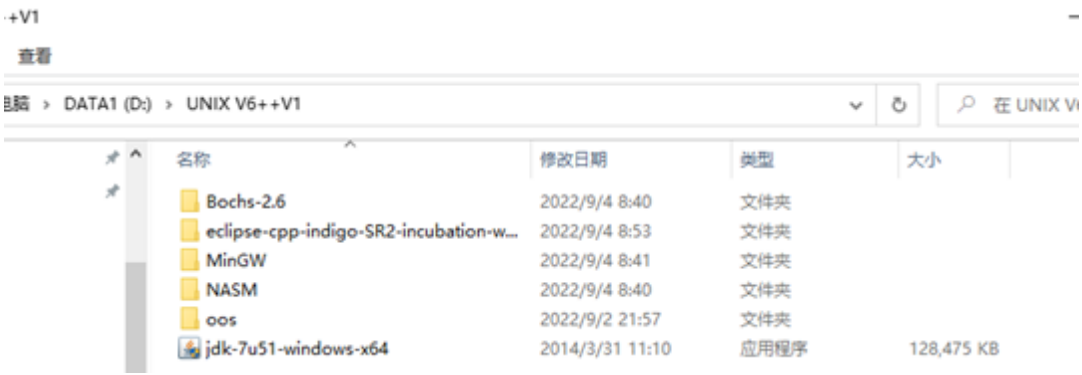
实验一、Unix V6++系统简介，安装方法 和集成开发环境的配置（本实验 5 分）

2022 年 9 月 4 日 10:08

Unix V6++的工程包和软件依赖：

- oos 文件夹是 Unix V6++工程包，含系统源代码、配置文件 和 开发工具。
- Bochs-2.6，这个工具帮我们虚拟出一台 X86 架构（基于 32 位 Intel 芯片 i386）的 PC 机。Unix V6++系统运行其上。
- NSAM，汇编器。用来汇编自举程序中的 IBM 汇编代码。
- MinGW，里面有编译、链接、调试 Unix V6++内核 和 所有应用程序 的 GNU 工具。gcc，g++，am，make，gdb……



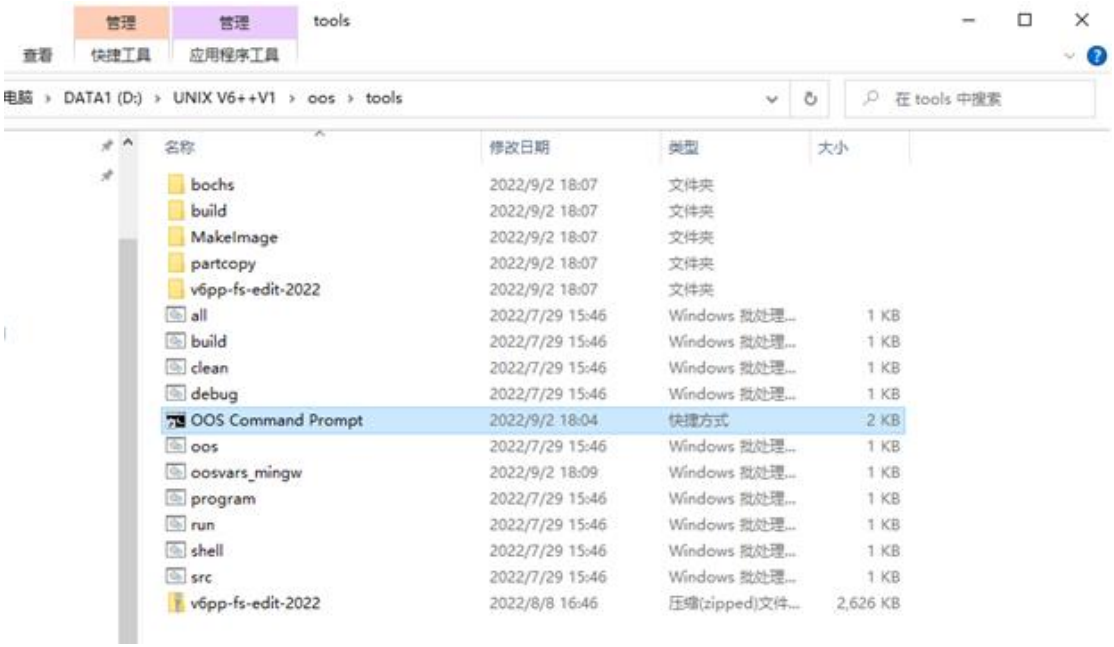
未来，要求大家装 2 套 Unix V6++。本次实验，只装稳定版 ！！！

- 一套是稳定版（就是今天给大家的包，里面是性能较稳定的新系统），上课读源代码。新系统中有龚天遥同学开发的文件系统工具。制作系统盘的时候，all 命令用到了这个工具。未来，需要将外部文件送进 Unix V6++系统的时候，也要用到这个工具。

- 另一套开发版，用来在稳定版的基础上做实验、进行个人探索。

Part 1：按 **P01** 的指示，安装 **Unix V6++**系统。

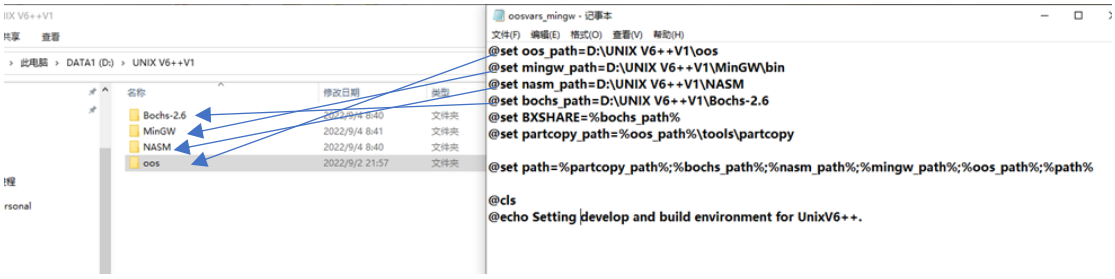
目标：双击如下图标，可以在这个 cmd 界面中重新编译、build 和 运行系统。



实验步骤：

0、解压 UNIX V6++包。

1、编辑 oosvars_mingw.bat 文件，设置 oos_path, mingw_path, nasm_path 和 bochs_path.....关键路径，令其指向你解压 UNIX V6++包的位置。



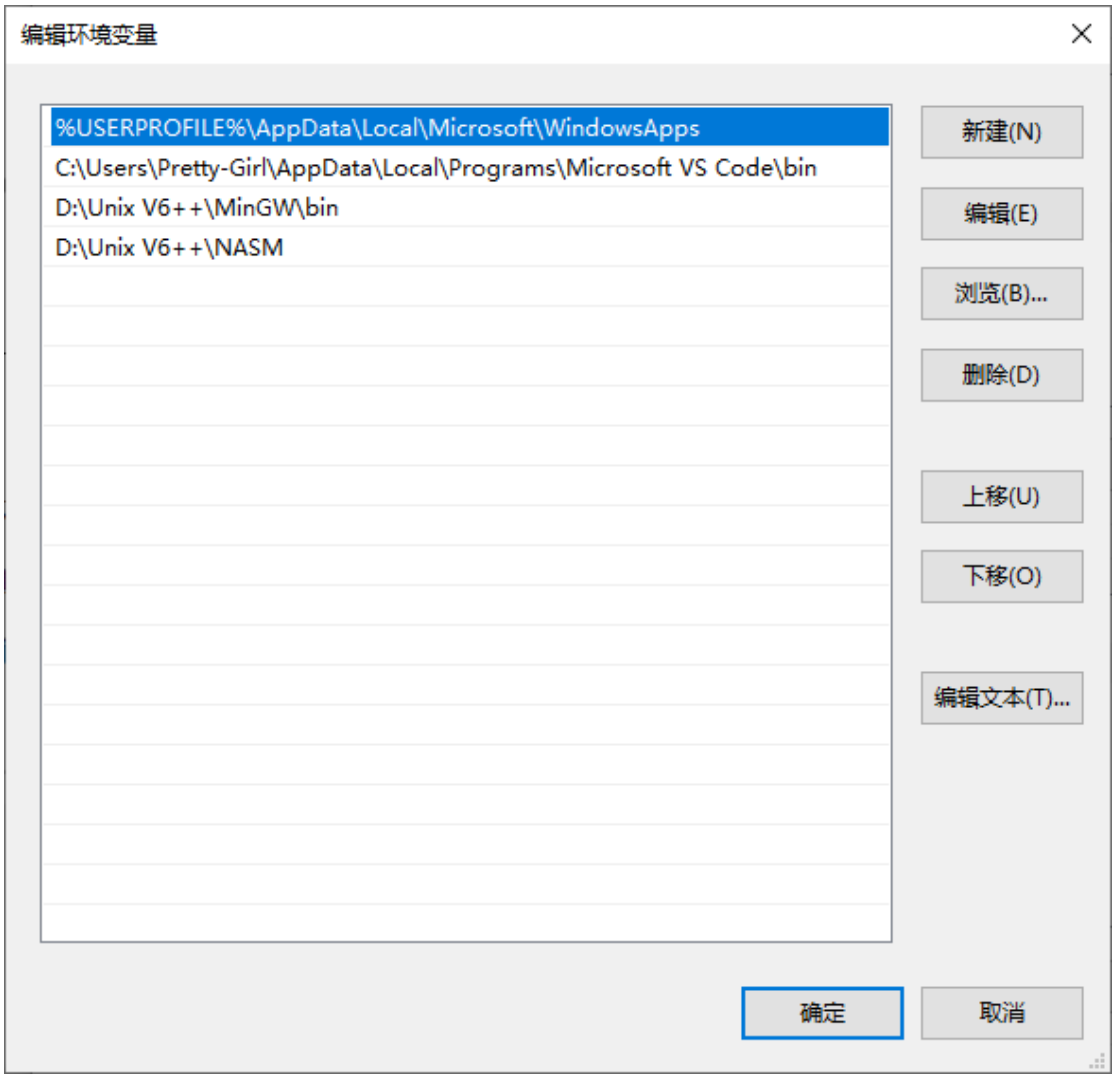
2、执行批处理命令 clean，清除已有全部目标文件（二进制文件，.o），擦除已有镜像文件 c.img。

注：c.img 是 UNIX V6++系统的硬盘，里面有 UNIX V6++操作系统内核、文件系统、引导扇区、盘交换区。这些是一个标准通用操作系统所需要的全部。文件系统中有 UNIX V6++可以执行的全部应用程序（包括命令行界面 shell 程序）。

3、执行批处理命令 all，重新编译内核和应用程序、制作新的系统镜像文件 c.img。

3.1 预备：在 windows 操作系统中设置**系统环境变量 path**。加入 MinGW 和 NASM 的路径。重新编译、链接内核的时候（也就是执行 all 命令的时候），我们要用里面的工具。比如，编译、链接内核要用 g++，make；编译、链接 shell 和其它应用程序，要用 gcc；发调试命令，我们要用 gdb。这些工具在 MinGW/bin。

UNIX V6++的自举程序独立于内核，是一段 IBM 汇编程序。这段代码的汇编程序是 nasm.exe。我们需要让集成开发环境知道这些工具的位置。（应该不需要，但保险点，用户环境变量 path 也一样设置一下这 2 个路径）。

