Logiscope

实验报告

课程名称：	软件质量保证与测试技术
姓　　名：	武敬信
学　　号：	202000800525
专业班级：	21软件工程1班
授课教师：	康钦马

2024 年 4 月 3 日

1 实验目的

安装并配置Logiscope，熟悉Logiscop三大功能Audit、RuleChecker、TestChecker的使用，以及解决一些其它相关问题。

2 实验环境

虚拟机：Windows XP

软件：Logiscope 6.1

CPU：AMD Ryzen 7 5800H

内存：8GB

3 实验步骤

3.1 Logiscope 安装与配置

安装 Logiscope 6.1。

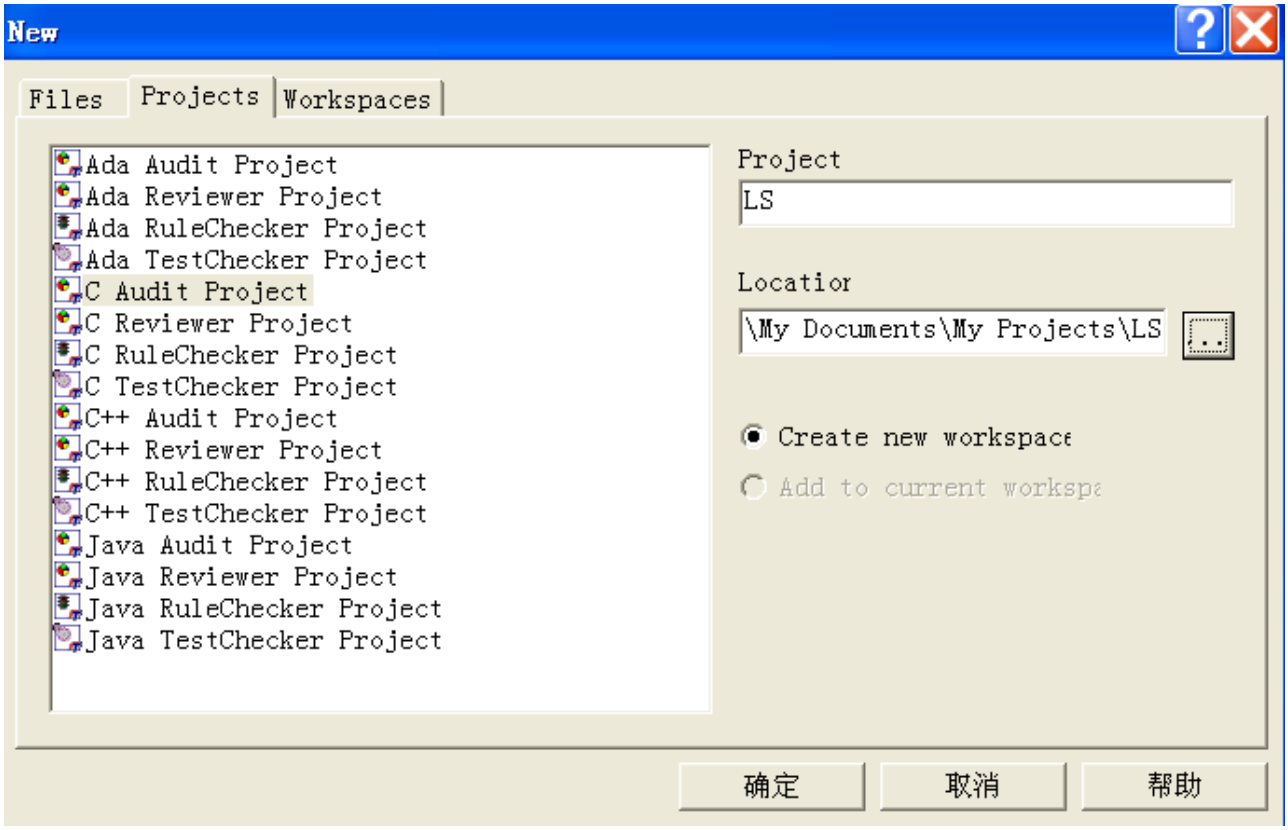
将系统时间设置为 2000 年。激活 Logiscope。

