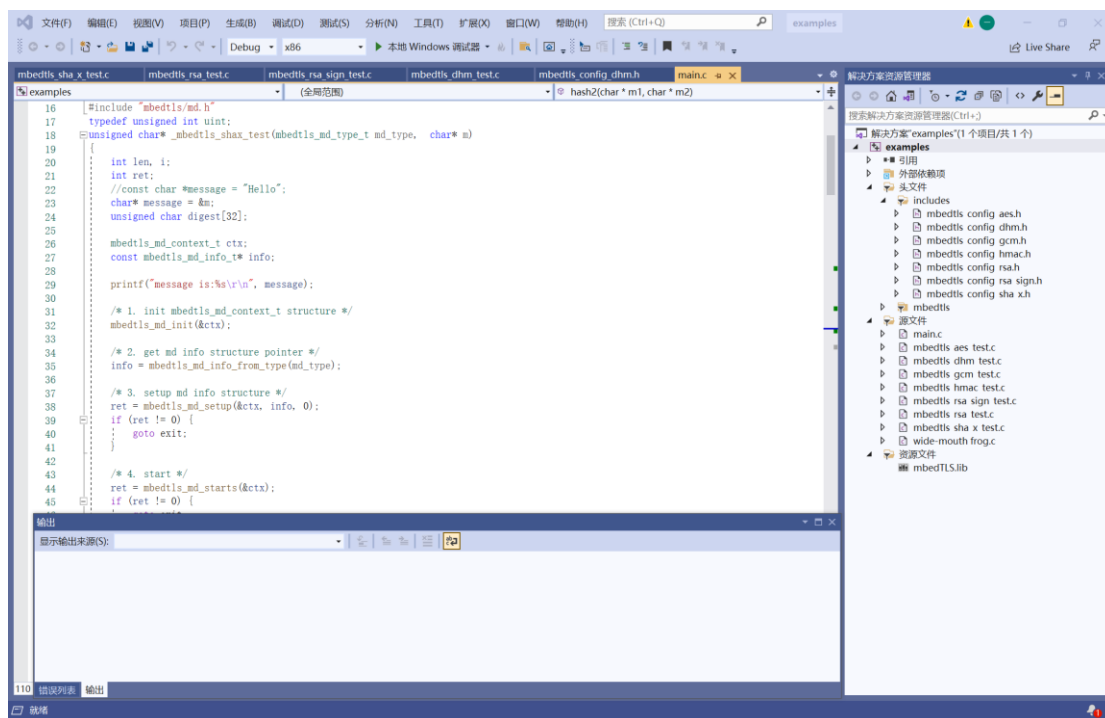


## Merkle tree 代码说明

Merkle tree 是一种特殊的满二叉树，按照 RFC6962 标准来实现 merkle tree，对于 hash 函数的选用是 sha256。



使用了 mbedtls 库，具体的代码使用的是上学期密码学引论中使用的代码。

Merkle 树的最底层，储存的是一个一个 data，它的父结点储存的是它的 hash 值。接着就是普通的二叉树，每一个节点储存的是 hash(leftchild||rightchild)。

```
// 计算一个整数的hash值
unsigned char* hash1(unsigned char* m) {
    unsigned char* result = _mbedtls_shax_test(MBEDTLS_MD_SHA256, m);
    return result;
}

// 计算两个整数的hash值
unsigned char* hash2(char* m1, char* m2) {
    char* tmp = NULL;
    sprintf(tmp, "%s%s", m1, m2);

    unsigned char* result= _mbedtls_shax_test(MBEDTLS_MD_SHA256, tmp);
    return result;
}
```

Hash1 是针对一个字符串进行哈希。

Hash2 是先将两个字符串级联起来在进行哈希。

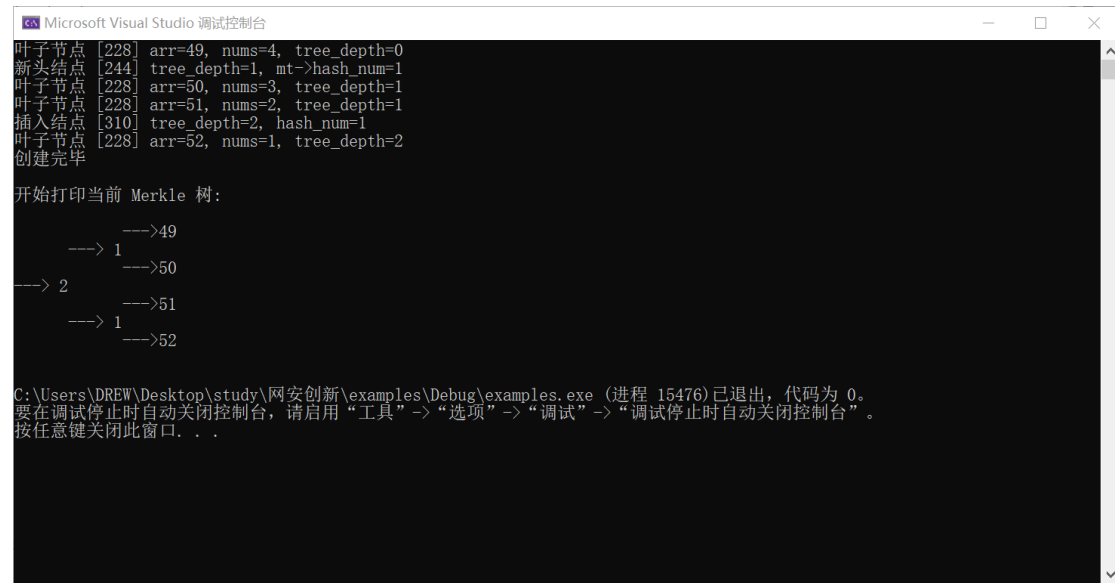
Print\_MTree() 是将树打印出来的函数，使用了递归的结构。

hash\_Merkle() 是用来计算每个节点 hash 值的函数。

Creat\_MTree(MTNode\* mt, char\* arr, int nums, int tree\_depth) 是用来建立树的函数，mt 是根

节点，arr 是用来建立树的数据，nums 是数据的数量，tree\_deeth 是树高。同样采取了递归结构。

运行结果：



```
Microsoft Visual Studio 调试控制台
叶子节点 [228] arr=49, nums=4, tree_depth=0
新头结点 [244] tree_depth=1, mt->hash_num=1
叶子节点 [228] arr=50, nums=3, tree_depth=1
叶子节点 [228] arr=51, nums=2, tree_depth=1
插入结点 [310] tree_depth=2, hash_num=1
叶子节点 [228] arr=52, nums=1, tree_depth=2
创建完毕

开始打印当前 Merkle 树:

    --->49
    ---> 1
    --->50
---> 2
    --->51
    ---> 1
    --->52

C:\Users\DREW\Desktop\study\网安创新\examples\Debug\examples.exe (进程 15476) 已退出, 代码为 0。
要在调试停止时自动关闭控制台, 请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口. . .
```