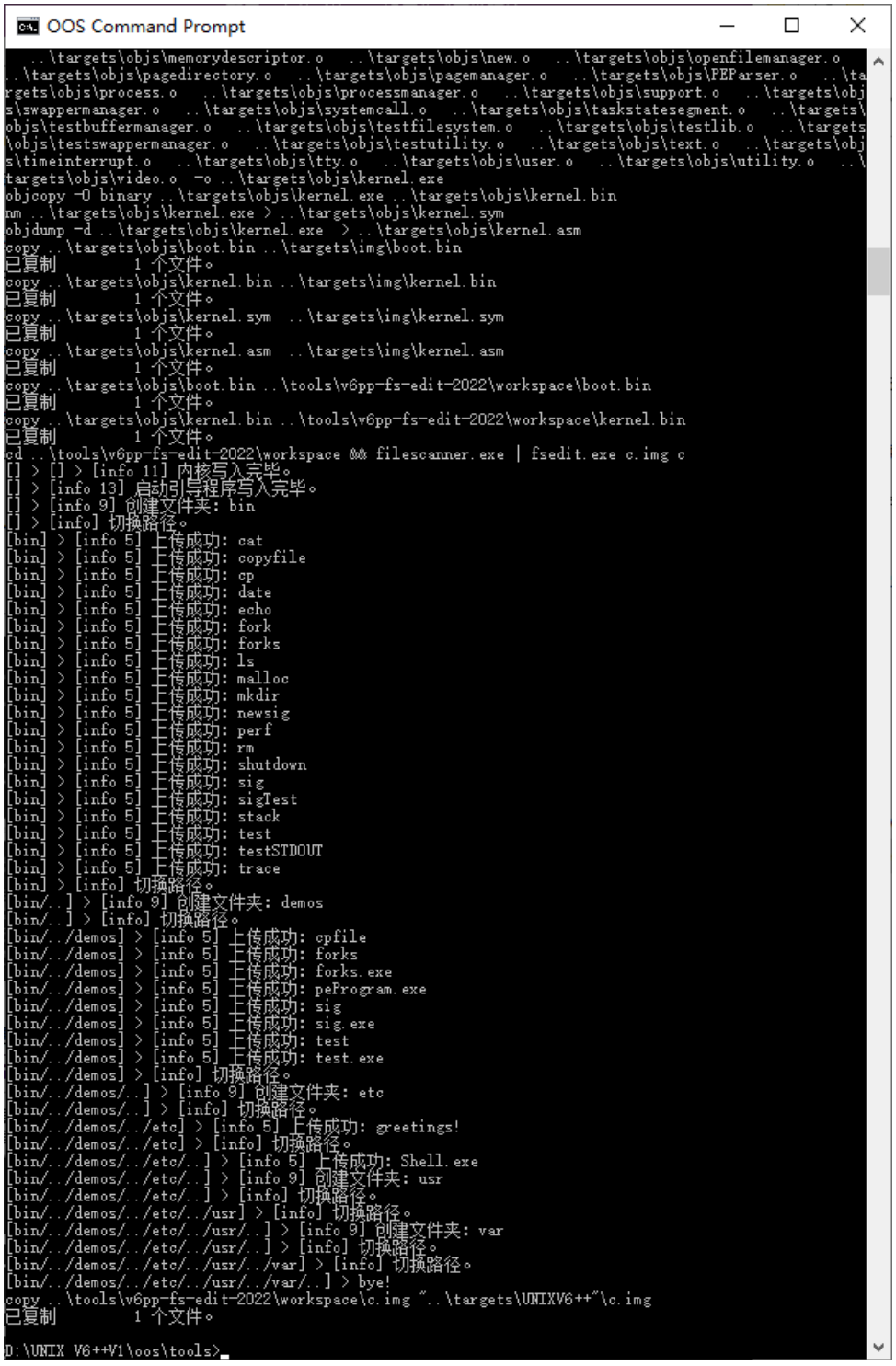


注意，如果你的机器先前安装过 MinGW 或 CgWin。注意把我们的路径排在最上方。

3.2　　预备：我们的新系统，all 命令不需要关闭实时防护。

老系统需要关闭 win10 的实时防护。否则，批处理命令 all 通不过。

3.3　　执行批处理命令 all，重新编译内核和应用程序、制作新的系统镜像文件 c.img。



这是完成以后的样子。

all 会进源代码中的所有子目录，执行 make 命令。编译该目录下的所有 .cpp 或 .c，生成目标文件 .o。具体执行过程如下，

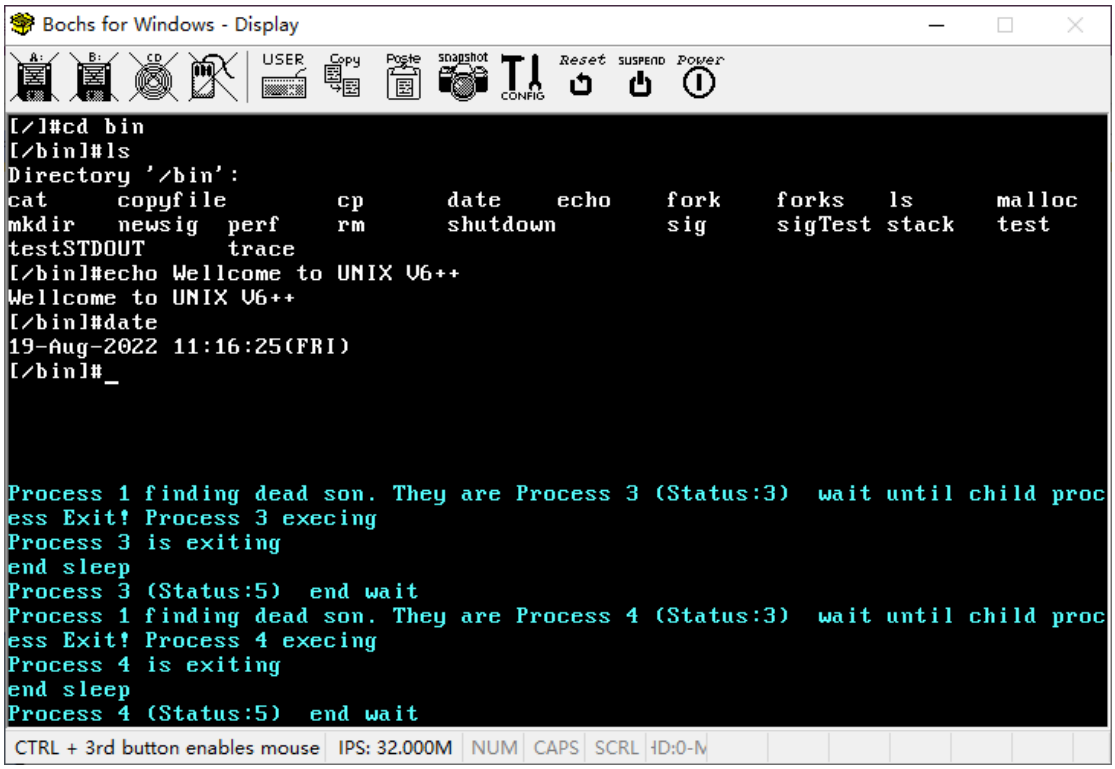
- 链接所有必要的 .o，生成内核 kernel.exe。
- 链接所有必要的 .o，生成每个应用程序。
- 把 kernel.exe 复制到新建的镜像文件 c.img 合适的位置。
- 格式化 c.img 上的文件系统，之后将 shell 程序 和 其它应用程序（UNIX V6++的工具）放入文件系统。
- 完成。新的 c.img 包括你先前对系统所作的一切改动。

注意：对系统中的程序做过任何改动后，必需 clean，all。让你的改动在新的 c.img 中生效。

4、run，运行 UNIX V6++系统

右侧，是 bochs 虚拟机的显示屏。

5、试试已有工具（系统中已有的应用程序）。它们在 Unix V6++文件系统的 bin 目录下。



echo 命令，回显字符行。

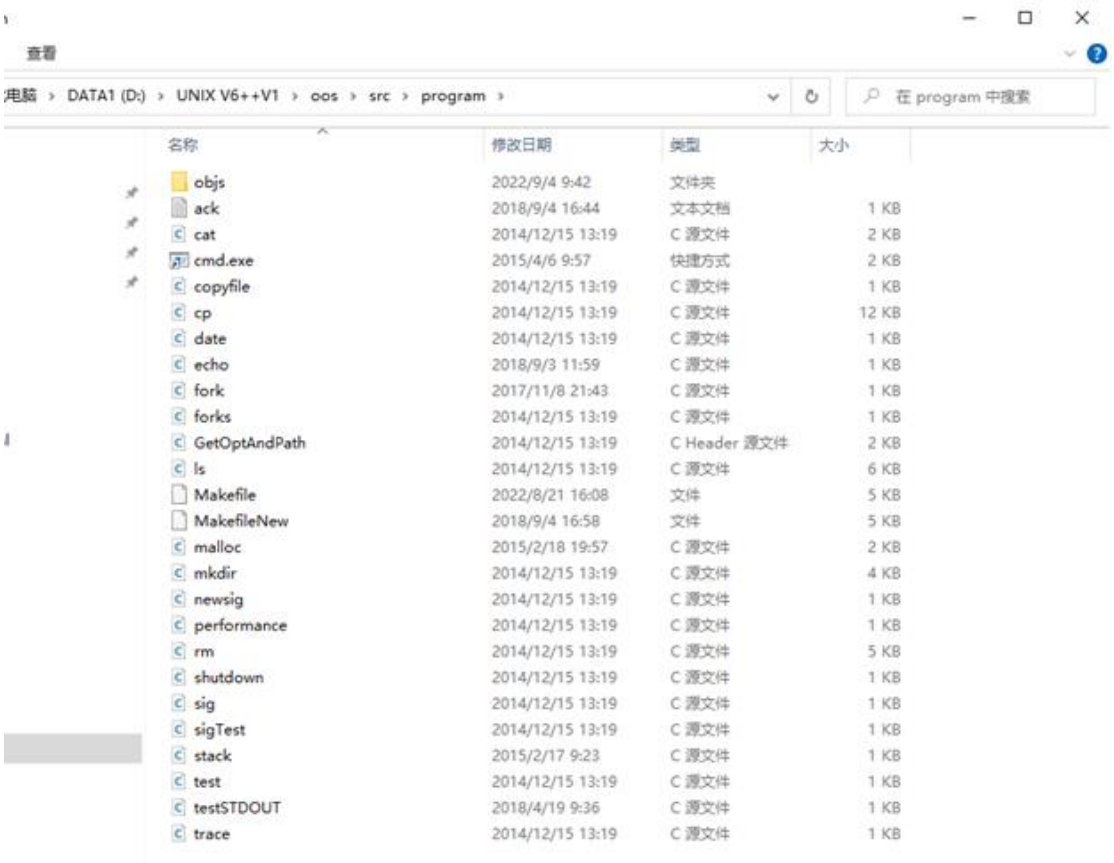
date 命令，显示当前时刻。

ls 命令，列目录。

cp 命令，复制已有文件。

shutdown 命令，安全关机。本次使用系统，如果你有创建、删除 文件/文件夹。执行这个命令后，更新会反应在 c.img。不 shutdown 的话，更新不会生效，关机后启动系统，c.img 还是原来的样子。

