**AUBO SDK环境部署**

程序安装

环境配置及测试用例的编写

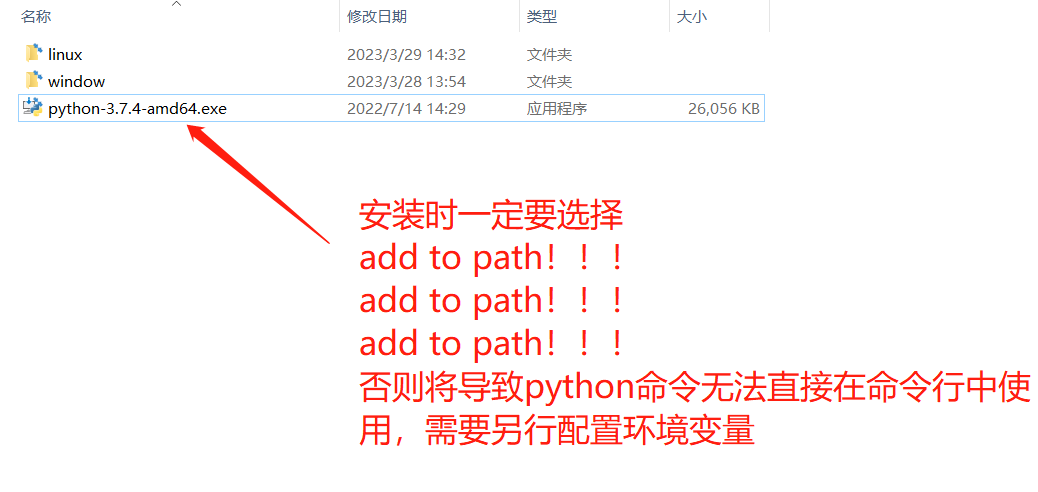
# 1、软件安装

## 2.1 WIN

### 2.1.1 Python（安装基础python环境，安装PyCharm）

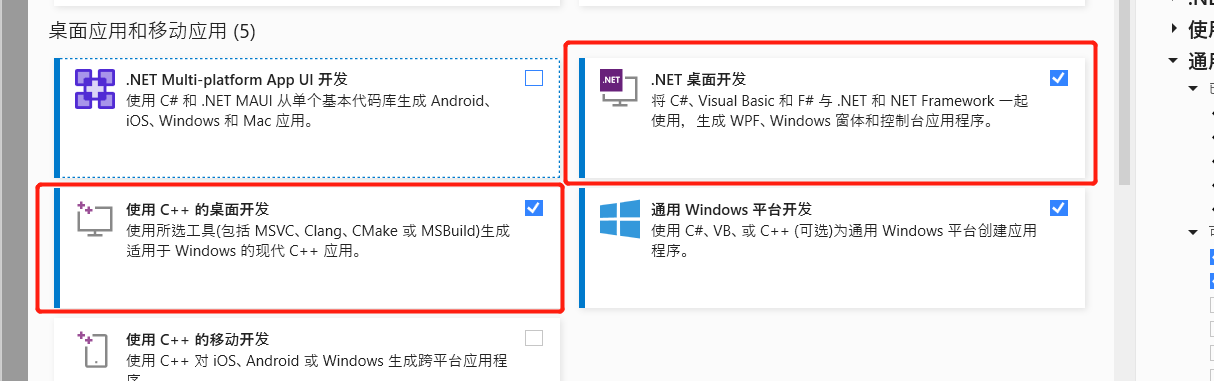
安装教程链接：[python环境配置](https://blog.csdn.net/m0_48912797/article/details/119245039?spm=1001.2014.3001.5502)