3.2 Logiscope 质量模型与 Audit 结果分析

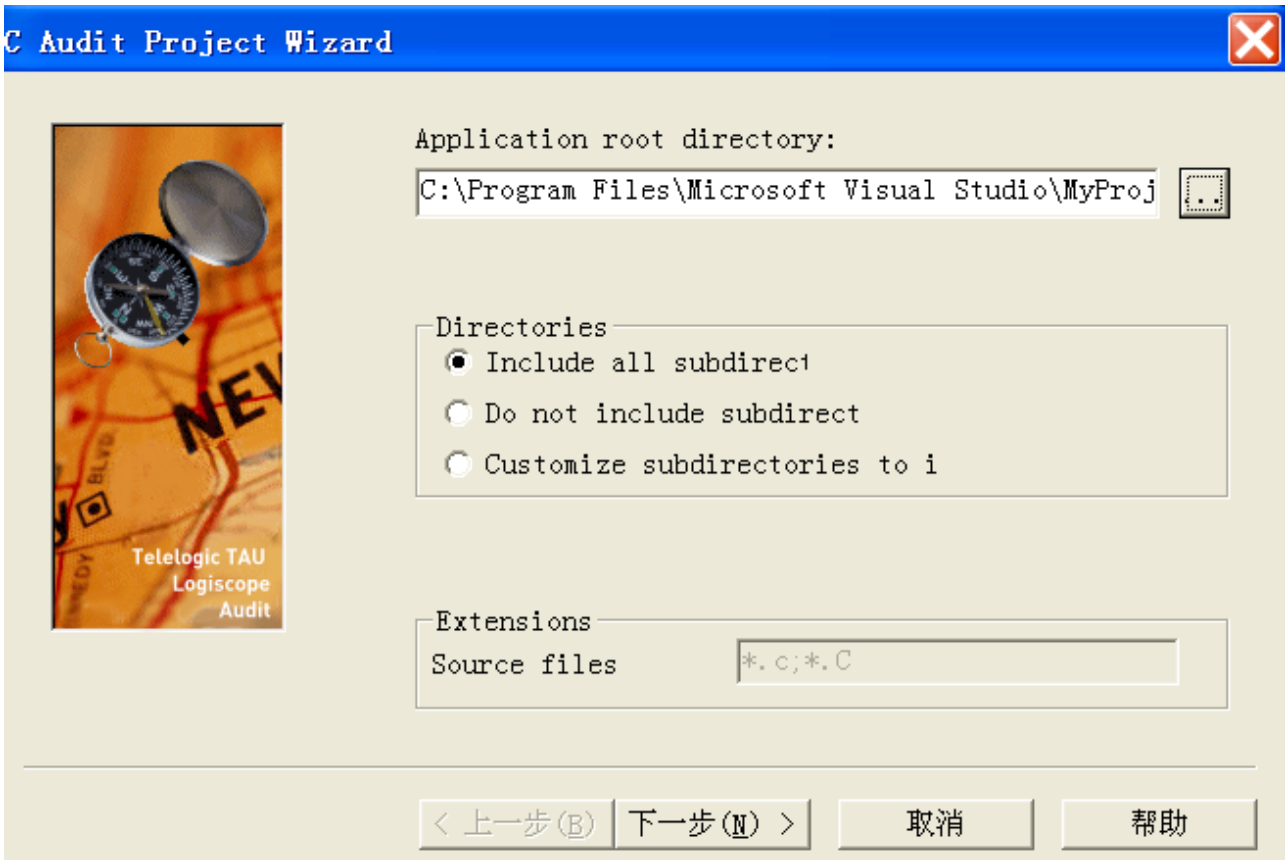
首先，需要有一段用来测试的 C 语言代码：

```
#include <stdio.h>

void main()
{
    int a = 0;
    int b = 0;
    printf("请输入两个整形数据a和b: \n");
    scanf("%d%d", &a, &b);

    if (a != b)
        if (a > b)
            printf("a>b\n");
        else
            printf("a<b\n");
    else
        printf("a=b\n");
}
```