这些应用程序，源代码在这里。

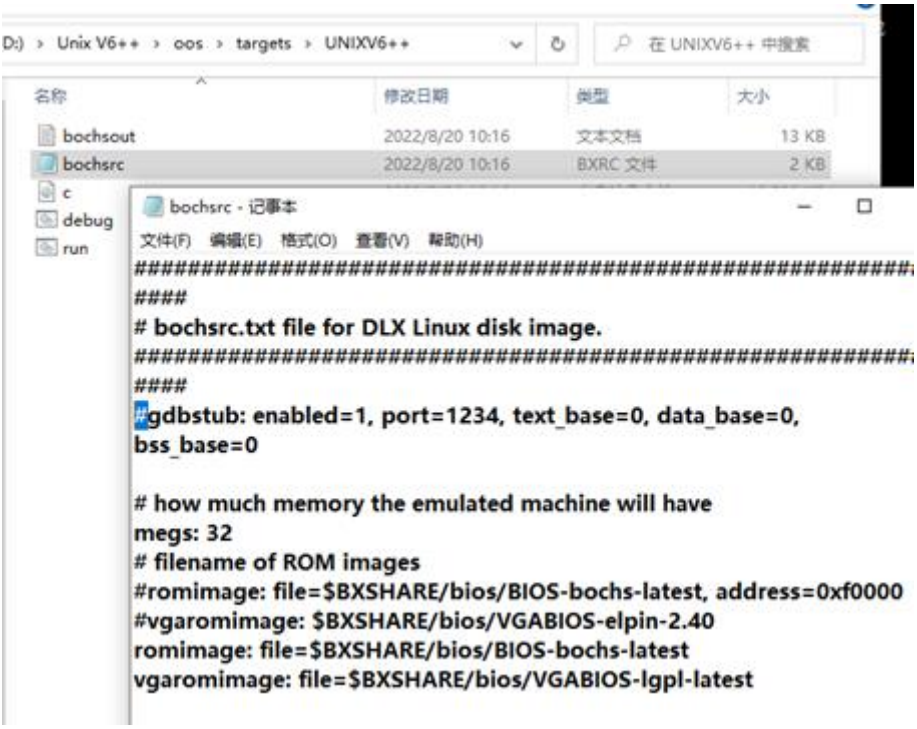


6、本部分实验报告要求：

6.1　完成 Unix V6++基本开发、运行环境的安装与配置，执行几条简单的 Unix 命令，关键步骤截图说明。（1 分）

Part 2、配置 UNIX V6++的集成开发环境（eclipse 版）

1、 使能 bochs 机的 gdb stub。方法，去掉 bochsrc.BXRC 文件中，下图我高亮的那个#。随后，bochs 虚拟机可以通过 1234 号端口接收外部，比如集成开发环境 eclipse 或 VSCode 向它发送的 gdb 命令。我们用这种方法调试 UNIX V6++内核。



这个 bochsrc.txt 文件是 bochs 虚拟机的配置文件。看，megs:32 表示内存 32M 字节。它还指定了虚拟机的磁盘。

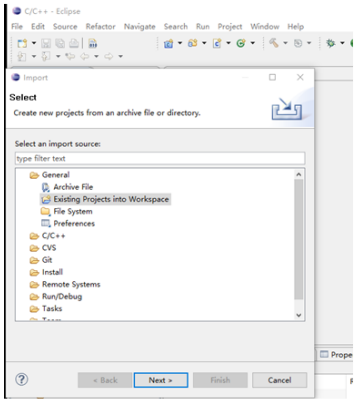
2、 安装 JDK，运行 eclipse

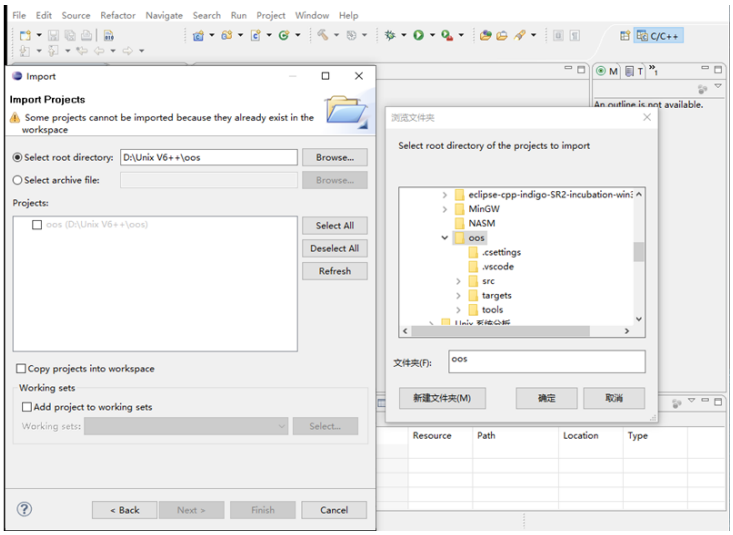
1、装包里的Java，记下你的安装路径

2、双击、启动包里的 eclipse

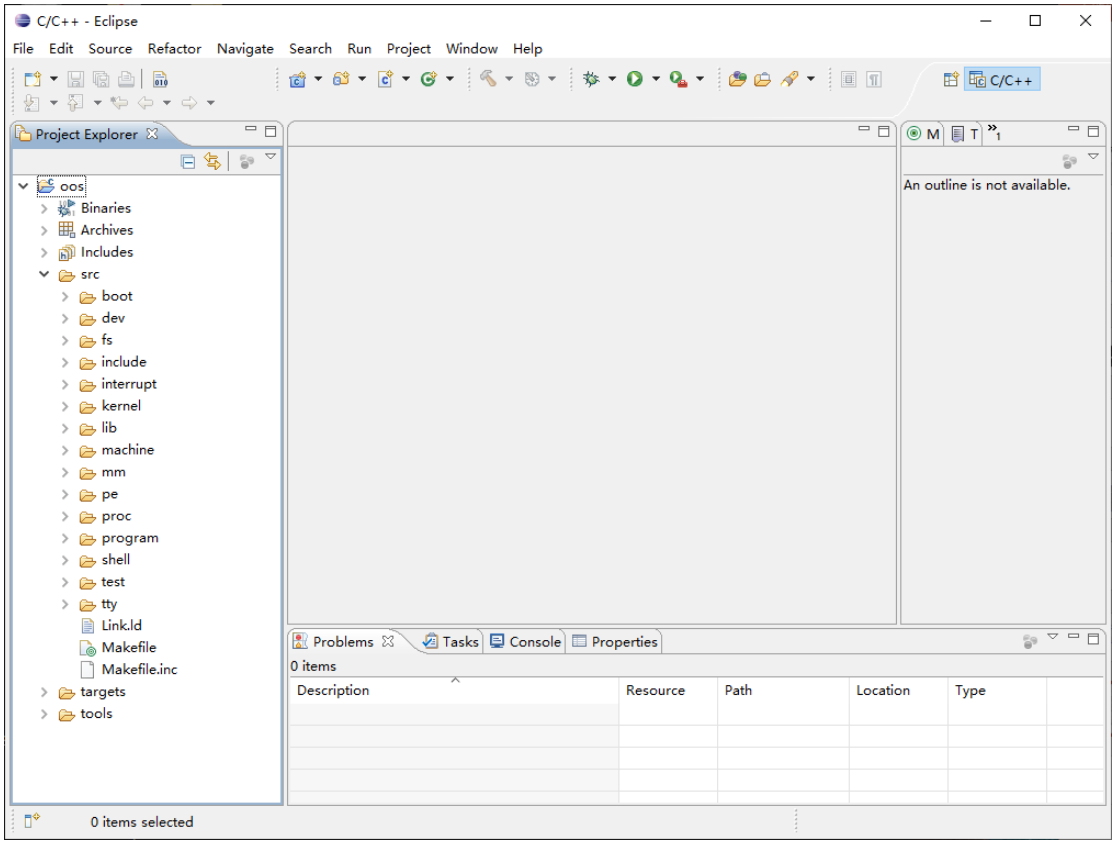
3、配置编译环境

3.1　　File -> import 导入 UNIX V6++工程。





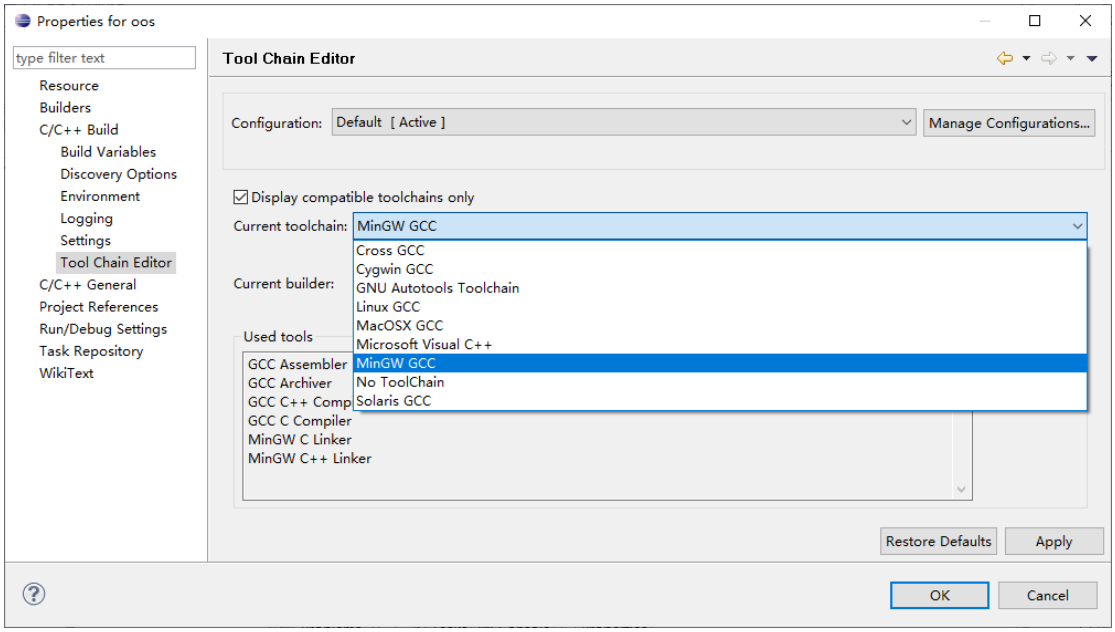
3.2　关闭 Welcome 页面，你会看到 UNIX V6++ 系统的软件构成。src 是系统源码。



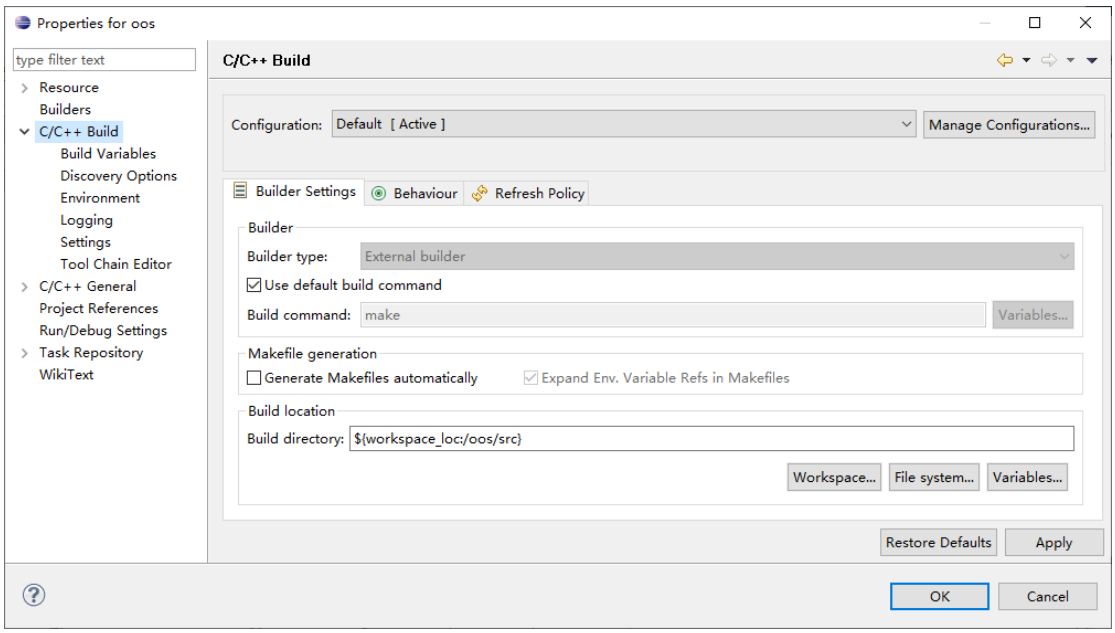
3.3 设置 oos 工程属性

Project Explorer 选中 oos 工程。

File-->Properties-->C/C++ Build --> Tool Chain Editor，将 Current toolchain 改为 MinGW GCC。之后，Apply、OK。