Tips：Python版本优先选择包内提供的3.7版本安装包，之后按照链接中配置相应的Pycharm环境即可



### 2.2.2 Visual Studio

官网链接：<https://visualstudio.microsoft.com/zh-hans/>



vs中选择如上两个包进行安装

## 2.1Linux

AUBO虚拟机中自带QT，可以直接运行使用

基础代码的使用

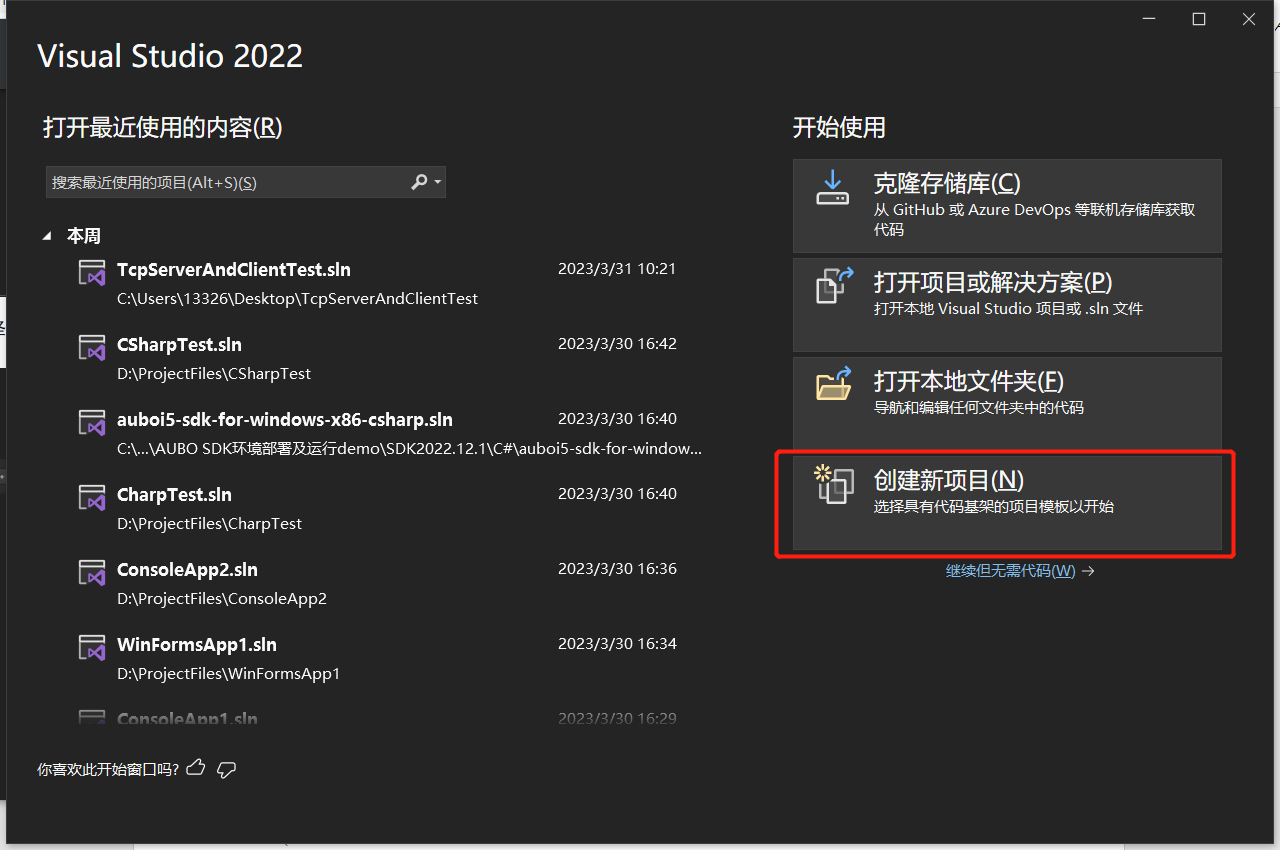
# 项目运行起来的一些支持包



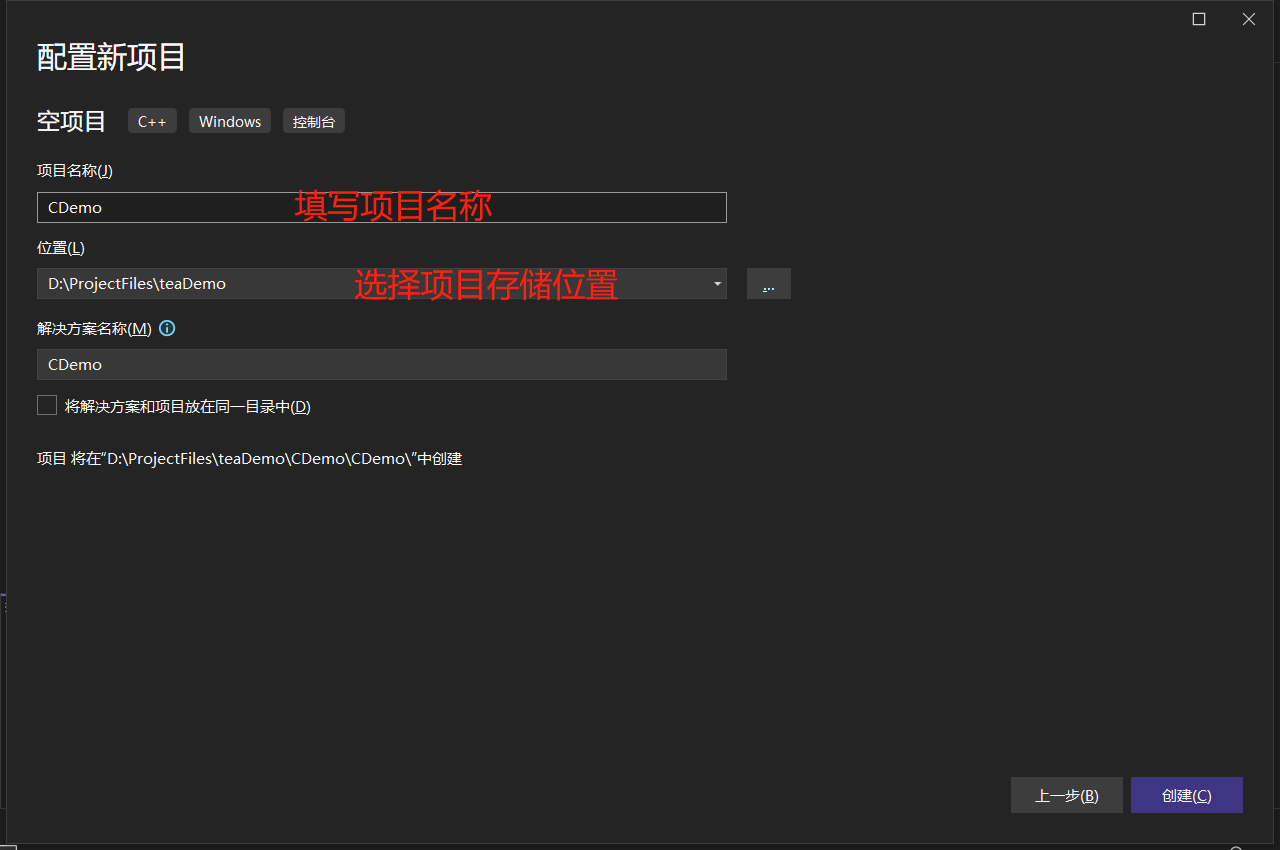