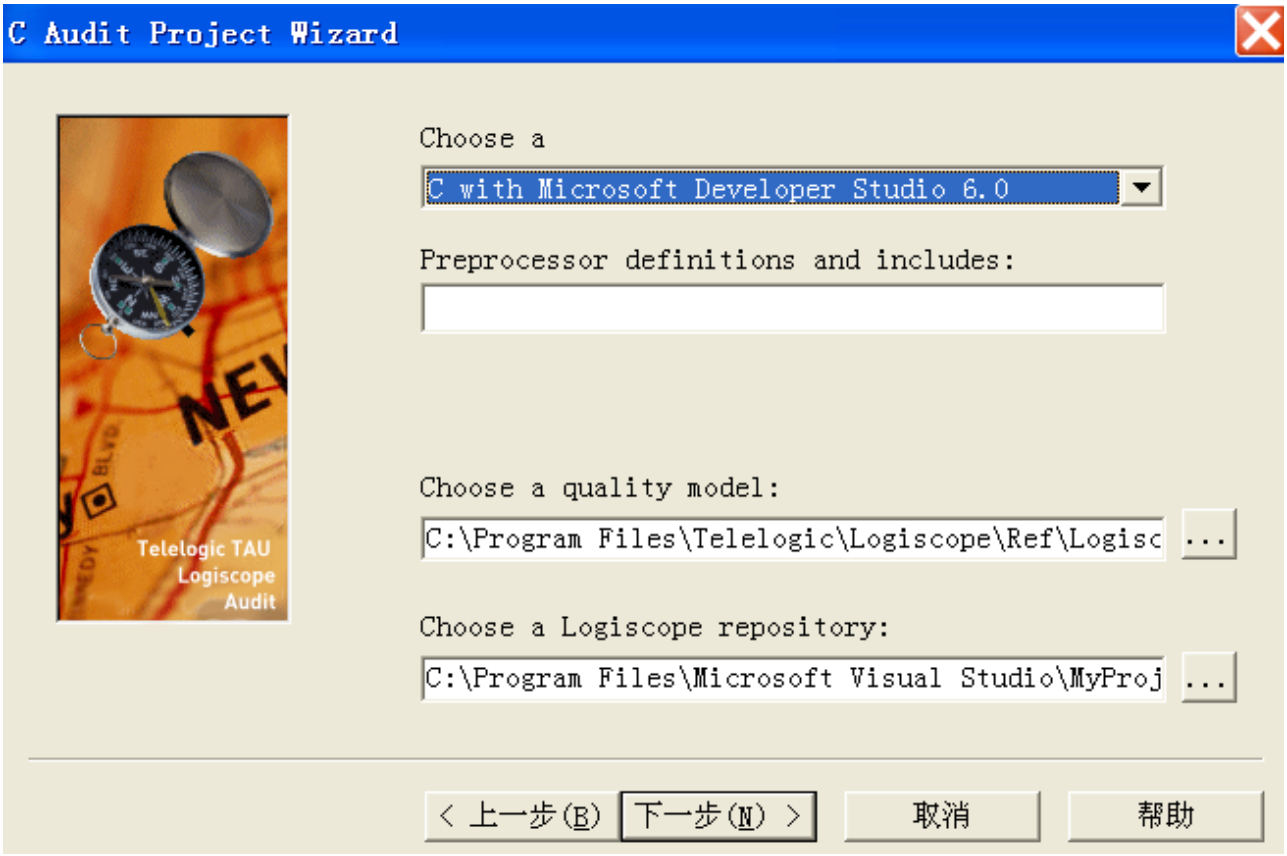
这段代码需要在 VC 中编译好。



点击 确定，然后在对话框中选择刚创建的 VC 工程所在的路径：



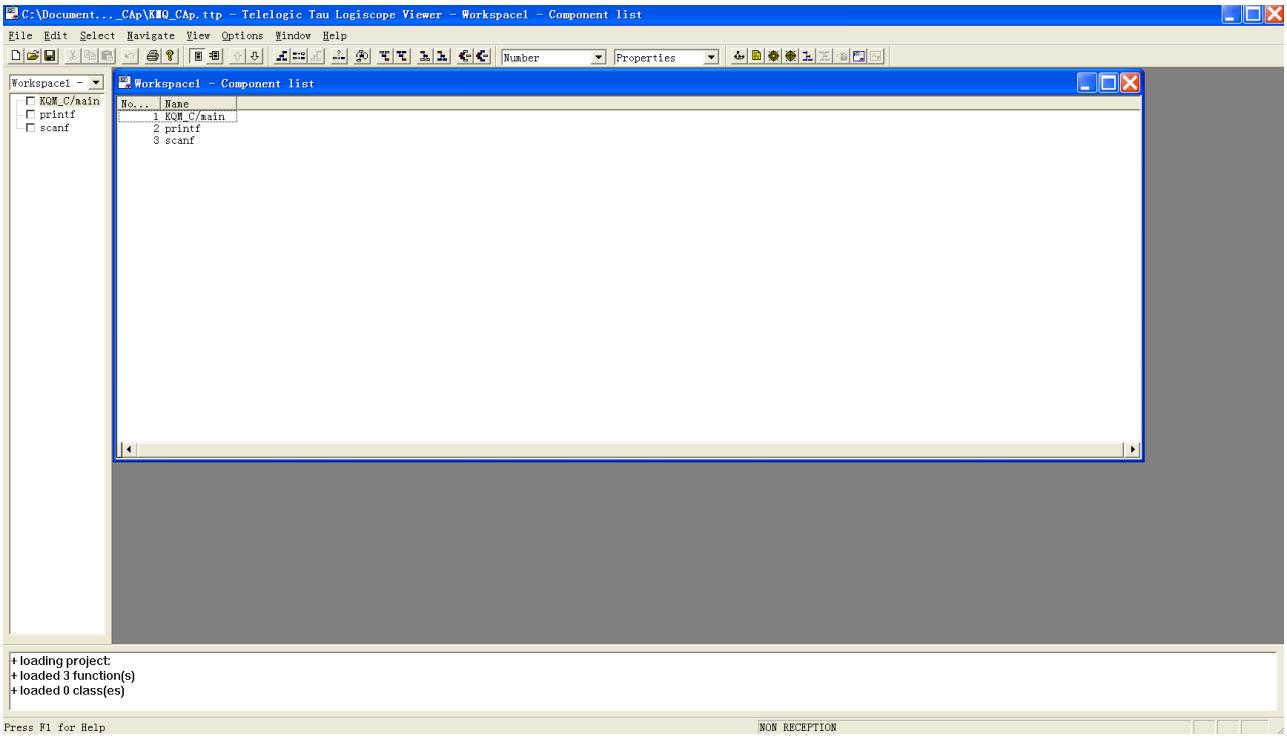
接着点击 下一步，选择默认的 VC 6.0 编译器，质量模型保持默认。



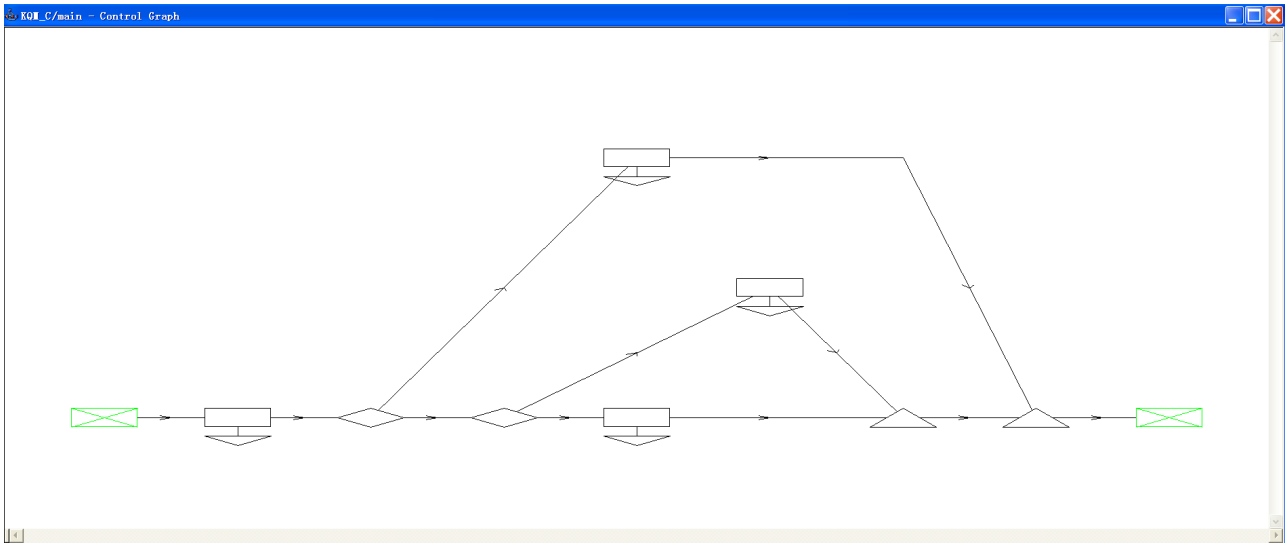
单击 完成 按钮，进入 Logiscope 主界面。

Logiscope 的质量模型认为衡量代码质量的好坏有三个级别。而我们将要利用它对上面的代码进行分析。首先进行度量元级结果分析。

选择 Project/Build 对其编译，然后选择 Project/Start Viewer ，启动 Logiscope Viewer 。

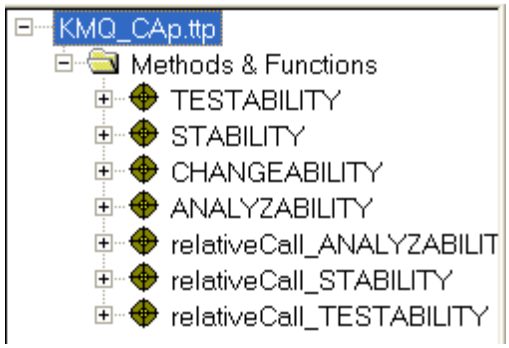


调用 Logiscope Viewer 模块，并且进行分析。



同样，点击`显示函数源代码`、`显示函数的度量元检测结果`都会展示结果。

返回主程序，点击`Browse/Quality/Criteria Level`，会自动生成被测程序在质量标准级的检测结果。



点击`Browse/Quality/Factor Level`，软件会自动生成被测程序在质量因素级的检测结果。

点击`Browse/Quality/Quality Report`，可以看到软件自动生成的被测程序的结果分析报告。

Metric Name	Max	Min	Out
Program length	+oo	-oo	0%