回到 C/C++ Build，主选项卡应该是这样：

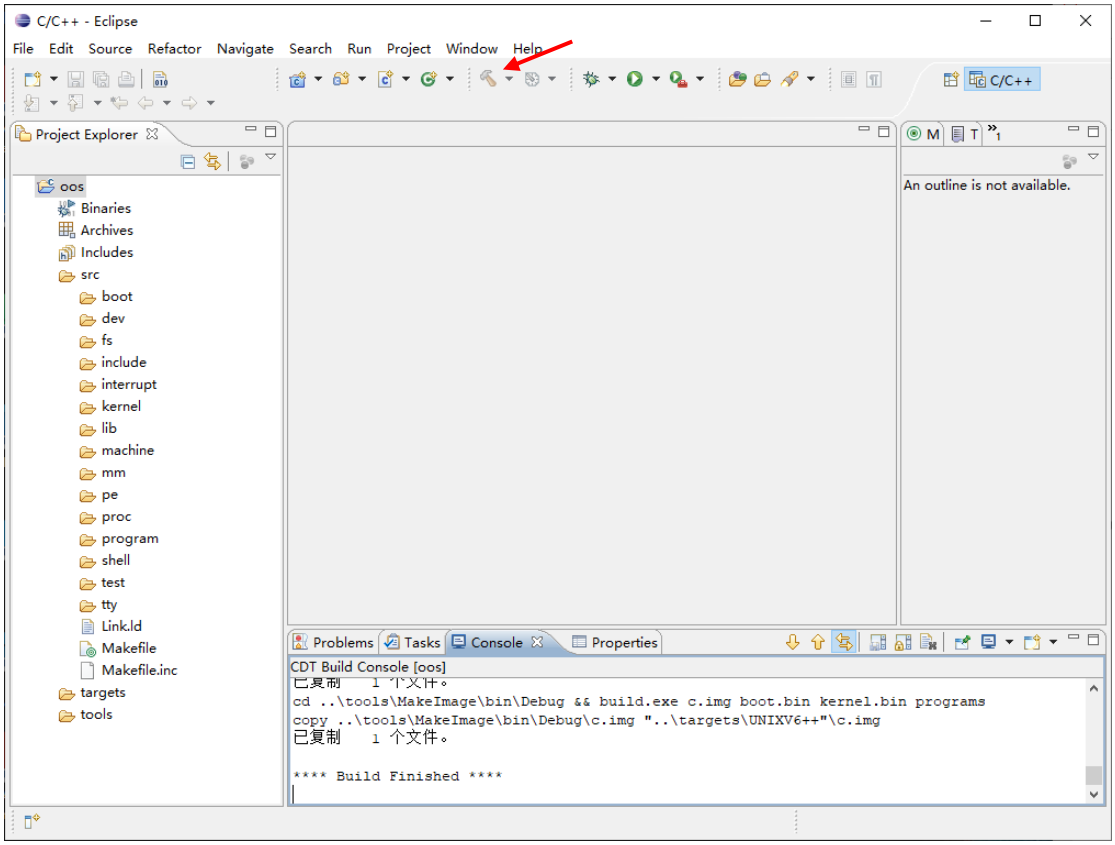


点 OK（其实无所谓）。

现在，我们配好了编译环境。若有打开 Bochs 虚拟机，把它关掉。

老系统确保 windows 实时防护已关闭。新系统不必。

点小锤子，系统会执行批处理命令 all。Console 选项卡中会显示 all 命令的执行日志。一切顺利的话，你会看到日志 Build Finished（老版本）。新版本，见第一步的 3.3。

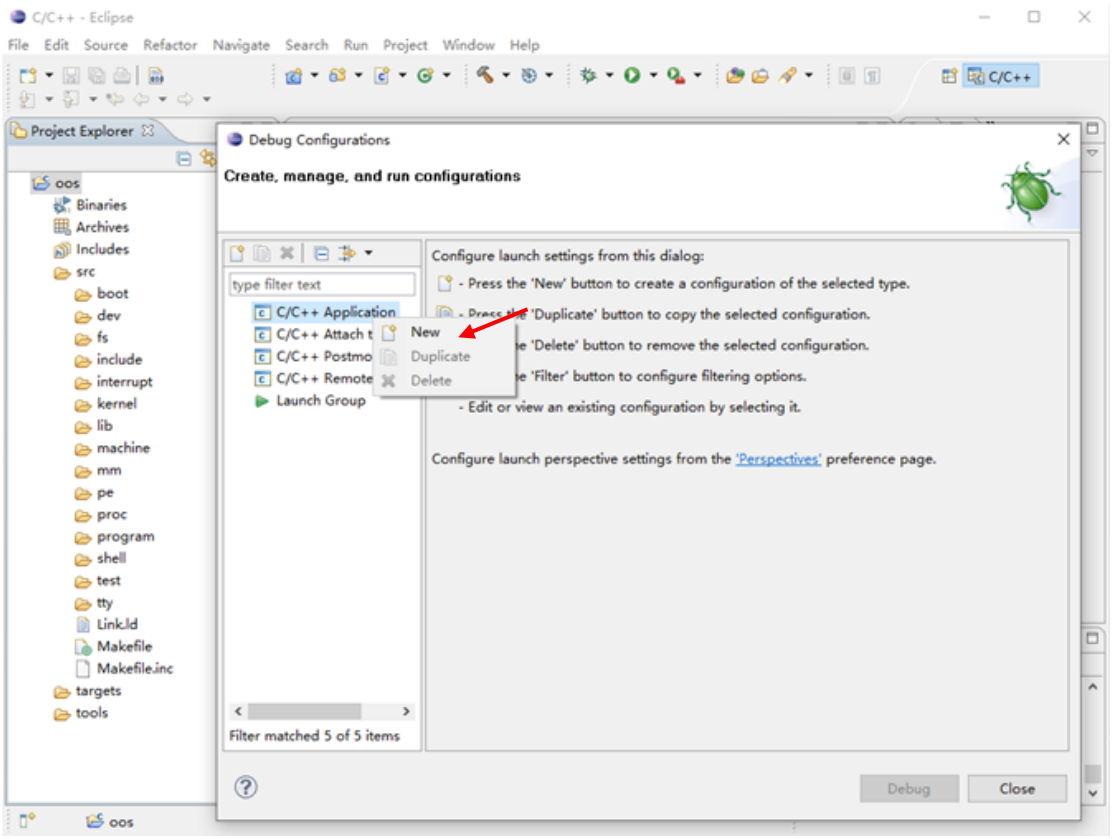


4 为 UNIX V6++配置 gdb 远程调试

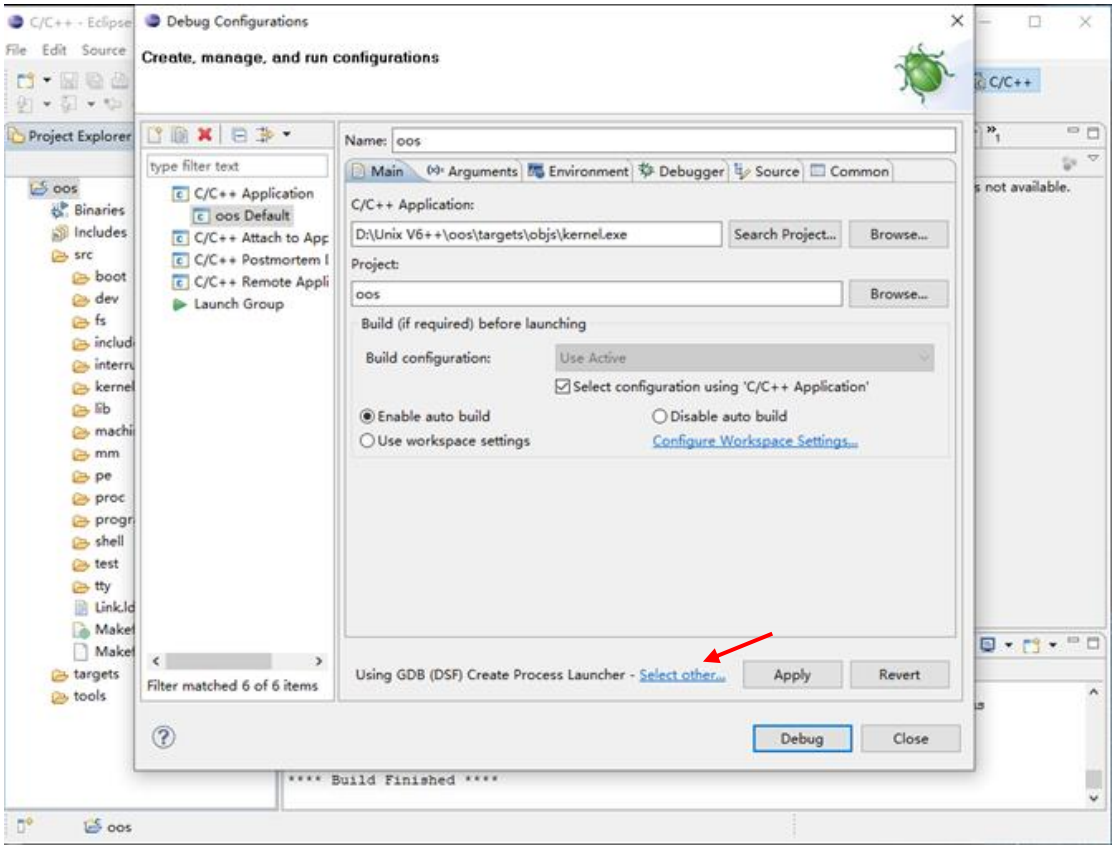
Project Explorer 中选中 oos。

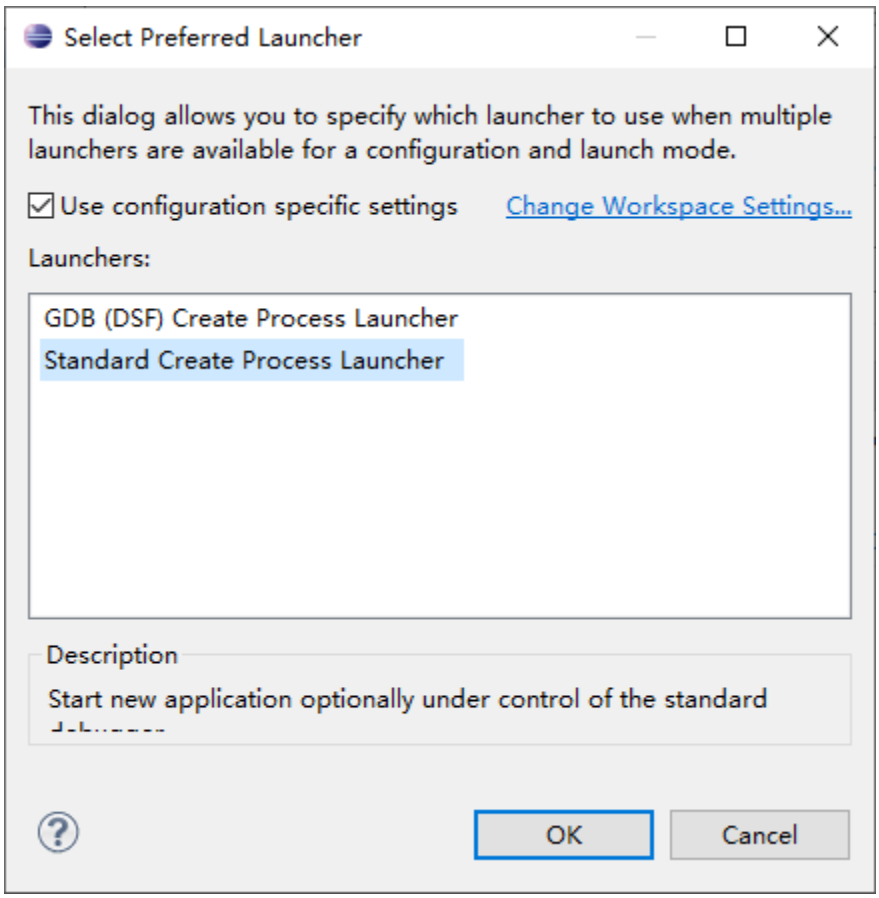
Run-->Debug Configurations：

进入“Debug Configurations”菜单项，新建一个 C/C++ Application 配置（方法：右击 C/C++ Application 选项）