# 2、sdk环境配置

## 2.1 C项目创建

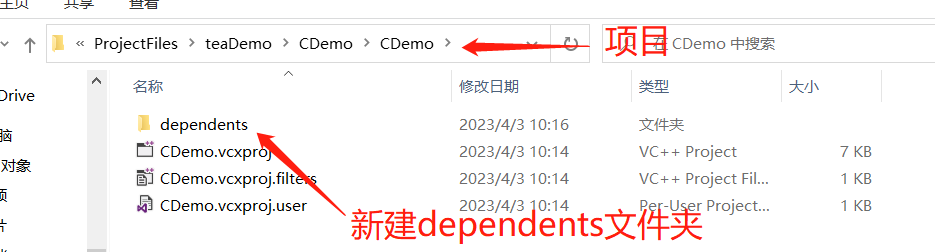
### 环境引入



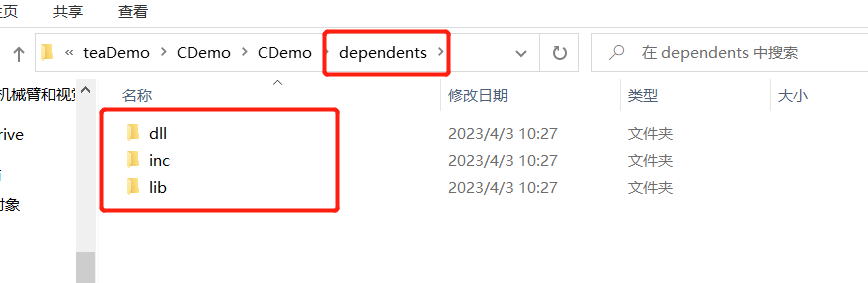




打开项目所在的文件夹，新建文件夹，命名为 dependents，用来存放库文件等依赖。

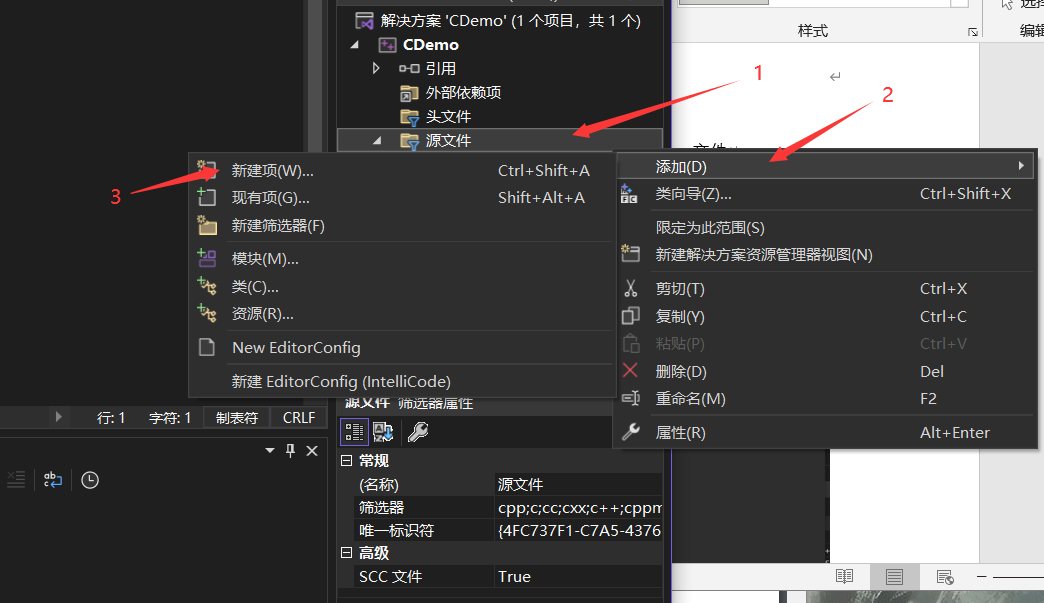


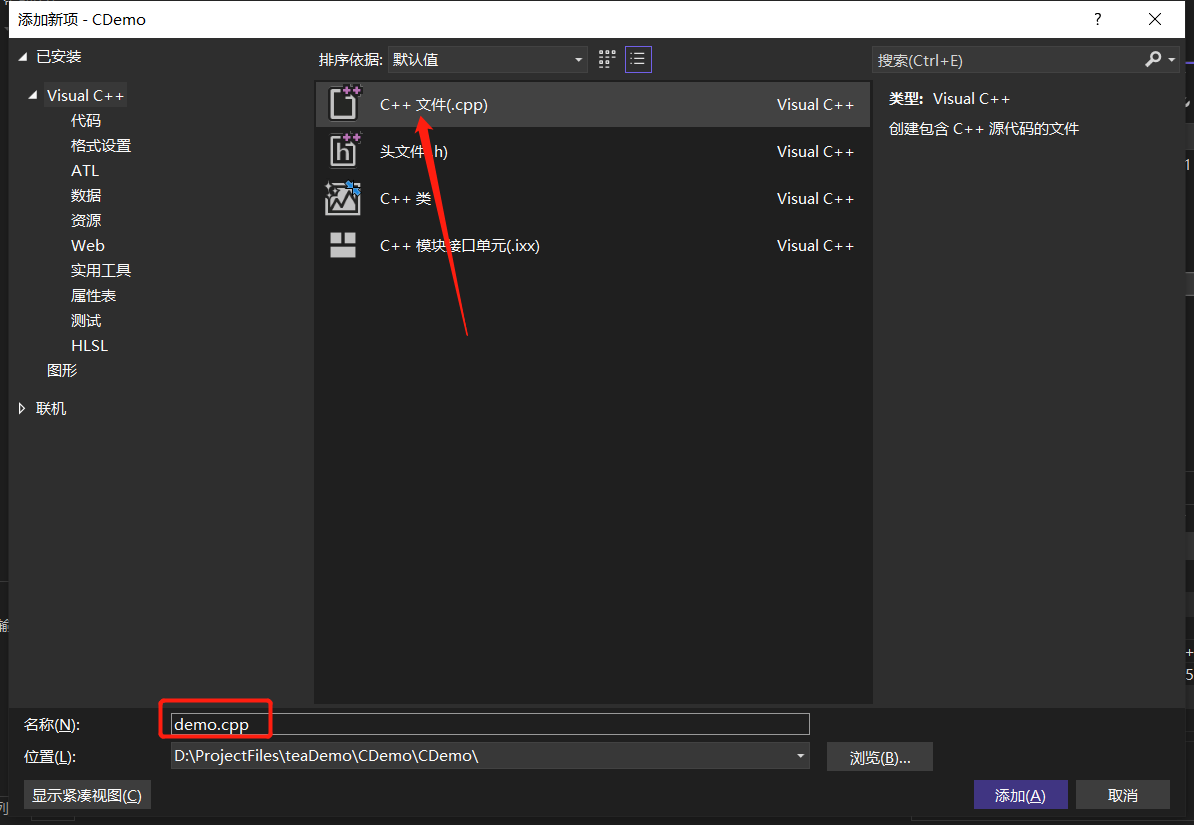
根据自己电脑配置，选择对应文件夹下的dll，inc，lib，复制到dependents文件夹下：