Vocabulary	+oo	-oo	0%
Estimated Length	+oo	-oo	0%
Volume	+oo	-oo	0%
Difficulty	+oo	-oo	0%
Level	+oo	-oo	0%
Mental Effort	+oo	-oo	0%
Intelligent Content	+oo	-oo	0%
Vocabulary frequency	4.00	1.00	0%
Number of nestings	+oo	-oo	0%
Number of macro-instructions	+oo	-oo	0%
Average size of statements	9.00	1.00	0%
Comments frequency	+oo	0.20	100.00%
Number of statements	50.00	1.00	0%
Number of direct calls	7.00	0.00	0%
Cyclomatic number (VG)	10.00	1.00	0%
Number of GOTO statements	0.00	0.00	0%
Number of RETURN statements	1.00	0.00	0%
Number of local variables	5.00	0.00	0%
Number of levels	4.00	0.00	0%
Number of callers	5.00	0.00	0%
Number of function parameters	5.00	0.00	0%
Number of paths	80.00	1.00	0%
Number of relative call graph levels	12.00	1.00	0%
Relative call graph Hierarchical complexity	5.00	1.00	0%

此外还有其他的功能，这里就不一一展开解释。

3.3 编码质量模块 RuleChecker 和 RuleChecker 结果分析

同样，首先需要创建一个 RuleChecker 工程。这里也需要一个程序。

被测程序的代码如下：

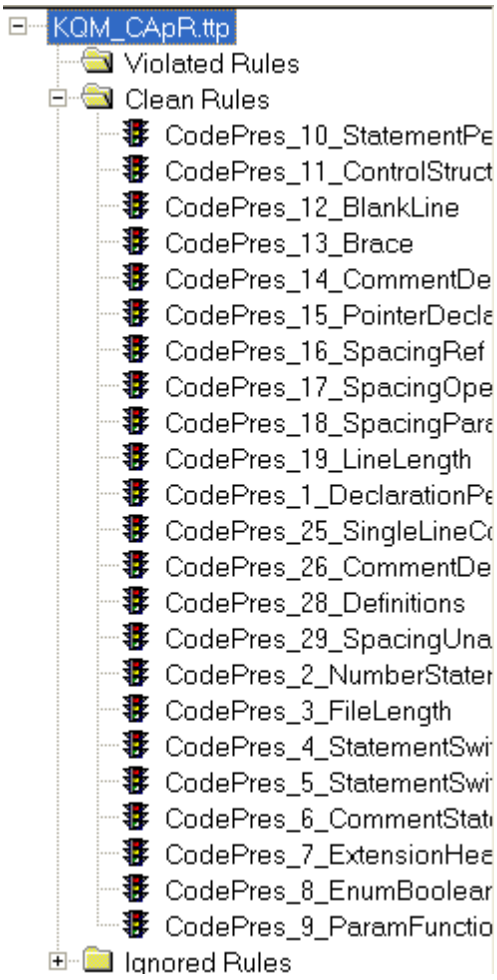
```
#include <stdio.h>

void main()
{
```

```
    printf("输入一个字符:\n");
    scanf("%d", &a);
    if (a < 32)
    {
        printf("控制字符:\n");
    }
    else if (a >= '0' && a <= '9')
        printf("数字\n");
    else if (a >= 'a' && a <= 'z')
        printf("小写字母\n");
    else if (a >= 'A' && a <= 'Z')
        printf("大写字母\n");
    else
        printf("其他字符\n");

}
```

和之前一样，也是首先要在 VC 中编译好之后，启动 Logiscope。

选择 `File/New`，创建 `C RuleChecker Project`。路径选择也是和之前一样。其他全选择默认即可。进入界面之后，还是首先 `Project/Build` 。