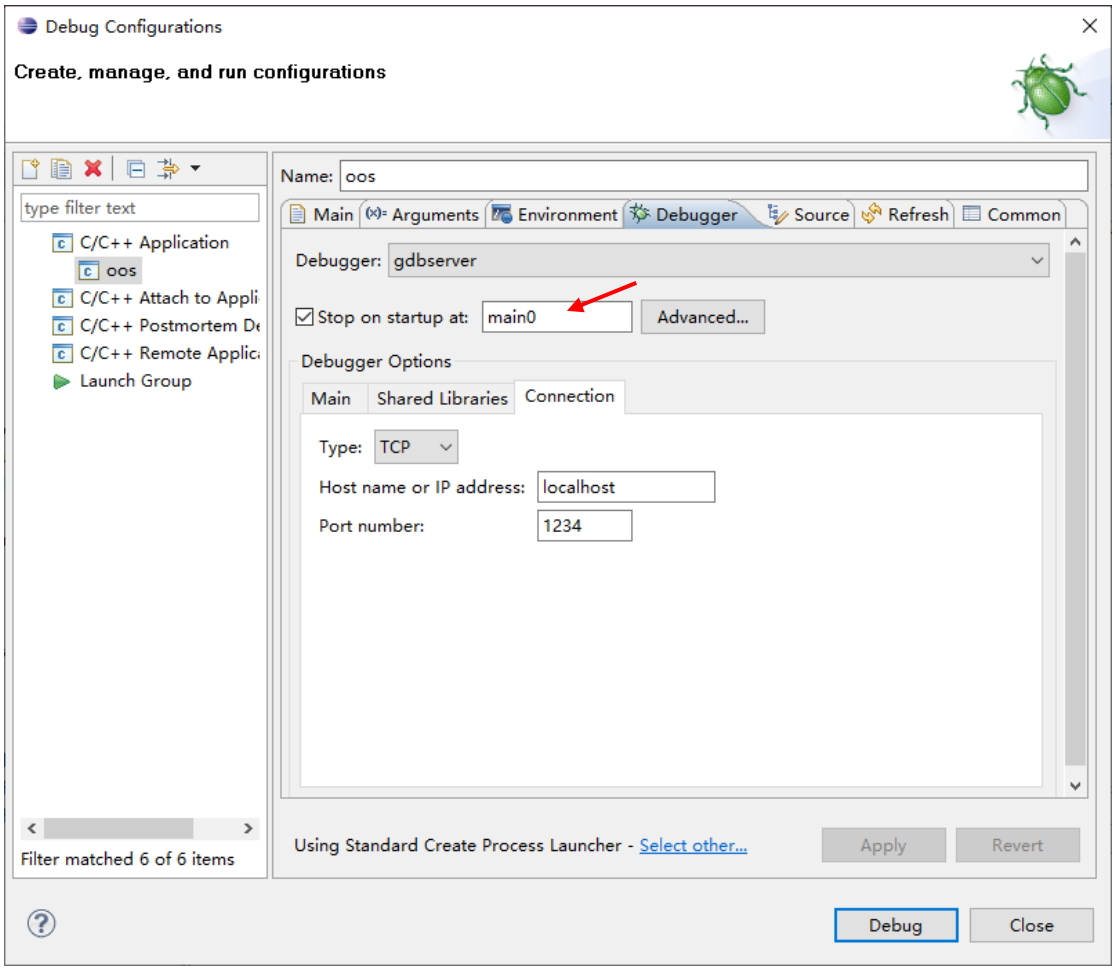


起名“oos”，设置调试对象为 UNIX V6++的 kernel.exe 文件。点选 SelectOther，设置为 Standard Create Process Launcher 方式。


Debug Configurations
Create, manage, and run configurations
Name: oos
Main
Arguments
Environment
Debugger
Source
Common
C/C++ Application:
D:\Unix V6++\oos\targets\objs\kernel.exe
Search Project...
Browse...
Project:
oos
Browse...
Build (if required) before launching
Build configuration:
Use Active
Select configuration using 'C/C++ Application'
Enable auto build
Disable auto build
Use workspace settings
Configure Workspace Settings...
Using GDB (DSF) Create Process Launcher - Select other...
Apply
Revert
Debug
Close
type filter text
C/C++ Application
oos Default
C/C++ Attach to App
C/C++ Postmortem I
C/C++ Remote Appli
Launch Group
Filter matched 6 of 6 items
**** Build Finished ****

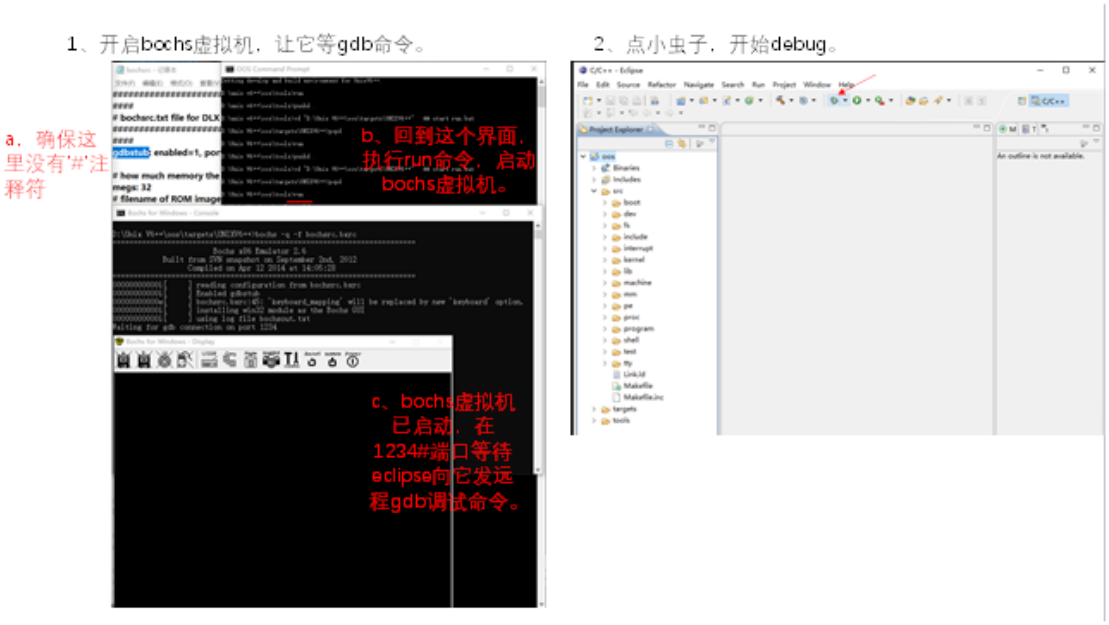


在“Debugger”选项卡中，设置调试器为“gdbserver”，起始调试点为“main0”函数，“connection”中设置连接方式为“TCP”，端口号为“1234”，与 bochs 虚拟机中的设置一致。

