Xcode笔记

初识Xcode

作为Mac上相对来说最好的IDE，我觉得还是有必要学习一下的。

新建C++项目

1. 欢迎界面: `Create a new Xcode project` -> `macOS` -> `Command Line Tool` 。
2. `Product Name` 和其他几行随便填， `Language` 选 `C++` 。
3. 选则项目位置后进入项目，在界面左侧可以看到文件目录。

利用lldb进行debug

有时程序出现类似Linux下 `Segment fault: 11` 的问题，不知道出错在哪。我们可以利用Xcode自带的**lldb**代替Linux下的**gdb**来debug，Mac下的gdb简直有毒，建议不要安装。

1. 鼠标**左键**点击代码前的**行数**即可以这一行的开头(**这一行代码未执行**)设置断点 `Breakpoint` ，再左键点一次可以 `Disable Breakpoint` ,也可以右击进行编辑和删除等操作。
2. `Command+R` 或者点击**运行**按钮，在代码下方会出现 `debug area` 和 `output area` ，在 `debug area` 里会追踪各个**变量**的值，甚至可以看到数组中每个值的大小。
3. 点击 `debug area` 上方第三个小图标 `Continue program execution` ,继续执行程序直到遇到下一个断点，可以根据 `debug area` 中各变量数值的变化来判断程序出错位置。

Xcode与GitHub

- 在安装配置完 `git` 后，我们可以在Xcode的 `Preference` -> `Accounts` 中添加GitHub账户，随后我们就可以通过菜单栏上的 `Source Control` -> `Clone` 下载自己在GitHub上托管的代码了。

- 有时 git clone 下来的代码里没有Xcode的工程文件 `.xcodeproj` ,我们需要新建一个工程项目Project(**新建Project时取消勾选** **`Create Git repository on my Mac`**)，然后在左侧文件目录里删除初始文件夹，关闭Xcode项目窗口，将clone下来的文件夹拖至Project文件夹中，然后重新点开工程文件 `.xcodeproj` ，**右击**左侧文件目录最上方的**工程文件**或者点击最左下方的 `+` 号，选择 `Add files to "PROJECTNAME"...` ，弹出窗口中的 `Added folders:` 后选 `Create groups` (一般默认就是这个)，添加完成后就可以直接在 `Source Control` 里 git commit 和 git push 到 GitHub 上了。

- 如果是一个新的Project想传到GitHub上去，可以点击左侧文件目录上方的**第二个小图标** **`Show the Source Control navigator`** , 会显示 git 的相关信息，包括**Branch**、**Tags**以及**Remotes**, 然后**右键**第一个项目文件夹 -> `Creat "PROJECTNAME" remote...` 就会弹出一个类似 GitHub 新建 Repository 的窗口，填完就会自动在 GitHub 创建对应的 Repository，以后也可以直接在菜单栏 `Source Control` 里 git commit 和 git push 了。

Xcode下Python环境的搭建

1. 欢迎界面: `Create a new Xcode project` -> `Cross-platform` -> `External Build System` 。

2. `Product Name` 和其他几行随便填， `Build Tool` 需要找到 `python` 的位置，可以在 Terminal 中输入 `which python` ,找到python的位置。

3. 菜单栏 `Product` -> `Scheme` -> `Edit Scheme` 。

4. 弹出窗口中右侧选 `Run` ,左边先点 `Info` 选项卡，将第二个 `Executable` 选则 `Other...` 在弹出窗口中找到刚才的python；然后取消勾选下面一行的 `Debug executable` 避免出错。

5. 点击 `Arguments` 选项卡，在 `Arguments Passed On Launch` 里添加**目标py文件**(eg: test.py，类似C++里的main.cpp，只要添加这一个就行了)。

6. 点击 `Options` 选项卡，勾选 `Working Directory` 后面的 `Use custom working directory:` ,然后找到**目标py文件**所在文件夹，然后 `close` 就OK了。

7. 创建**目标py文件**(eg: test.py)，后续可以添加其他py文件，只要全部import到**目标py文件**里就行了，不用重新配置 `Scheme` 。