选择 `Browse/Rule/Rule Violations`，程序会自动将代码中不合编码规范要求的地方列举出来。



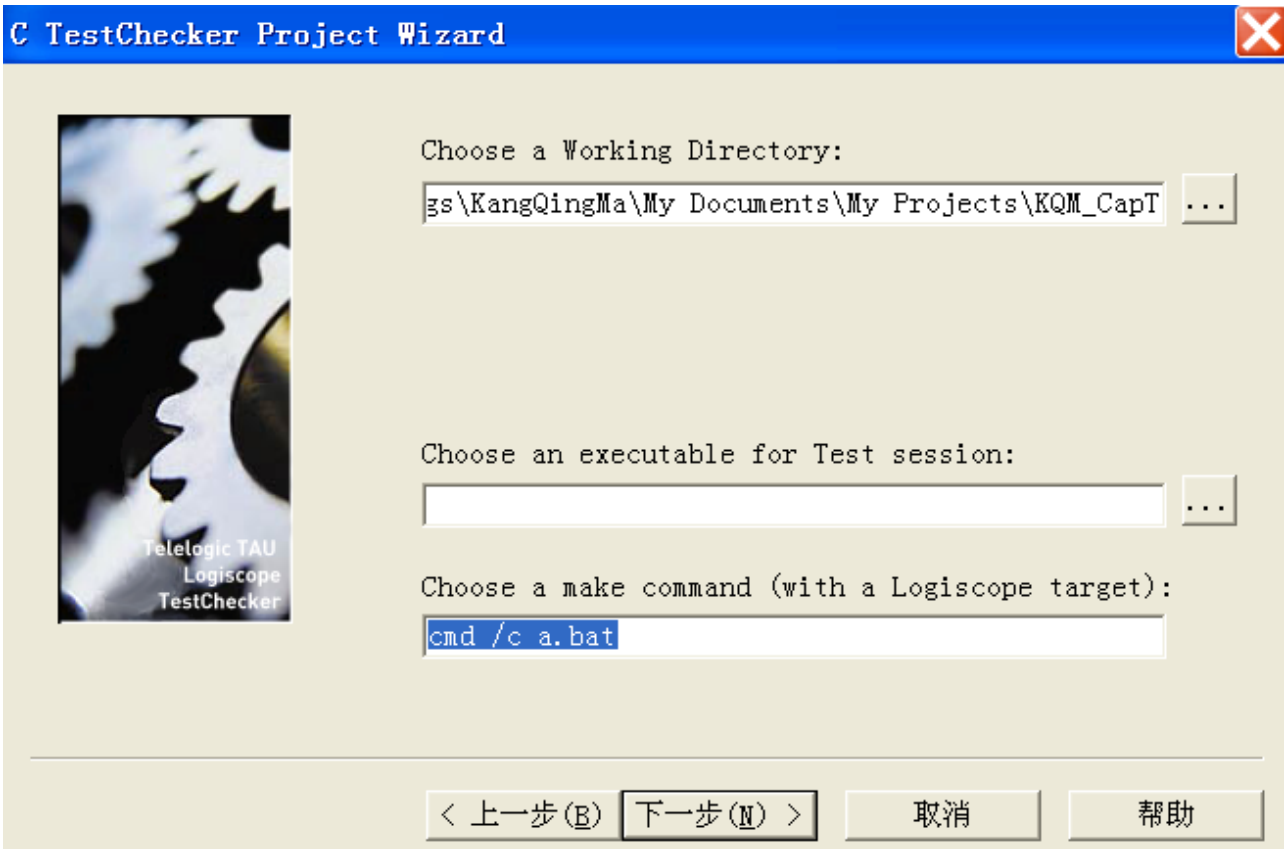
可以看到，上面给出的代码没有产生问题。

此外，RuleChecker 还可以自动生成统计报告：点击 `Browse/Rule/Rule Violations Report` ，程序就会自

3.4 动态测试模块 TestChecker

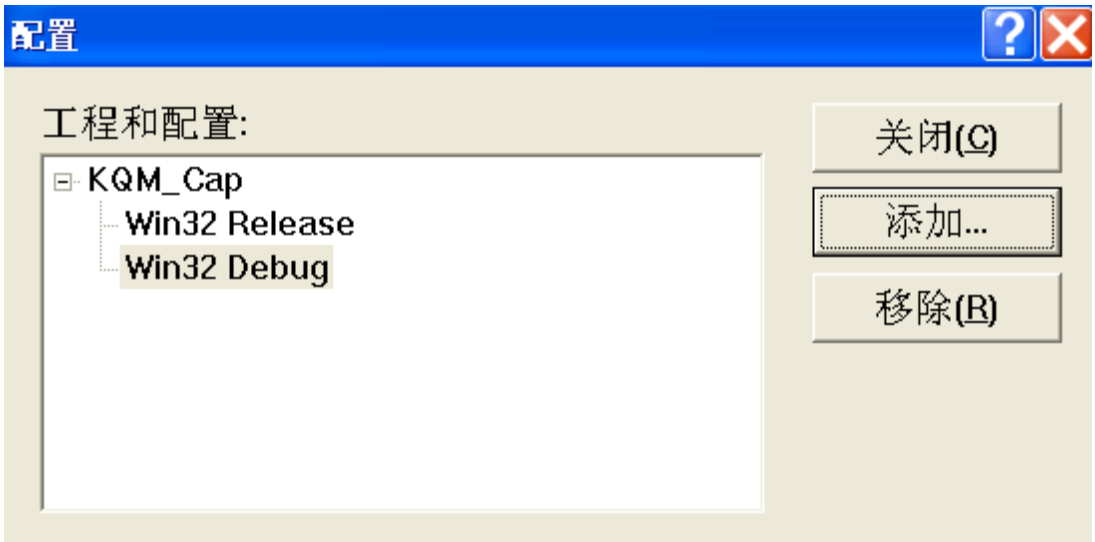
与之前一样，创建一个工程。