进入项目，配置环境

在窗口右侧，选择源文件（右键）-> 添加 -> 新建项，新建一个 .cpp 文件

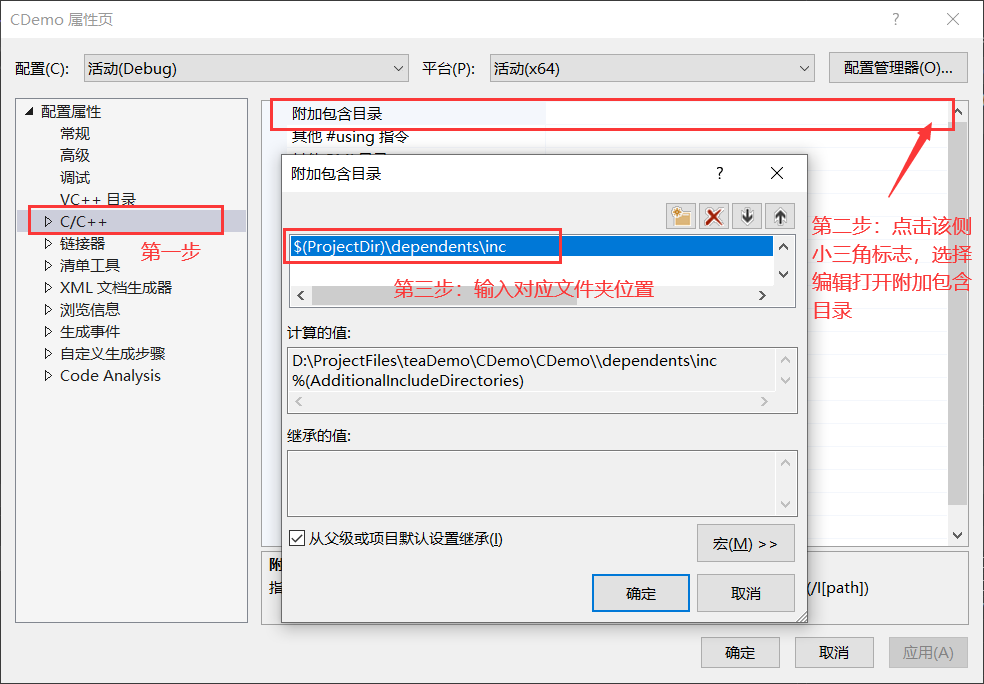




菜单栏选择项目 -> 属性（会有自己新建的项目名作为前缀）

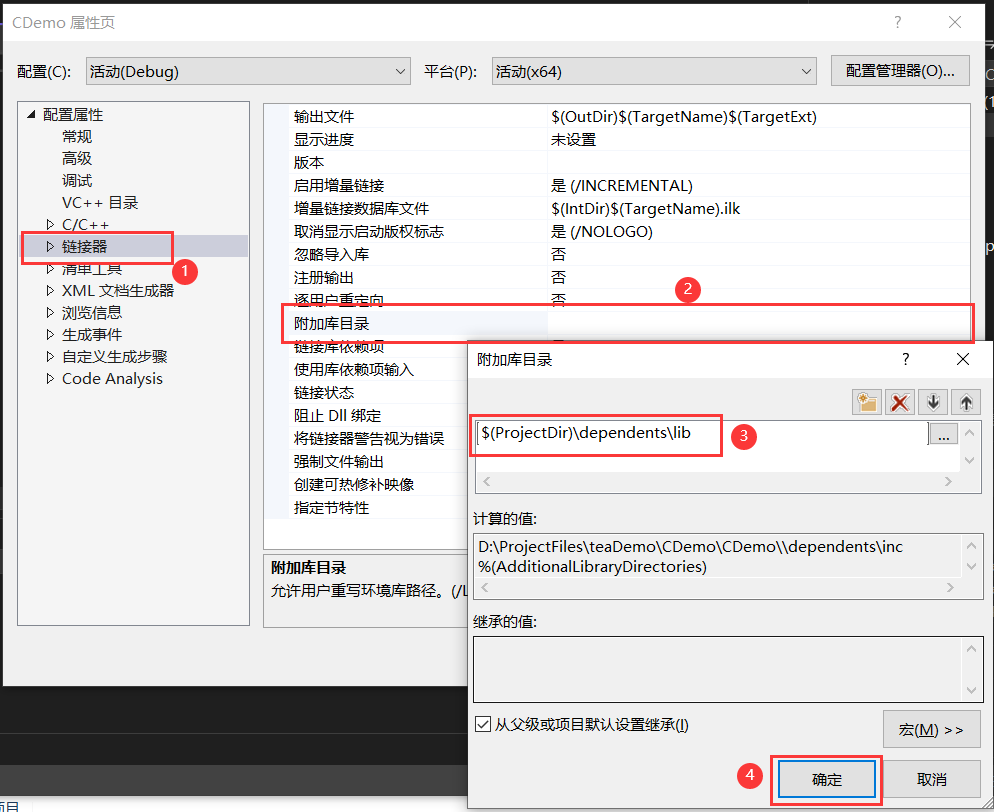
添加工程的头文件目录：点击“C/C++”→“附加包含目录”→添加头文件所在的

路径→点击“确定”。



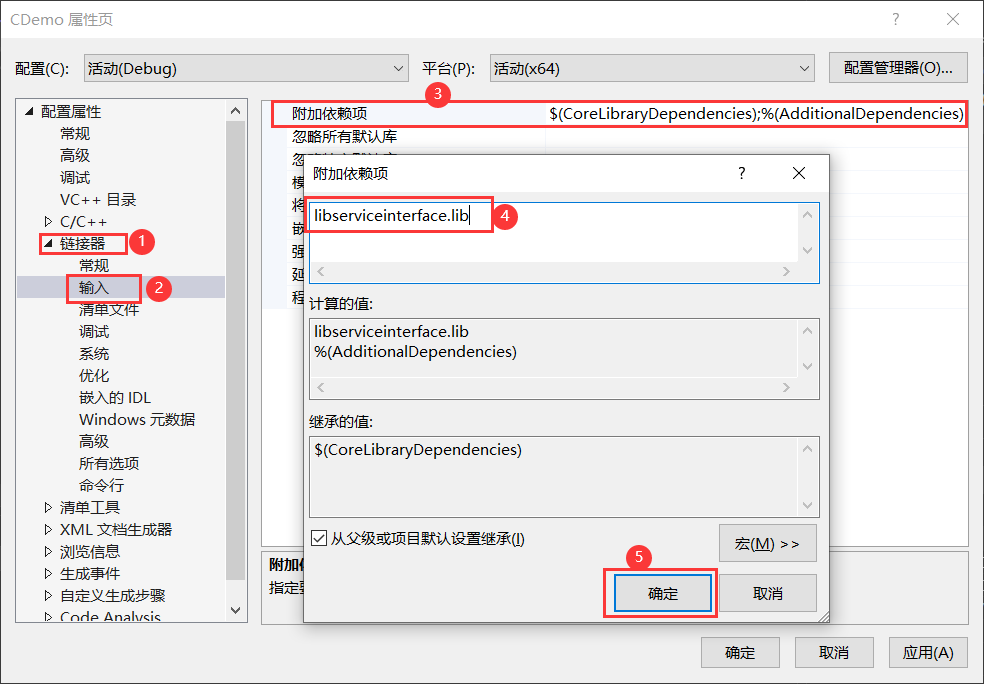
$(ProjectDir)\dependents\inc

添加 lib 库文件：“链接器”→“附加库目录”→添加路径→点击“确定”。

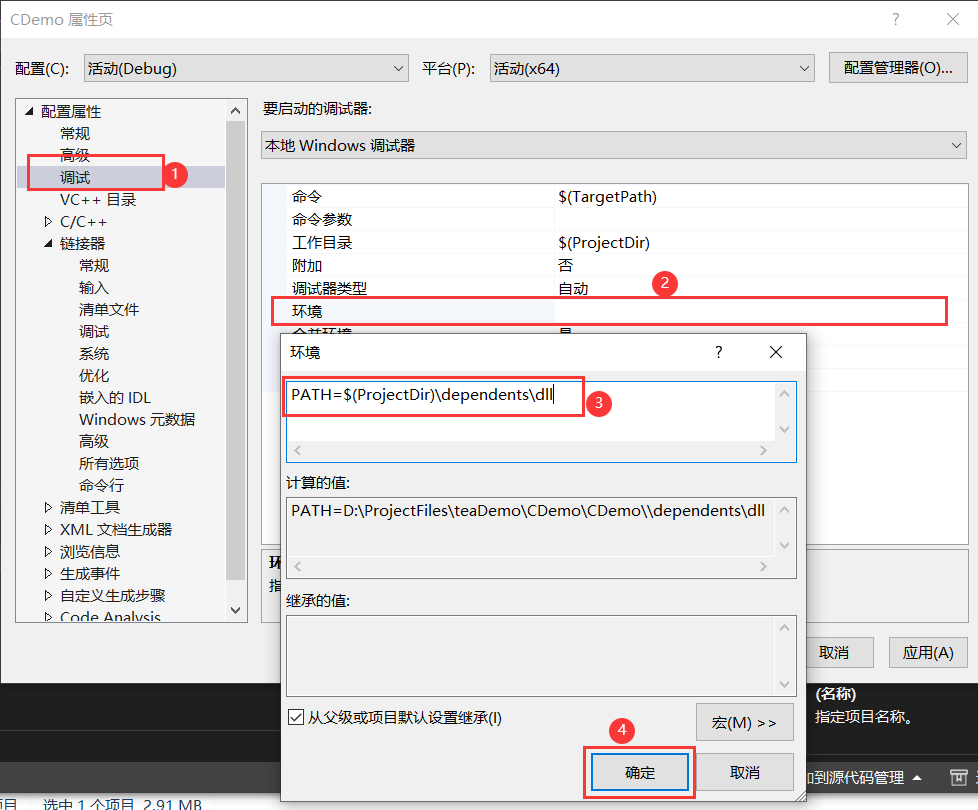


$(ProjectDir)\dependents\lib

“链接器”→“输入”→“附加依赖项”→输入 lib 库文件名→点击“确定”。



添加 dll 文件：“调试”→“环境”→添加路径→点击“确定”→点击“确定”。



PATH=$(ProjectDir)\dependents\dll

至此，环境配置成功

### 示例代码

编写测试用例进行测试：

#include "rsdef.h"

#include <iostream>

#include <string>

#define M\_PI 3.1415926

#define ROBOT\_ADDR "192.168.47.128"

#define ROBOT\_POST 8899

RSHD rshd = -1;

int main() {

//初始化接口库

if (rs\_initialize() == RS\_SUCC) {

std::cout << "初始化成功" << std::endl;

//创建上下文

if (rs\_create\_context(&rshd) == RS\_SUCC) {

//登陆机械臂服务器

if (rs\_login(rshd, ROBOT\_ADDR, ROBOT\_POST) == RS\_SUCC) {

//登陆成功

std::cout << "登陆成功" << std::endl;