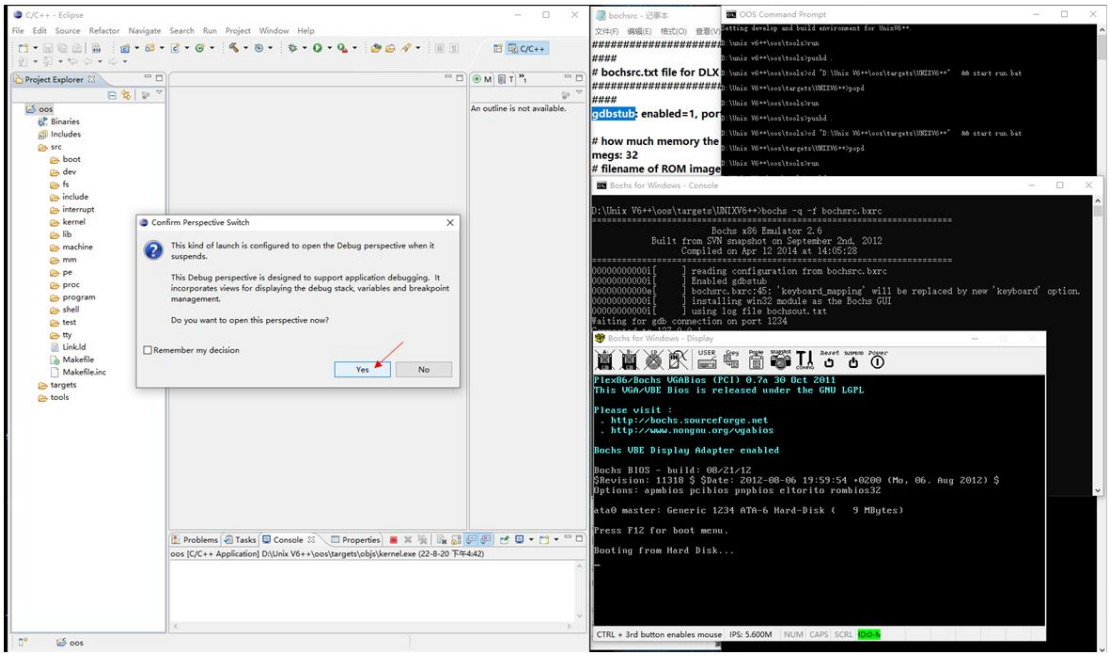


Apply。成功。

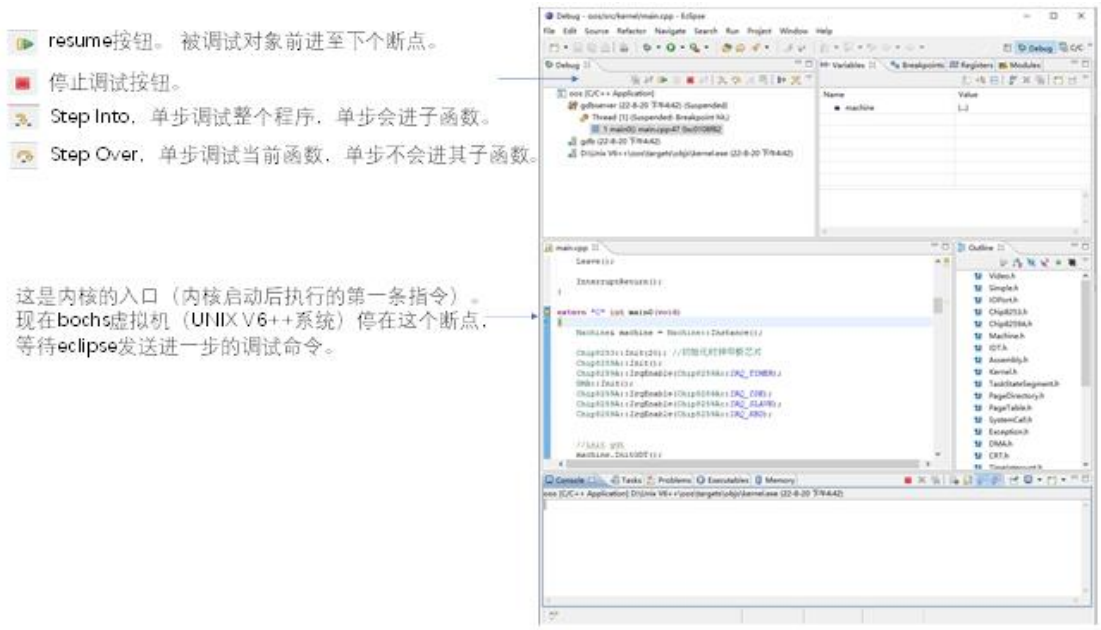
6、调试 UNIX V6++



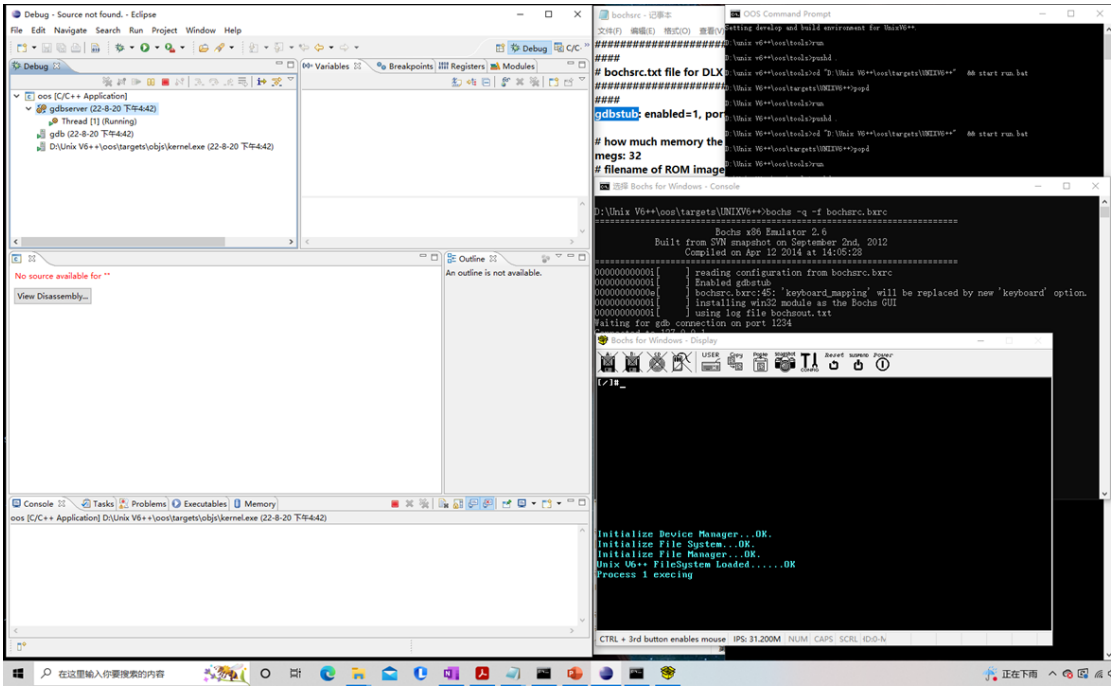
开始调试



eclipse 切换至 debug 视图（下图），可以向 bochs 虚拟机发命令。上图，右侧，是 bochs 虚拟机的状态：完成内核加载，等待 eclipse 向它发调试命令。



我们按 2 次 resume 直至 bochs 虚拟机给出 UNIX V6++的命令行提示符 #



现在，bochs 虚拟机正在执行 shell 程序。它是 UNIX 系统的命令行界面，在等着我们输入 UNIX V6++的命令行。

shell 程序不是我们的调试对象。所以，当前状态下，eclipse debug 过程找不到 debug 模式的被调试对象，看上图左边，它 No source available for ""。。。