运行 Logiscope，选择 `File/New`，创建 `C TestChecker Project`。路径选择也是和之前一样。其他全选择默认即可，但有一点需要注意的是，在这里，需要输入 `cmd /c a.bat`。



进入界面之后，还是首先 `Project/Build`，这时发现无法编译。需要进一步修改。

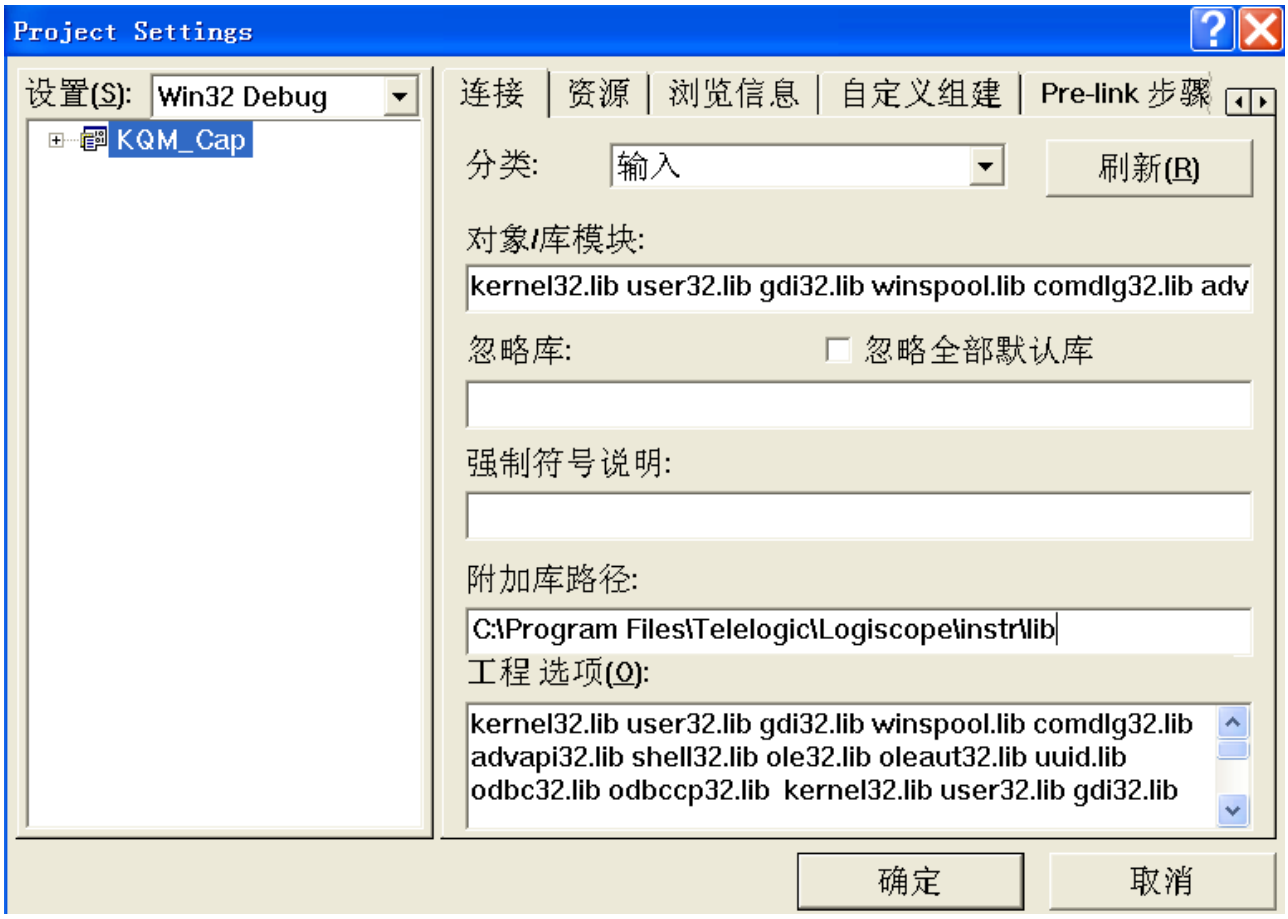
打开 VC 工程文件，选择 `Build/Configurations`，添加一个名为 `Logiscope` 的文件夹。