}

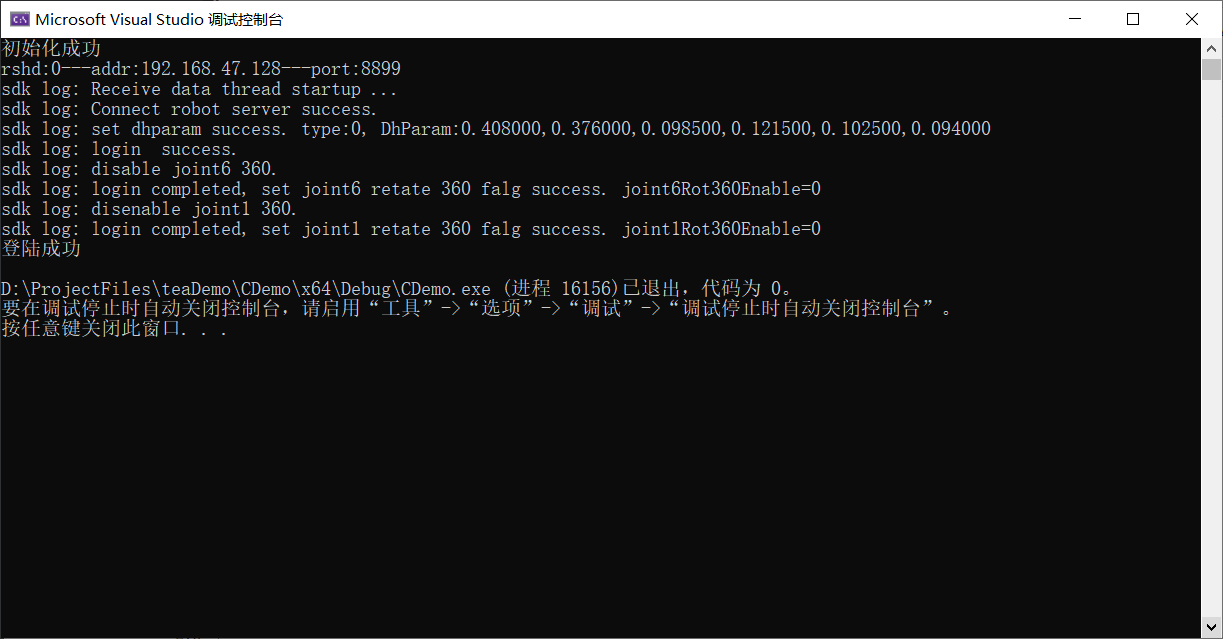
}

}

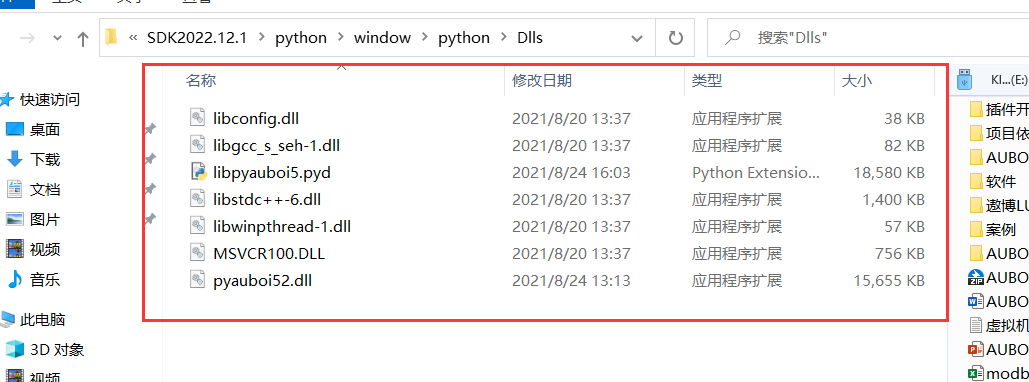
}

注意点：根据虚拟机的ip地址，修改测试用例

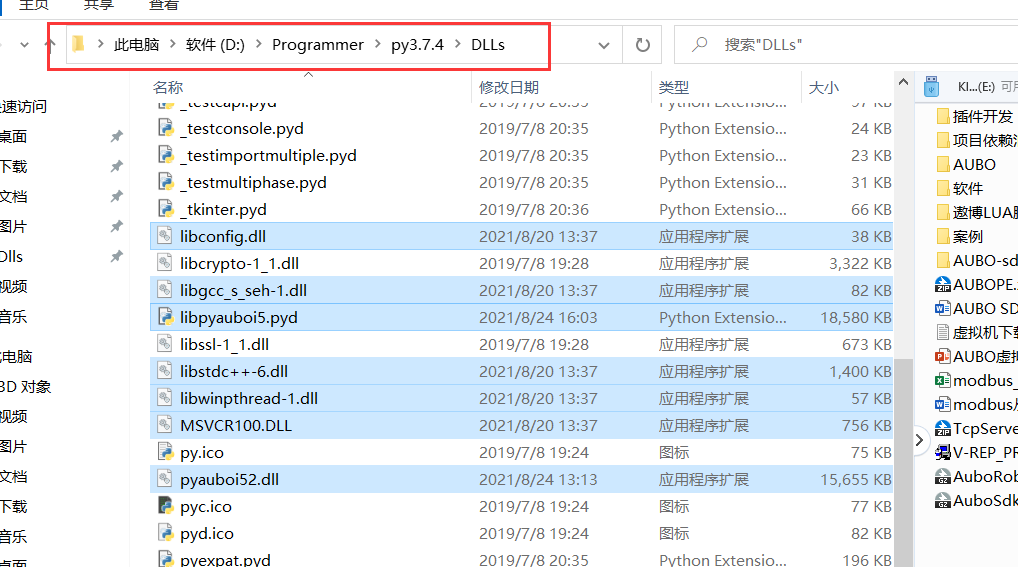
启动测试用例，如下图打印出“登录成功”的语句，说明环境配置成功。



## 2.2 Python（Win）项目构建



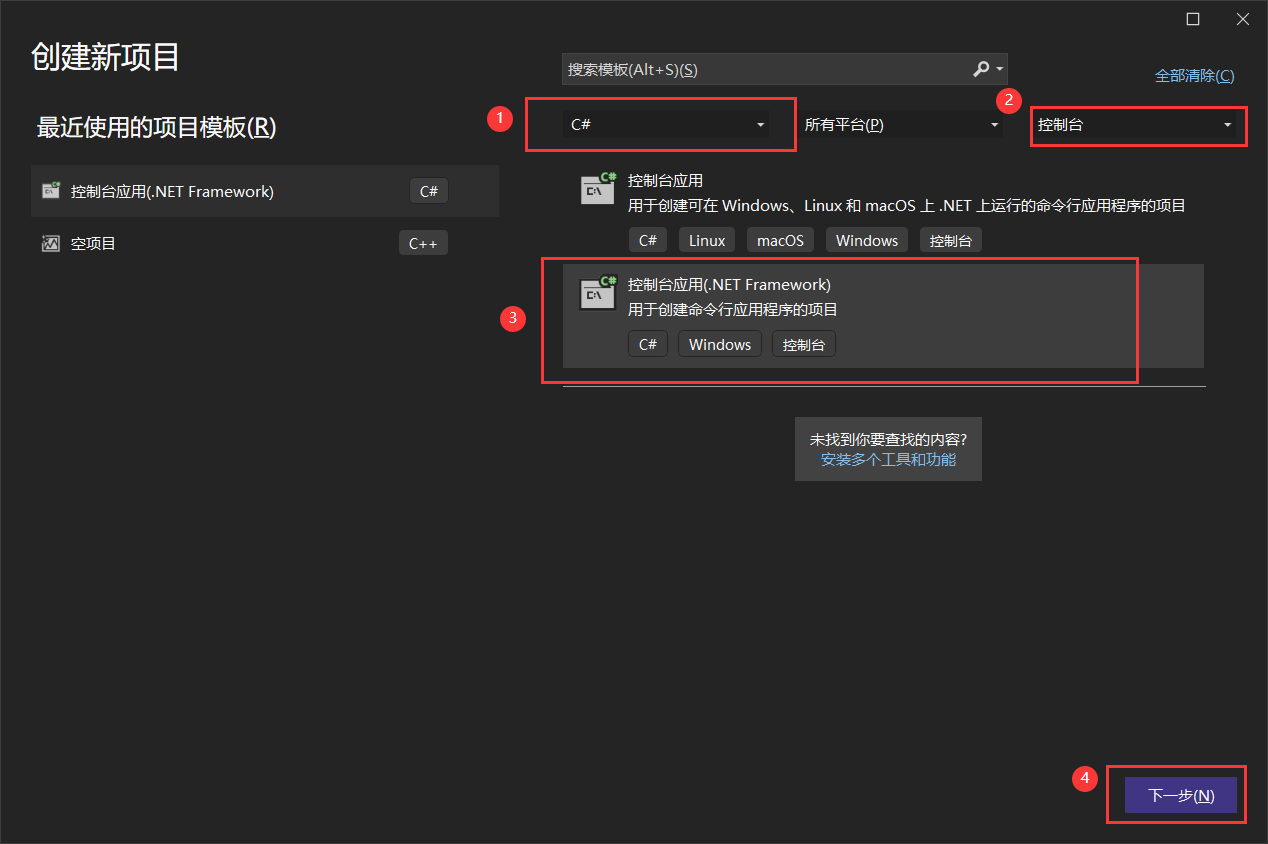
将复制好的文件，粘贴到 Python3.7 安装路径下的《DLLs》文件下。 此处 Python 3.7 的安装路径是 C:\Python\Python37。