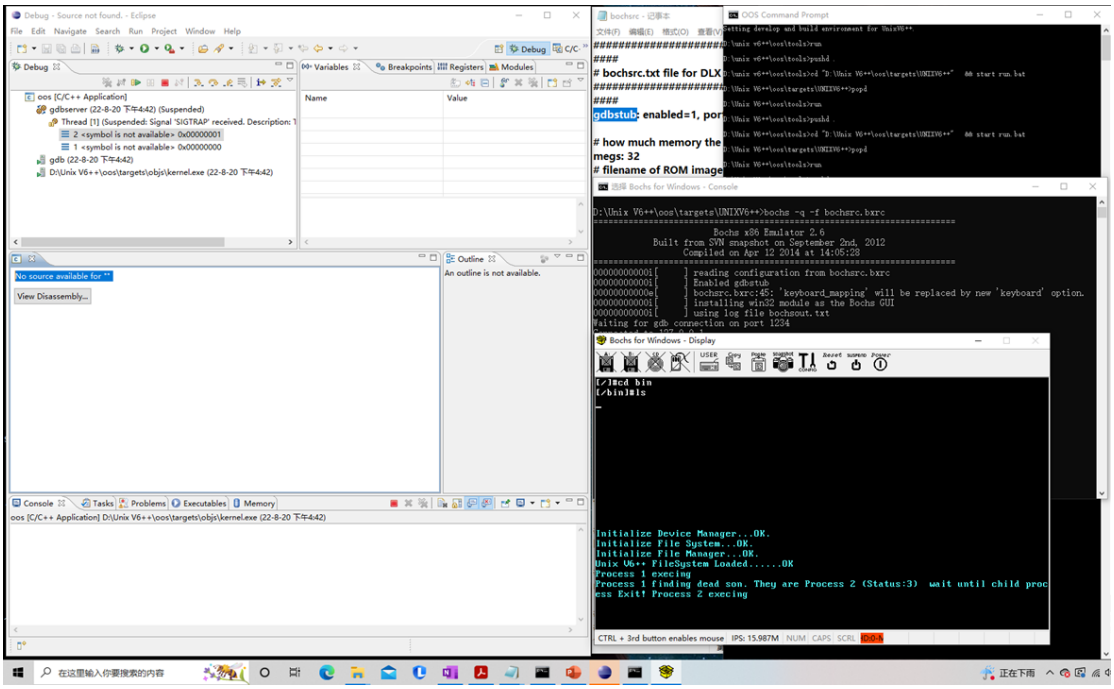
我们在右下侧窗口的命令行界面中输入系统支持的 UNIX 命令。

cd bin

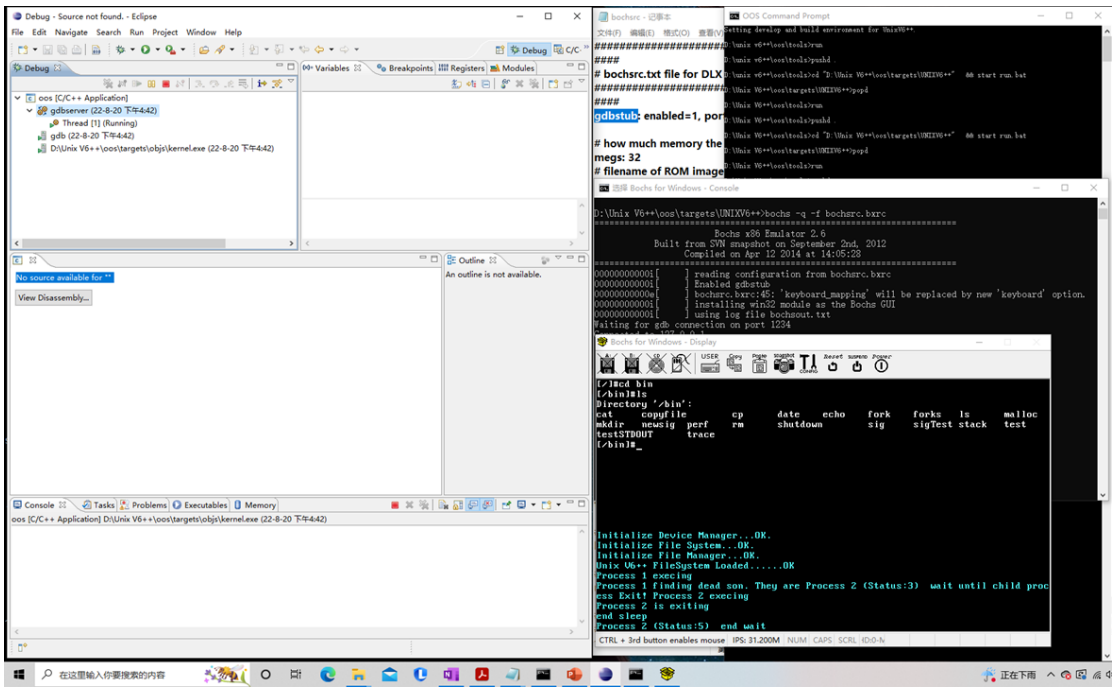
ls

执行 ls 命令时，bochs 虚拟机会从执行应用程序的状态转换到执行内核的状态。会碰到断点。看它又停下来了。

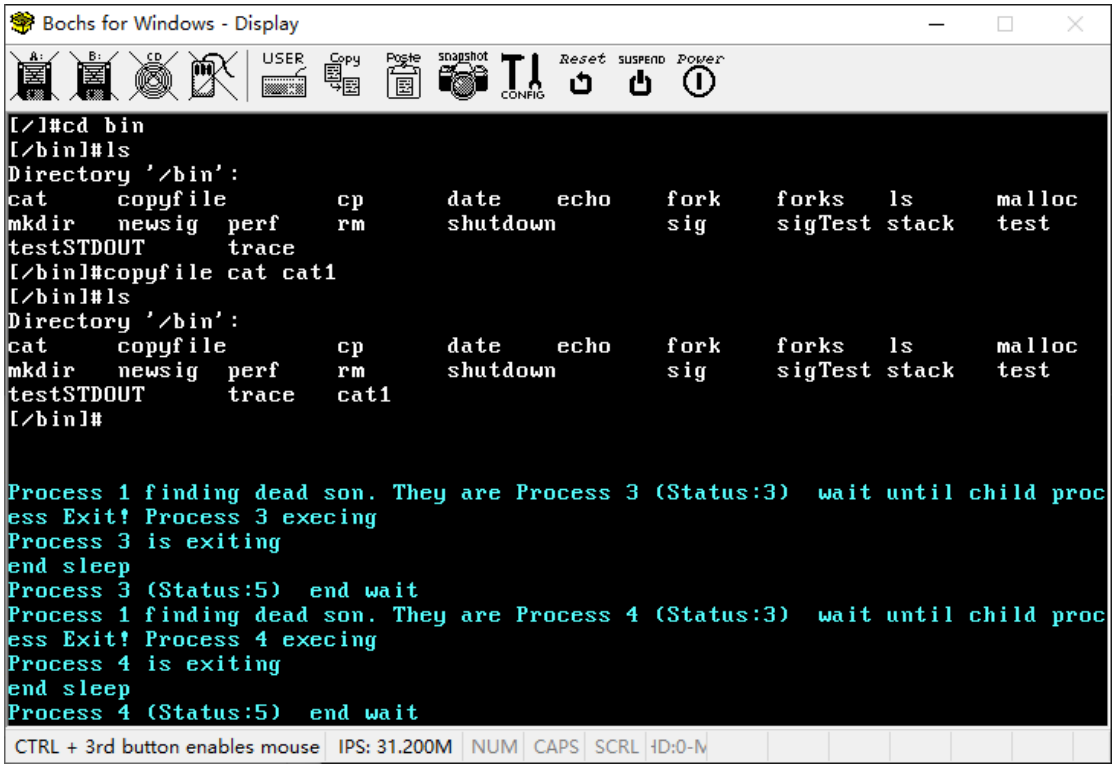
我们按 resume 按钮，让它跑下去。



resume 按钮按下后，bochs 虚拟机的显示屏（右下窗口）上出现了 ls 列目录的结果，这是 UNIX V6++系统已有的全部命令。



试一个。copyfile 命令给 cat 文件做一个复本，cat1。



我们执行命令 shutdown。安全关机，将改动过的文件系统同步回磁盘 c.img。

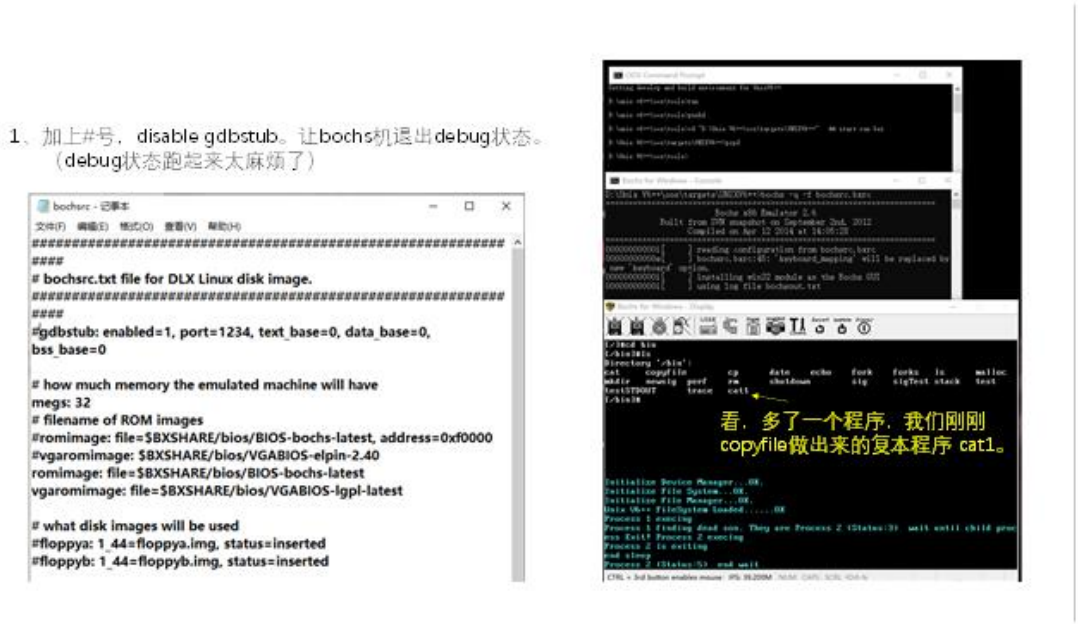
按下 bochs 虚拟机的开关。

按下 eclipse 的停止调试按钮。

这就成功完成了首次 debug 实验。

再一次 run 出来 UNIX V6++，查看它的文件系统。刚刚新建的文件 cat1 还在。

以后，只要你更新过文件系统，记得关机前执行 shutdown 命令。否则，更新不会生效。



删掉没用的程序 cat1

rm -f cat1

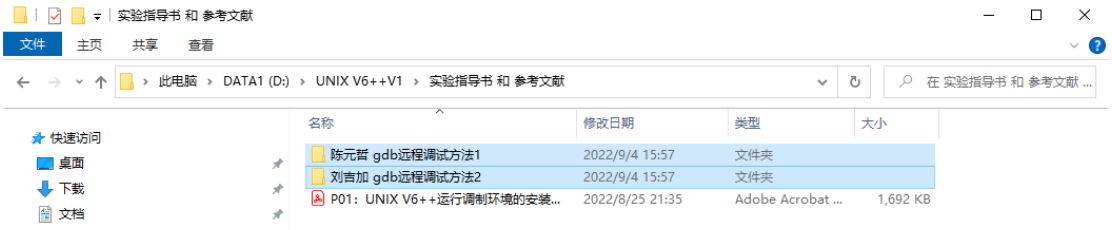
shutdown

7、本部分实验报告要求：

7.1 在 eclipse 中观察整个 UNIX V6++源代码目录结构，根据你的理解，尝试给出每一个 oos/src/下的子目录中所包含文件的用途。（1 分）

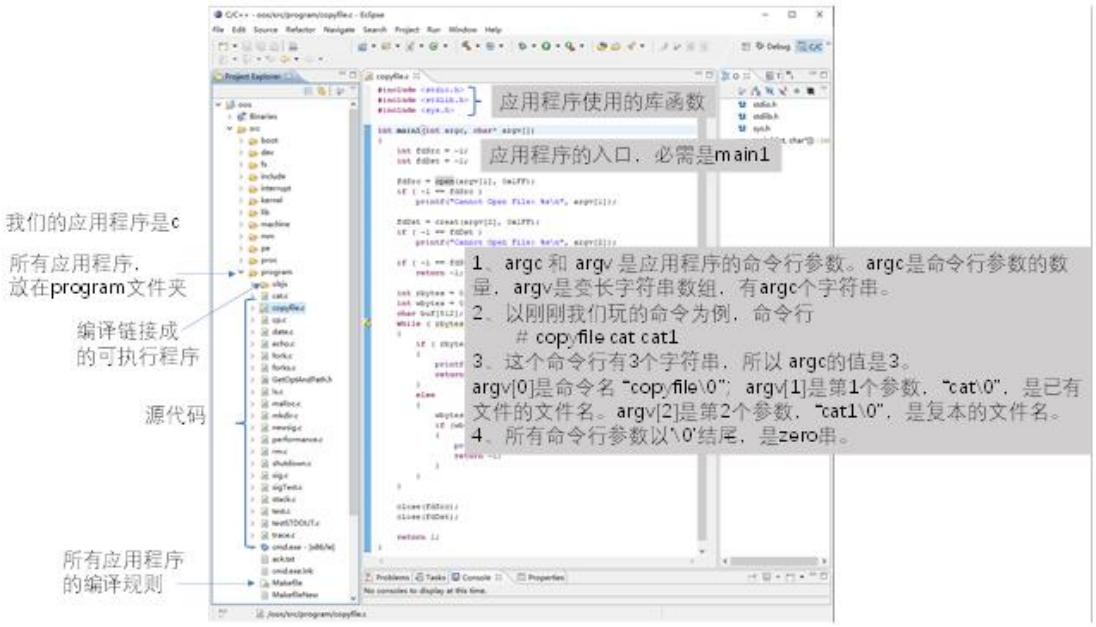
7.2　搭建基于 eclipse 的 gdb 远程调试开发环境。截图关键步骤。（1 分）