配置好之后，点击 `关闭`，完成配置。接着，点击 `工程/设置`，在对话框中选择 `C/C++`，在 `分类` 中选择 `预处理器`，在 `附加包含路径` 中输入 `C:\Program Files\Telelogic\Logiscope\instr\include`。

选择 连接 标签，然后在对象/库模块中输入 `vlgtc.lib`。

然后在 分类 中选择 输入 选项，在 附加库路径 中输入 `C:\Program Files\Telelogic\Logiscope\instr\lib` 。

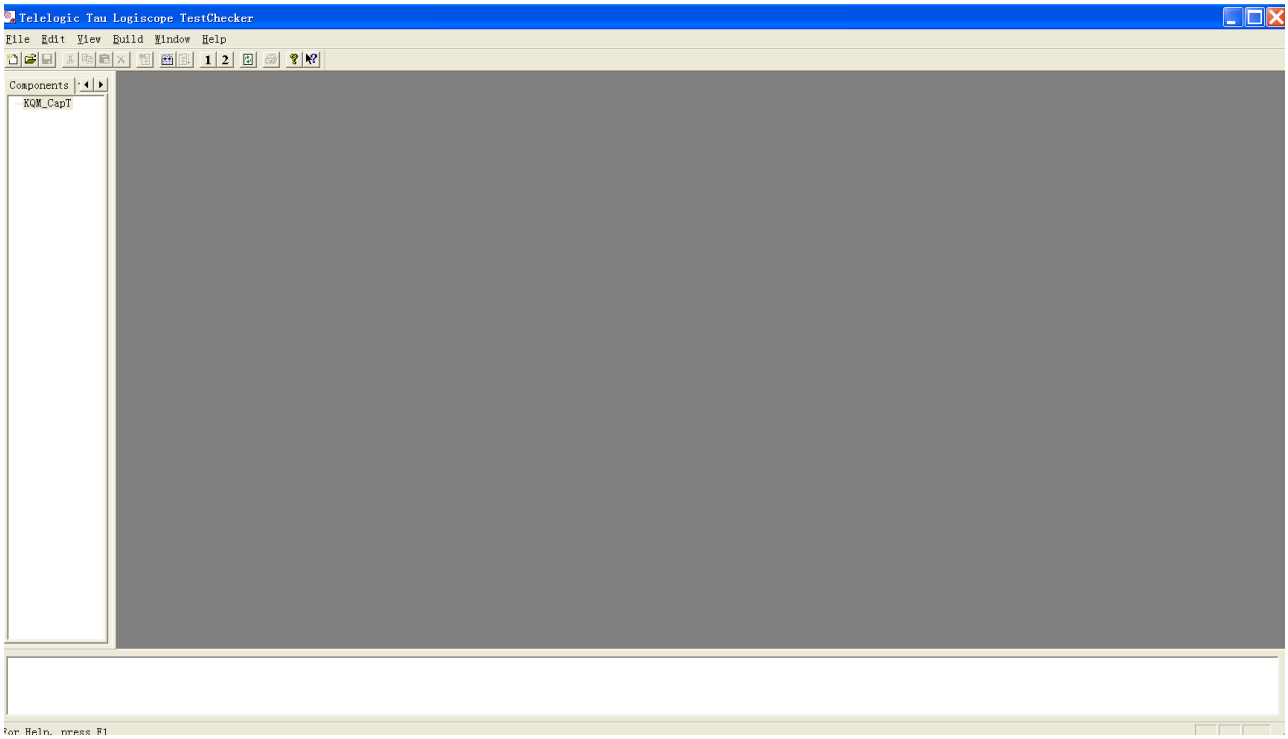


点击 确定 完成配置，选择 工程/导出制作文件 。从中选取需要的内容，然后点击 确定，然后在 VC 所在的工程文件的目录下，建立一个记事本，输入如下内容：

```
call"C:\program files\microsoft visual studio\vc98\bin\vcvars32.bat"

nmake /A /F KQM_Cap.mak CFG="KQM_Cap - Win32 Logiscope"
```

然后将这个文件重命名为 `a.bat` 。接着，回到 Logiscope，选择 `Project/Build` ，此时编译即可成功。然后选择 `Project/settings` ，选择可执行文件。

接着，选择 `Project/Start TestChecker` 。



接下来，运行时就可以在 TestChecker主界面看到分支覆盖率等信息了。通过添加测试用例，可以使得覆盖率达到 100%。