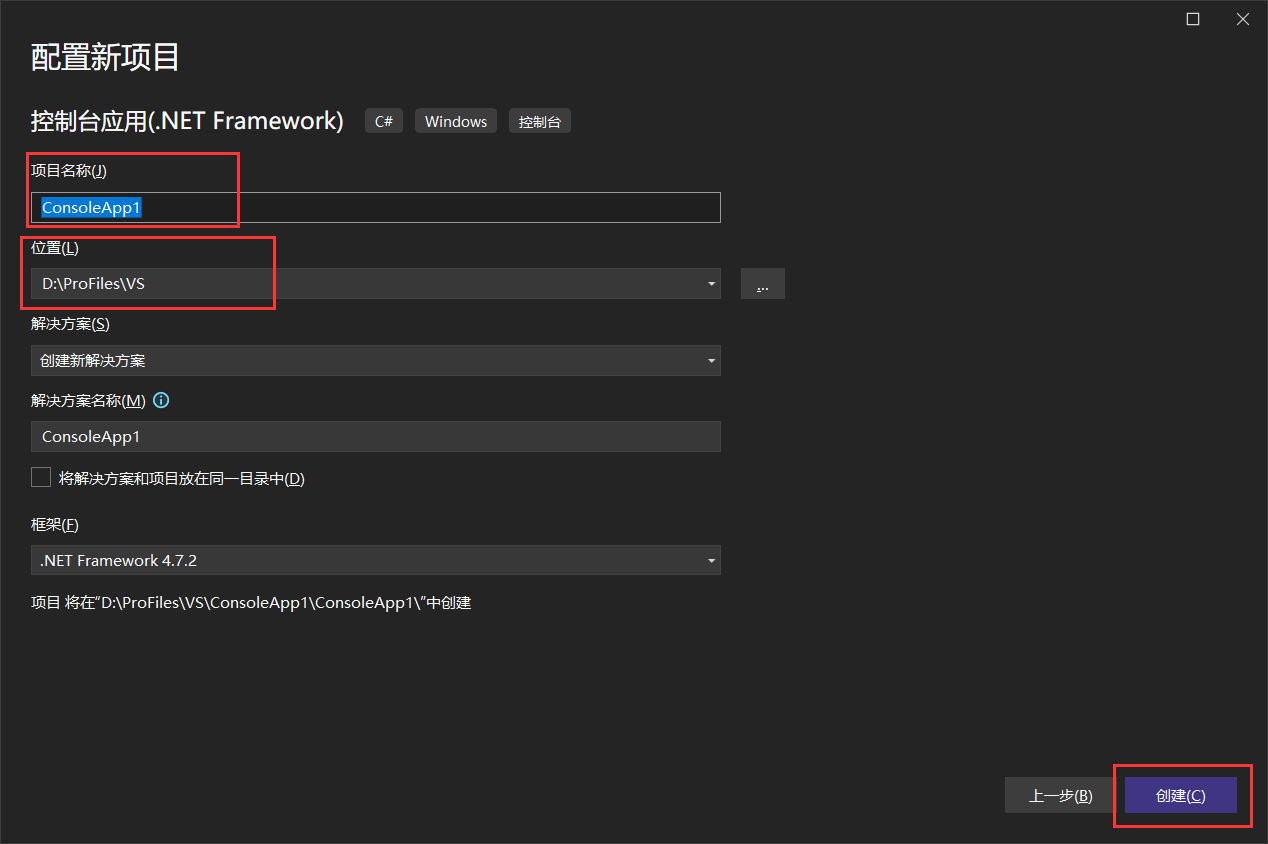


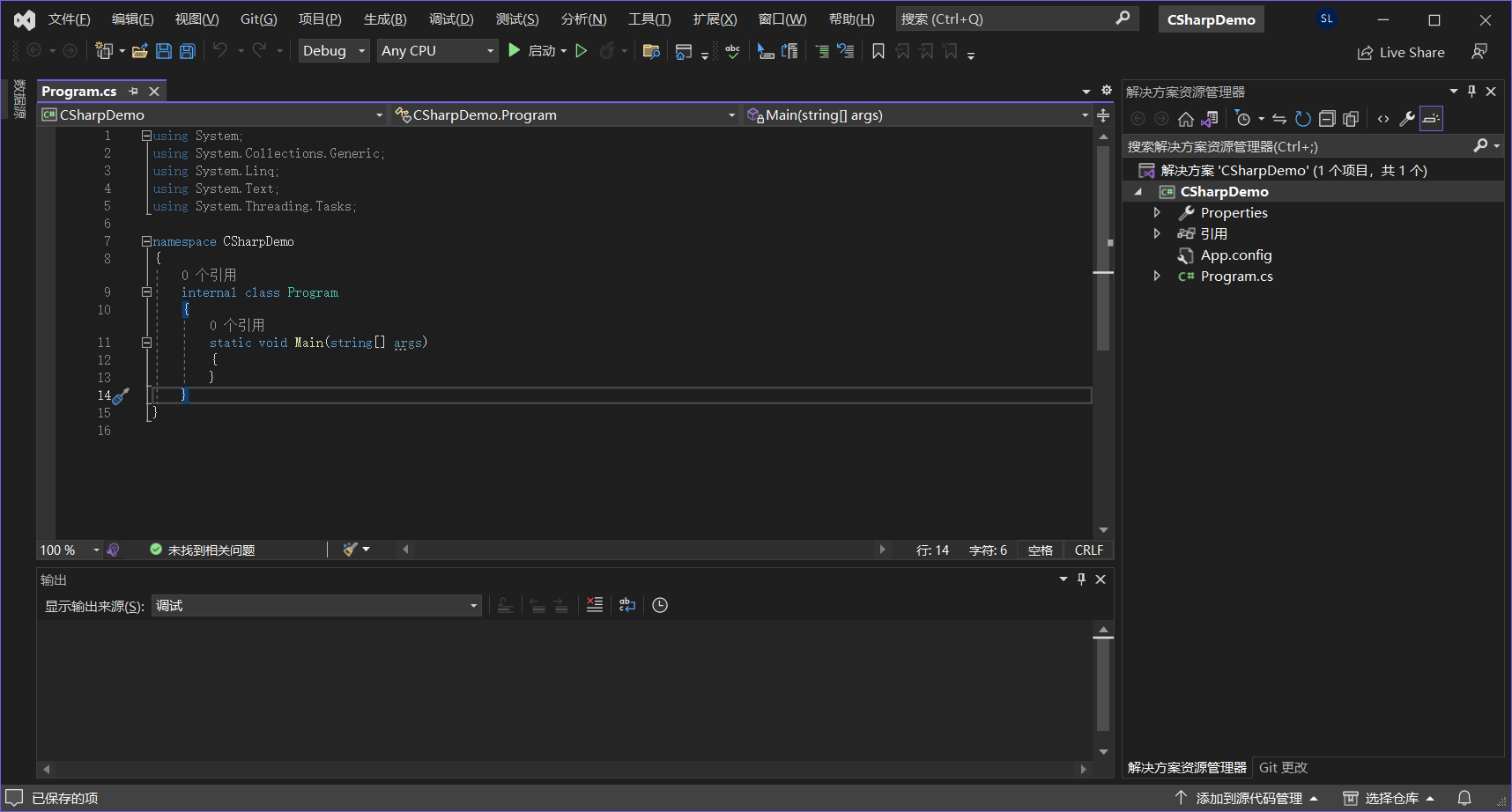
在pycharm中打开工程文件，查看调用的接口，运行环境

## 2.3 C#项目构建

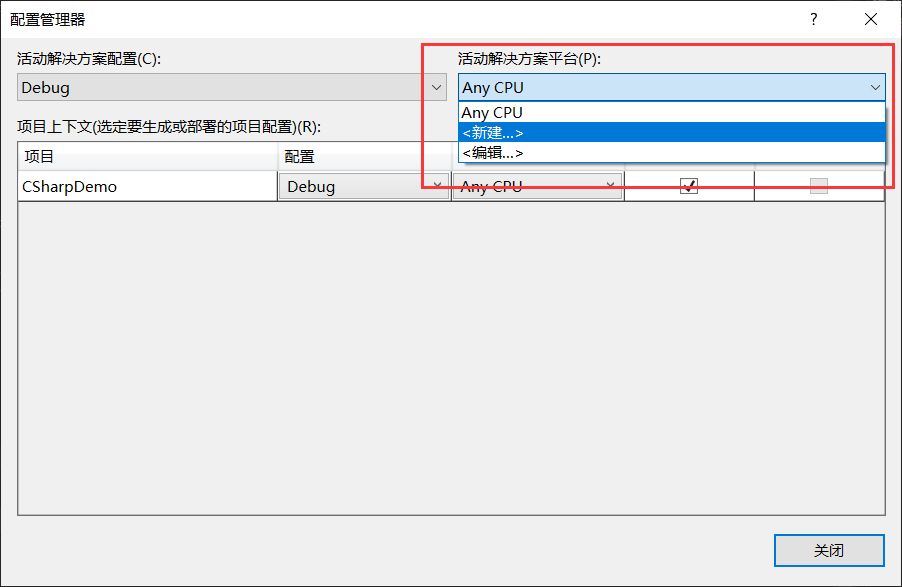
### 环境引入

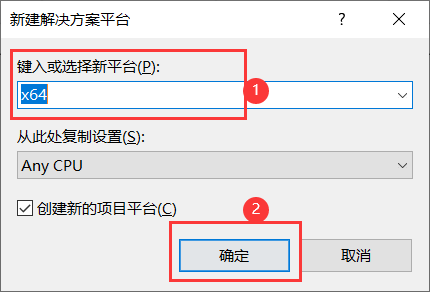




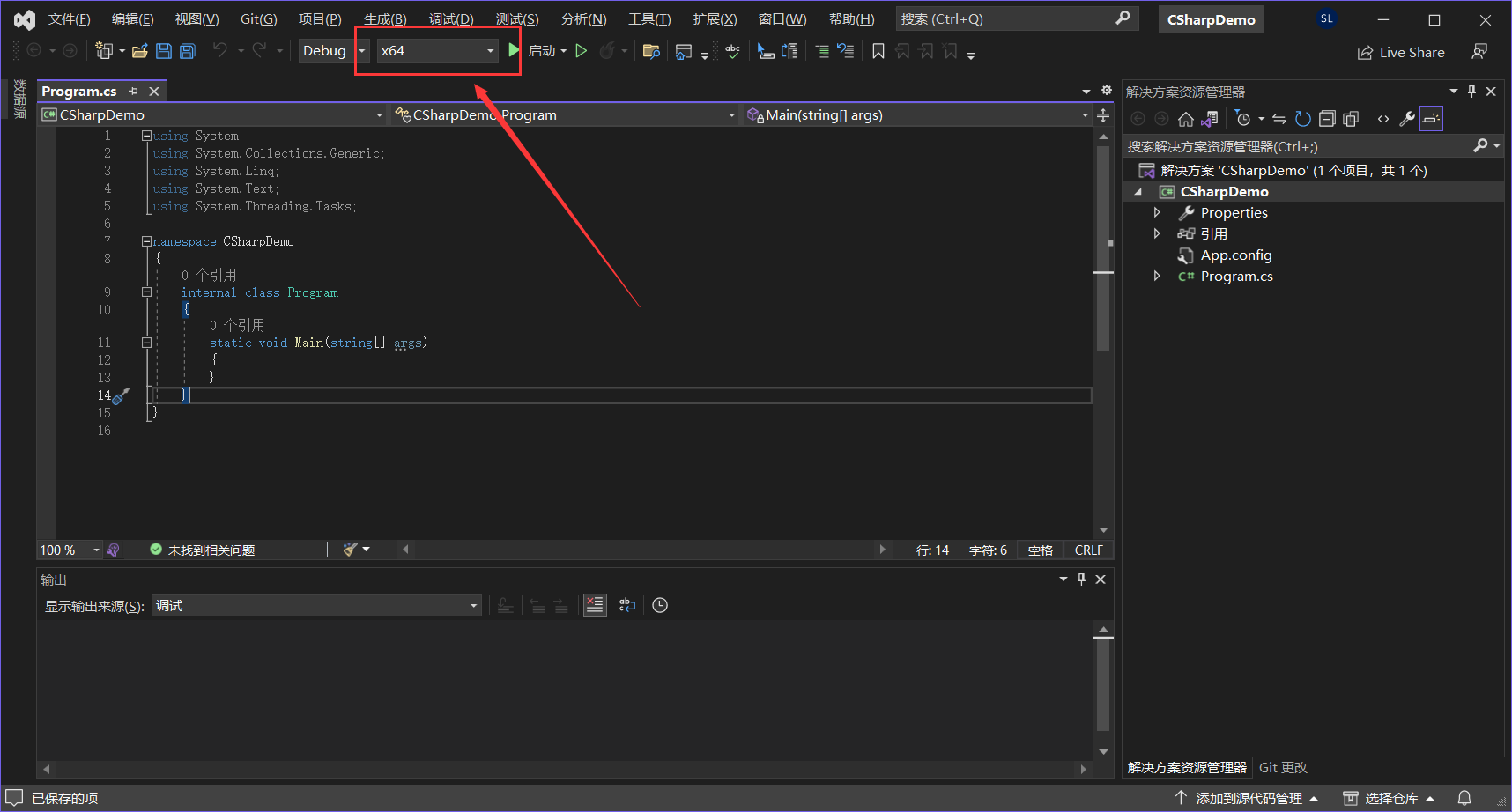




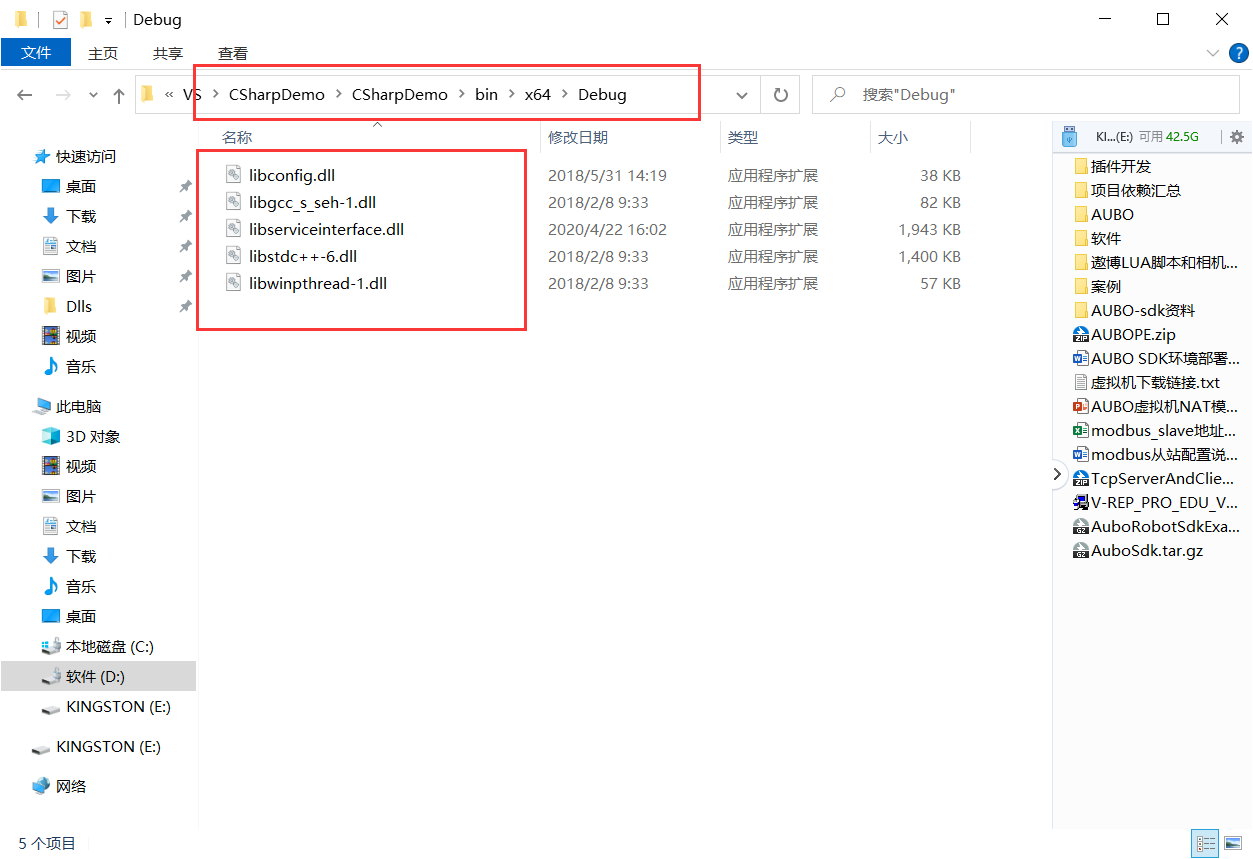




如下图所示，配置管理器由“Any CPU”改为“x64”。



将配置文件拷贝到 SdkTest/SdkTest/bin/x64/Debug 路径下。



### 示例代码

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Runtime.InteropServices;

namespace SdkTest

{

class Program

{

const string service\_interface\_dll = "libserviceinterface.dll";

const string ip = "192.168.71.128";

const int port = 8899;

const int RS\_SUCC = 0;

static UInt16 rshd = 0xffff;

//初始化机械臂控制库

[DllImport(service\_interface\_dll, EntryPoint = "rs\_initialize", CharSet = CharSet.Auto, CallingConvention = CallingConvention.Cdecl)]

public static extern int rs\_initialize();

//反初始化机械臂控制库

[DllImport(service\_interface\_dll, EntryPoint = "rs\_uninitialize", CharSet =CharSet.Auto, CallingConvention = CallingConvention.Cdecl)]

public static extern int rs\_uninitialize();

//创建机械臂控制上下文句柄