7.3　通过调试环境的配置和运行，并查阅相关资料，谈谈你对远程调试的理解。（1 分）

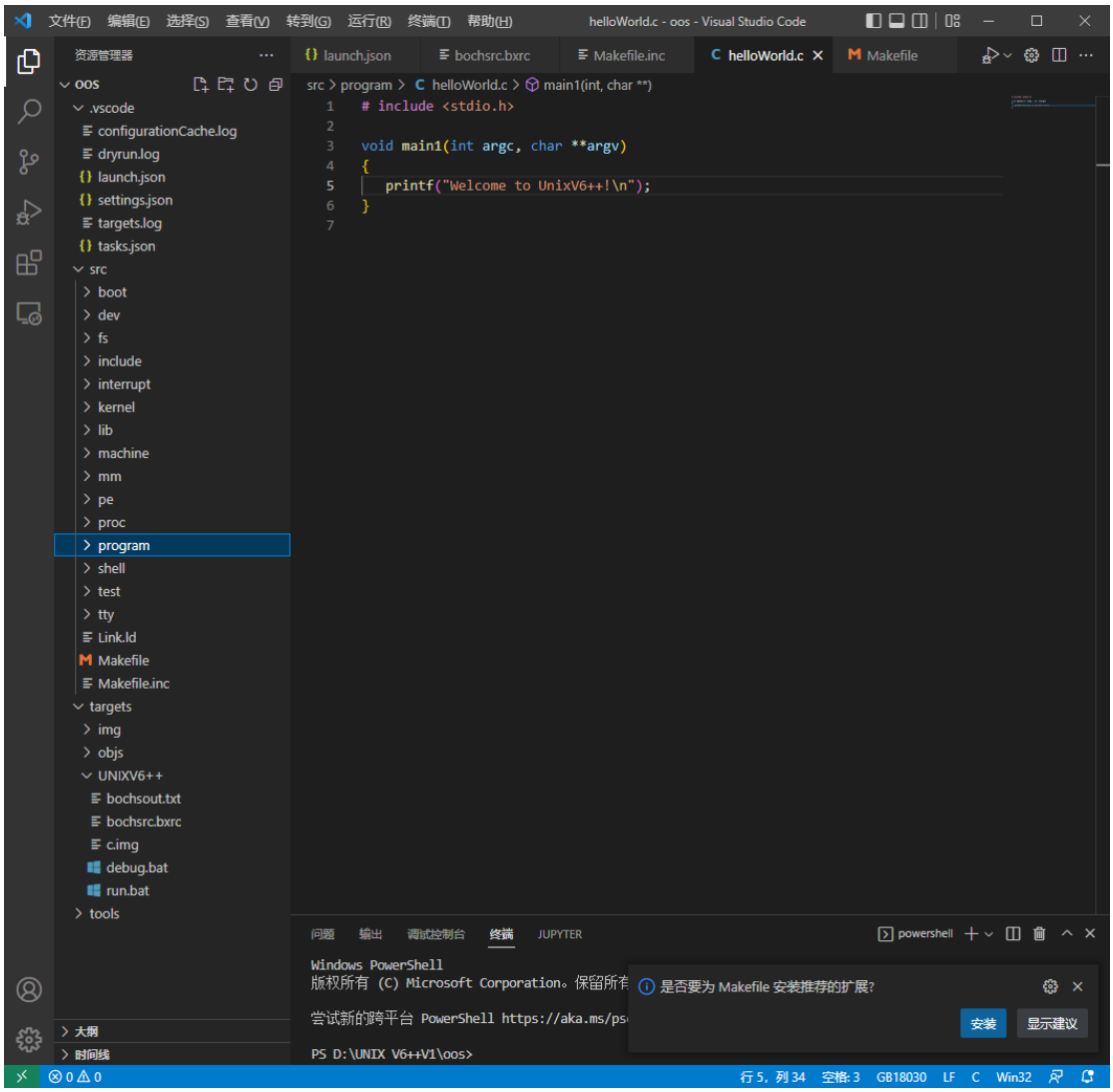


Part3，编写一个新的应用程序 HelloWorld

1、观察已有的应用程序

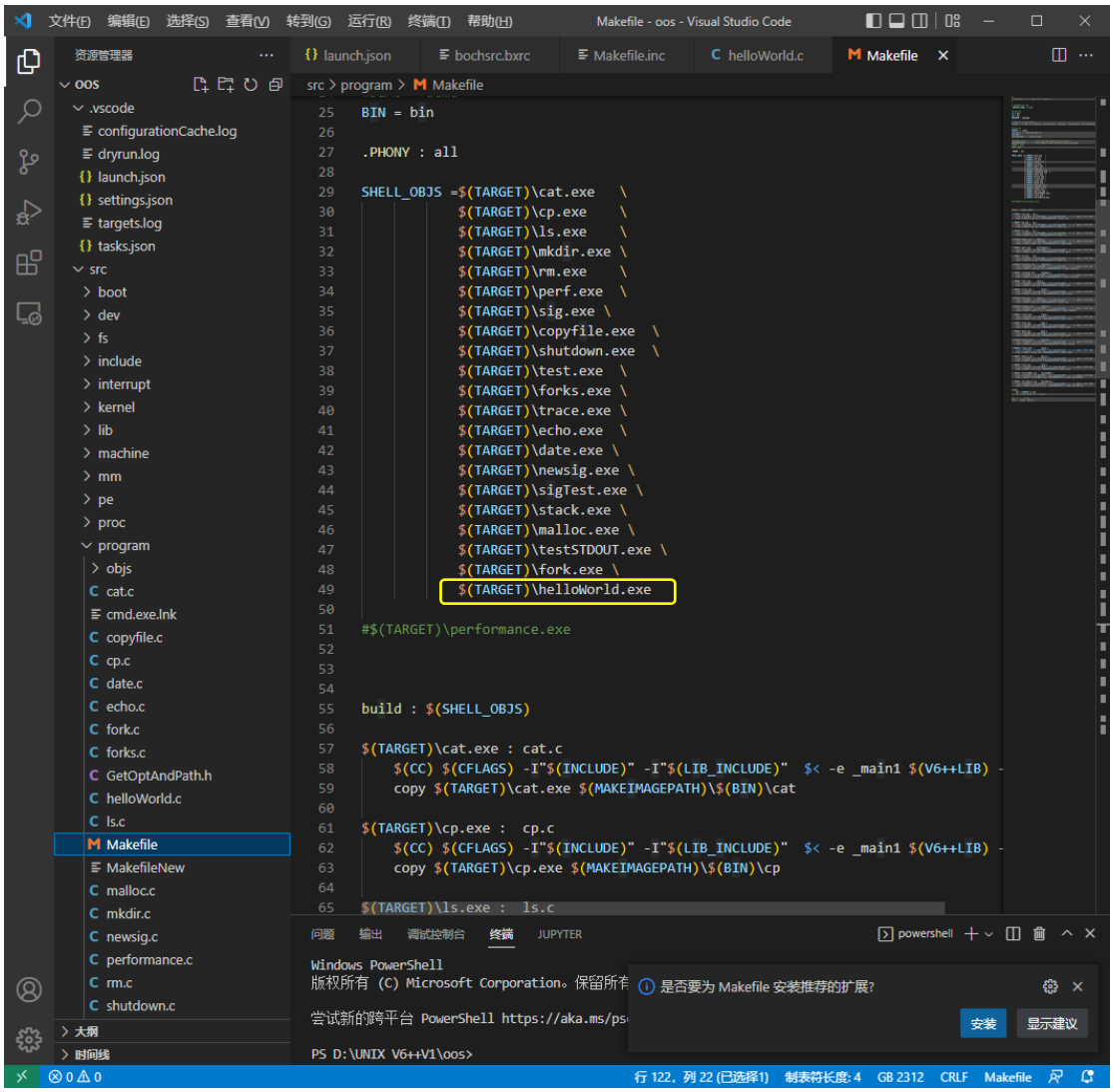


2、新建源文件 helloWorld.c

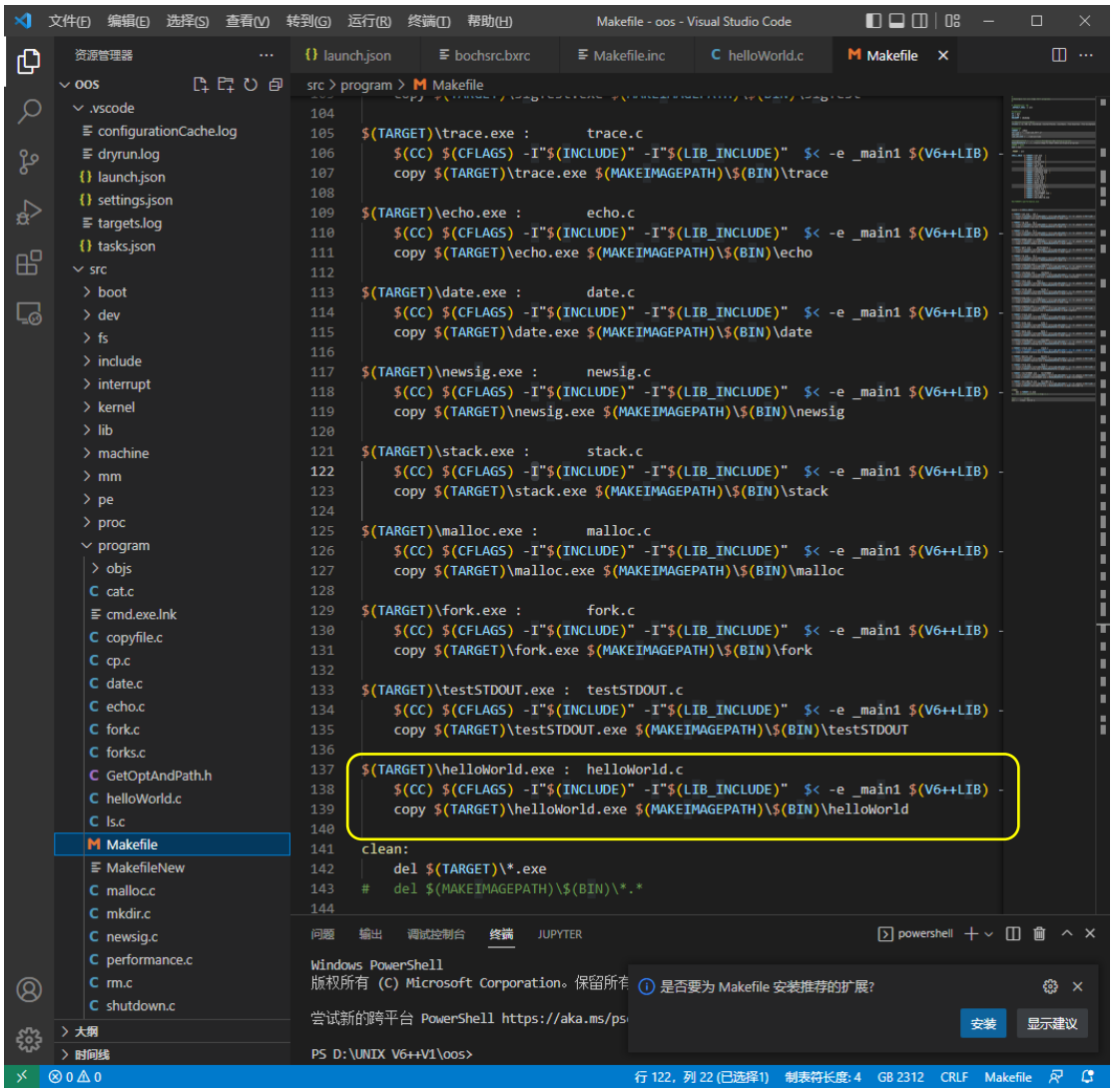


3、修改 program 目录下的 makefile 文件。该文件是 program 目录下所有应用程序的编译规则。要修改 2 处。

第 1 处：



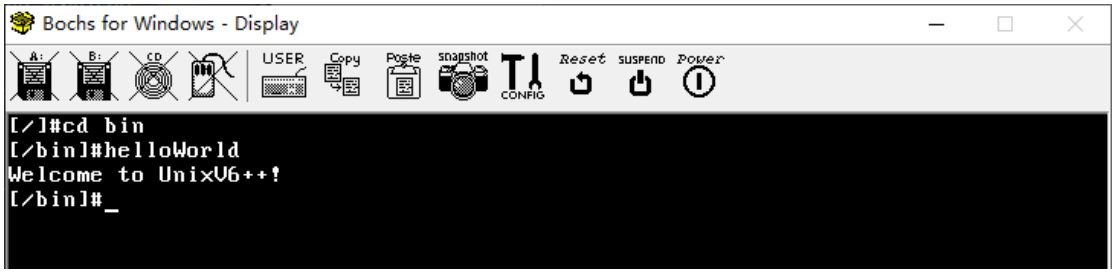
第二处：加这个新应用的编译规则



4、执行 helloWorld 程序

执行 part 1，步骤 3.3 中的 all 命令，重新制作系统盘 c.img。

步骤 3.4，run，开启 bochs 虚拟机。可以看到新程序 helloWorld 已经是 Unix V6++系统中的新命令了。实验成功。



5、本部分实验报告要求。

5.1　为 Unix V6++系统加入新应用 helloWorld。截图关键步骤。（1 分）

Part 4、第二个 Unix V6++系统

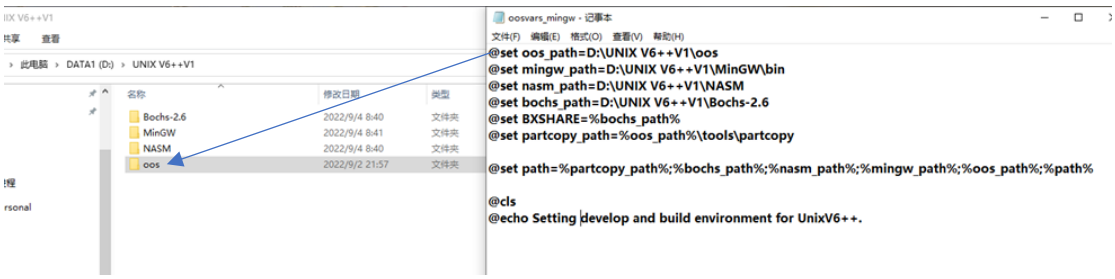
简介：

- eclipse 和 jdk，是旧版的集成开发工具。这个集成开发环境的作者是 计科 09 陈启航，我们用了很多年，文档和实验步骤非常成熟。建议，先装这个环境，用来读源代码。但这套环境，不是很好维护 2 套以上的 Unix V6++系统。
- VSCode，上学期 软院陈元哲 和 信安刘吉加 等两位同学探索出基于 VSCode 的集成开发环境。这套环境特别好用，能够方便地维护 2 套以上的 Unix V6++系统。缺点是，文档还不完善。

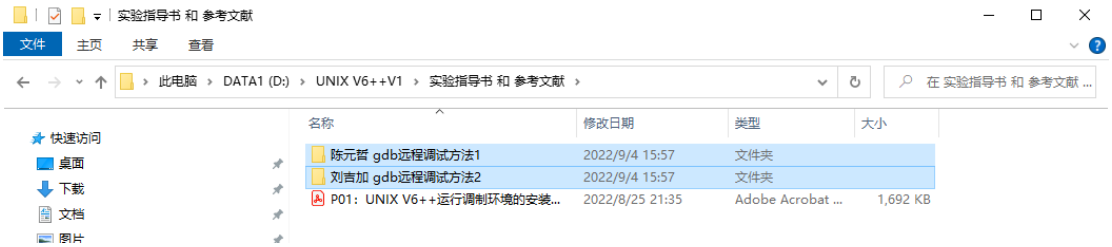
1、复制一份 oos 包。这样我们就有了自己的开发版。不必复制 NASM，MinGW，和 Bochs-2.6。开发版系统和稳定版系统共用一套依赖包。

2、用 code 打开它。就可以查看、修改自己的开发版的源代码了。

3、oos_path 指向开发版的 oos 目录。其余环境变量不动。



3、配置 gdb 远程调试，参考👇。先看陈元哲的。再看刘吉加的。



4、这是附加题（+5 分）。没有提交的截止时间。提示，上届荣誉班，学长学姐全有这个环境。