[DllImport(service\_interface\_dll, EntryPoint = "rs\_create\_context", CharSet = CharSet.Auto, CallingConvention = CallingConvention.Cdecl)]

public static extern int rs\_create\_context(ref UInt16 rshd);

//注销机械臂控制上下文句柄

[DllImport(service\_interface\_dll, EntryPoint = "rs\_destory\_context", CharSet = CharSet.Auto, CallingConvention = CallingConvention.Cdecl)]

public static extern int rs\_destory\_context(UInt16 rshd);

//链接机械臂服务器

[DllImport(service\_interface\_dll, EntryPoint = "rs\_login", CharSet = CharSet.Auto, CallingConvention = CallingConvention.Cdecl)]

public static extern int rs\_login(UInt16 rshd, [MarshalAs(UnmanagedType.LPStr)] string addr, int port);

static void Main(string[] args)

{

//初始化机械臂控制库

int result = rs\_initialize();

if (result == RS\_SUCC)

{

Console.Out.WriteLine("机械臂初始化成功！");

}

else

{

Console.Error.WriteLine("机械臂初始化失败！错误码：{0}", result);

}

//创建机械臂控制上下文句柄

result = rs\_create\_context(ref rshd);

if (result == RS\_SUCC)

{

Console.Out.WriteLine("创建上下文句柄成功！");

}

else

{

Console.Error.WriteLine("创建上下文句柄失败！错误码：{0}",

result);

}

//机械臂登录

result = rs\_login(rshd, ip, port);

if (result == RS\_SUCC)

{

Console.Out.WriteLine("登录成功！");

}

else

{

Console.Error.WriteLine("登录失败！错误码： {0}", result);

}

}

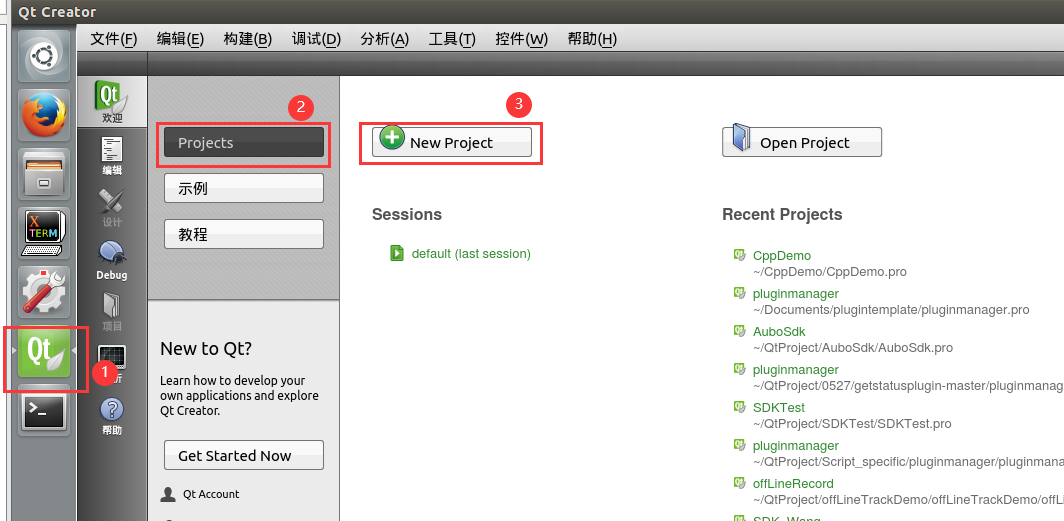
}

}

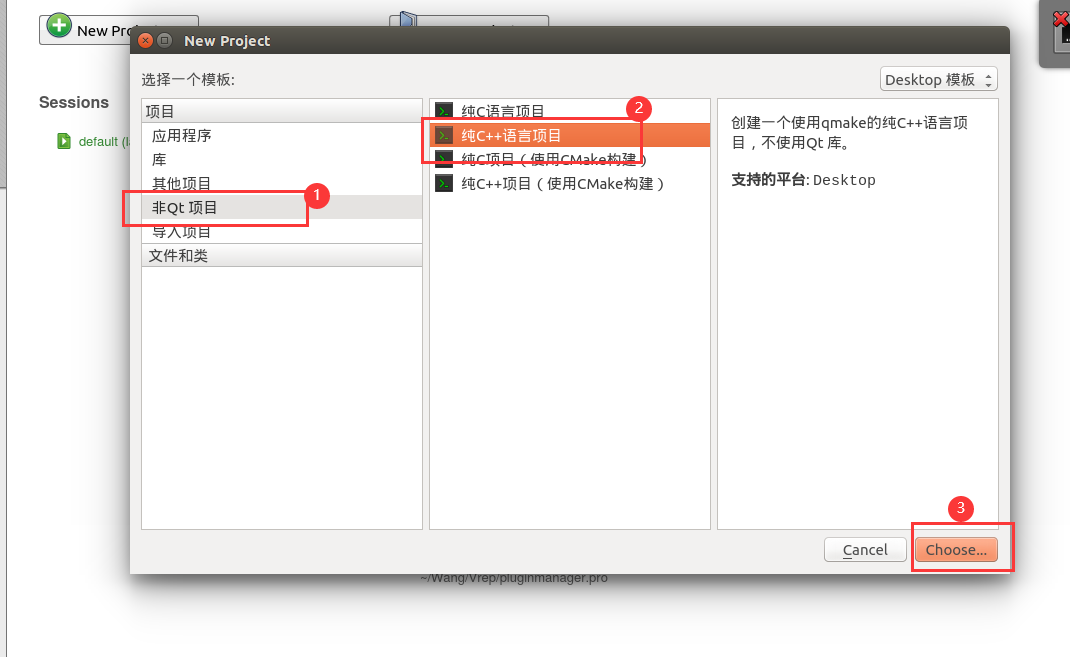
## 2.4 C++（Linux）项目构建

### 环境引入

打开虚拟机，运行Qt Creater，点击创建项目

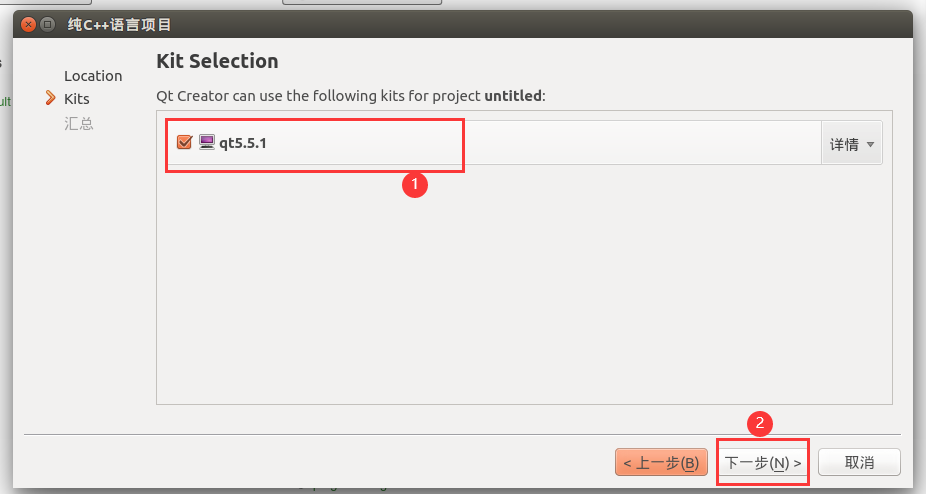


选择 非Qt项目 -> 纯C++语言项目



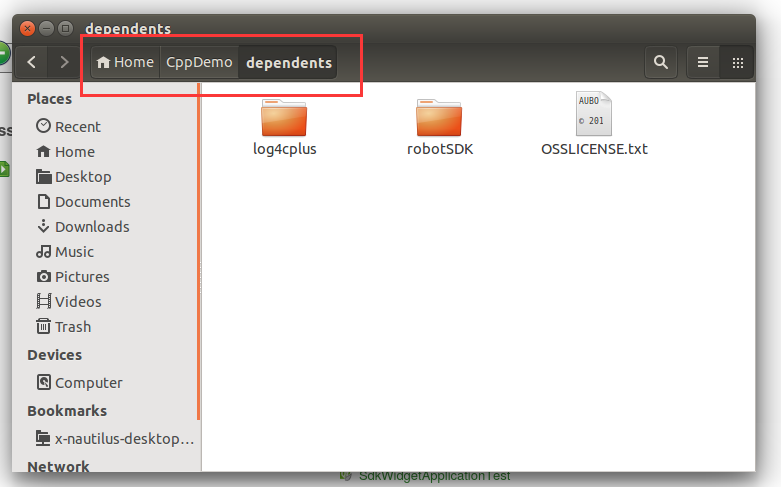
设置文件名，并设置存储位置







复制对应C++文件夹中的文件，在刚刚创建的文件目录中新建一个dependents文件夹，并粘贴文件到dependents文件夹下



### 示例代码

在pro文件中添加如下代码，以导入对应的依赖包

unix {

#32bit OS

contains(QT\_ARCH, i386) {

CONFIG += c++11

DEFINES += \_GLIBCXX\_USE\_CXX11\_ABI=0

INCLUDEPATH += $$PWD/dependents/robotSDK/inc

LIBS += -L$$PWD/dependents/robotSDK/lib/linux\_x32/ -laubo\_sdk

LIBS += -lpthread

}

#64bit OS

contains(QT\_ARCH, x86\_64) {

INCLUDEPATH += $$PWD/dependents/robotSDK/inc

LIBS += -L$$PWD/dependents/log4cplus/linux\_x64/lib/ -llog4cplus

LIBS += -L$$PWD/dependents/robotSDK/lib/linux\_x64/ -laubo\_sdk

LIBS += -lpthread

}

}

在主文件main.cpp中使用以下代码进行测试，运行出现login success!即为登陆成功

#include <iostream>

#include "AuboRobotMetaType.h"

#include "serviceinterface.h"

#define SERVER\_HOST "192.168.71.129"

#define SERVER\_PORT 8899

int main()

{

cout << "Hello World!" << endl;

ServiceInterface robotService;

int ret = robotService.robotServiceLogin(SERVER\_HOST,SERVER\_PORT,"aubo","123456");

if (ret == aubo\_robot\_namespace::InterfaceCallSuccCode){

std::cout<< "登陆成功"<<std::endl;

} else{

std::cout<<"登录失败"<<std::endl;

}

return 0;

}

## 2.5Python（Linux）环境构建

环境配置

将对应文件夹下的so文件复制

放到虚拟机/usr/lib/python3/dist-packages/目录下即可

模拟运行 -> 使用命令行python3命令运行对应的python文件即可