**《介绍》**

c++学习笔记

**《环境》**

**vscode**

使用vscode的优势在于vscode体量小、c++和python代码可以一起开发

vscode下载：https://code.visualstudio.com/

环境配置：下载mingw，在vscode插件中下载c++插件，搜教程配置编译环境。配置时注意使用g++替代gcc

MinGW：c++开发工具，包含GCC编译器，安装c++环境需要下载MinGW

MinGW下载地址：https://github.com/niXman/mingw-builds-binaries/releases

c\_cpp\_properties.json：配置好c++插件后会自动在.vscode下生成的环境配置文件

tasks.json：编译时的配置文件。自动生成：[终端菜单]->[配置文件]->[选择MinGW环境]。文件中的args参数的-g为debug编译(默认)，改为-02为release编译

编译：[终端菜单]->[运行生成任务]->[选择MinGW环境]。会生成.exe文件

运行：在下方终端中输入[.\文件名.exe]运行

vscode运行：点击[运行和调试]后会自动编译和运行，并且如果有断点会直接调试。快捷键[ctrl+f5]

新项目：创建新项目时，可以复制原来的.vscode配置文件

多文件编译：多文件编译(不同文件间函数调用)时要改造tasks.json

格式化：c++代码按ctrl+s保存时会自动格式化

**其他**

cmake：开源、跨平台的项目构建系统，支持多种操作系统和编译器。比如从github下载和安装库，可以通过cmake的方式构建来避免环境影响

**《c++基础》**

c++标准：c++语言规范，国际标准化组织制定，c++标准和编译器都在不断更新

c++库安装：由于c++的编译器没有统一的标准，导致很多库不通用，没有像python的conda一样的库管理工具，通常要手动下载和管理c++的库

编译器：c++语言需要编译器转换为对应系统的可执行文件，比如windows上编译后为.exe文件。常见的编译器有GCC、Clang、MSVS等。常用的编译器为GCC，支持多种系统和最新c++标准。通常会下载MinGW开发工具(包含GCC编译器)来配置c++环境

GCC：GCC编译器有gcc、g++两种，gcc针对c，g++针对c++

编译模式：编译器会将代码编译为可执行语言，有debug编译和release编译两种方式。前者包含调试信息，后者是经过大量优化的最终版本

字符串：c++中要使用双引号

静态库：.a或.lib。程序使用静态库时，会复制静态库的内容到系统上。优点是效率高，跨系统时不依赖环境；缺点是占用空间大，每次调试代码时要重新编译静态库

动态库：.so或.dll。动态库存在于程序外部，供程序调用

**数据类型**

不同编译器下的数据大小可能有区别，这里以常见的为例

字节：1个字节有8位二进制空间

short：短整型。2字节，-32768到32767

unsigned short：无符号短整型。2字节，0到65535

int 整型。4字节，-20多亿到20多亿

unsigned short：无符号整型。4字节，0到40多亿

long：长整型。4/8字节，64位linux为8，其他为4

unsigned long：无符号长整型。4/8字节，64位linux为8，其他为4

long long：长长整型。8字节

unsigned long long：无符号长长整型。8字节

float：单精度浮点数。4字节，7位有效数字

double：双精度浮点数。8字节，15位有效数字

long double：扩展精度浮点数。16字节，19位有效数字

Void：0字节

Bool：1字节

指针(\*)：4/8字节。32位系统为4，64位系统为8

枚举：变量默认4字节，可以定义数据类型

联合体(union)：所有成员共用一个内存

引用(&b=a)：引用成员的数值和内存地址都是同一个

**基本操作**

int a=1;：定义整型。也可以int a;//a=1;

&a：a的地址。所有的数据都有值和地址两个属性

int \*p=&a;：定义指针。注意定义后的指针是p不是\*p，指针p有自己的地址，值为其他数据的地址，\*p为a的值。即：&p=未知，p=&a，\*p=a。p+1指向a地址傍的下一个地址

int array[10];：定义整型数组。由10个连续int数据组成。数组名array=&array[0]

const int a=1;：定义常量，之后a的值不能改变

constexpr int a=1：编译时常量。在编译时会直接将a替换为1，性能比const高，但由于在编译时优化，不能包含函数调用和对象构造

**导入库**

#include<A.h>：导入头文件A.h。A.h中存放源文件A.cpp中的函数声明，相当于导入A库，系统将从系统目录开始搜寻A.h文件

#include”A.h”：系统先从当前目录搜寻A.h文件，再从系统目录搜寻

#include<iostream>：导入C++必要库iostream(不加.h)，包含标准名字空间std。对于一般的库，加或不加.h都可以

名字空间：即函数库

using namespace 库;：导入库中的函数。比如原本要使用std::cout，导入std中函数后可以直接使用函数cout

**条件语句**

if 条件 end(or) 条件://...//elif ...://...//else://...

for(i=0;i<a;i++)//...;：循环，i需提前定义

a++：自加1

a--：自减1

**位运算符**

优先级：算术运算符>关系运算符>位运算符>逻辑运算符

a<<n：二进制左位移n次。等于乘以2的n次方，运算效率更高

a>>n：二进制右位移n次。等于除以2的n次方，运算效率更高

&：二进制与

|：二进制或

^：二进制异或

~：二进制非

**逻辑运算符**

&&：与

||：或

!：异或

**函数**

类型 函数名(类型 参数=默认值){...}：函数定义

void 函数名(int x){...}：定义形式参数

static void ...：定义静态函数。静态函数不能被其他文件使用

extern int a：在导入其他c++文件后，使用其中定义的变量a(函数可以直接使用)

inline void ...：定义展开函数。在执行代码时会将函数中的内容作为一个单独的模块，如果展开后与调用代码合在一起编译会更简洁、提高运行速度，但有for循环等会导致代码体积膨胀，导致编译后文件变大，因此对于复杂的函数不使用。相比类似展开的宏定义，inline有类型检查更安全

函数重载：允许有相同名称的函数，但它们的参数类型要不同。比如void 函数(int a)和void 函数(float a)，当传入参数不同时会调用不同的函数

**《c++常用库》**

**<stdio.h>**

c语言的标准输入输出库，有printf、scanf,、puts,、gets等函数，c++也可以使用，但不支持面向对象编程、不提供类型安全。c++有自己的<iostream>库，提供更丰富的功能

**<iostream>**

#include <iostream> // using namespace std;

c++的标准输入输出库，早期版本使用<iostream.h>

**<string>**

处理字符串的库，不用关心内存分配和\0

string str1;：定义空字符串

getline(cin,str1);：输出带空格的字符串

str3=str1+“ “+str2;——字符串合并，中间可加空格

str1.insert(a,”A”);——在str1下标为a的字符前插入字符A

str1.replace(a,b,”A”);——从下标a开始往后共b个字符替换为A

str1.substr(a,b);——选取下标a开始往后共b个字符

str1.erase(a,b);——删除下标a开始往后共b个字符

str1.find(“A”);——查找str1中是否含有A，存在返回A第一个字符的下标，不存在返回-1

**<Eigen/Dense>**

#include <Eigen/Dense> // using namespace Eigen;

安装教程：<https://blog.csdn.net/qq_45948208/article/details/132434560>

**<opencv2>**

下载地址：https://opencv.org/releases/

**《ue5》**

**基础知识**

虚幻引擎5(unreal engine5)：

虚拟微多边形几何体(nanite)：可以让美术师创建出精致的几何体细节，无需考虑多边形数量、内存、绘制次数预算，不用将细节烘焙到法线贴图中也可以使用

全动态全局光照解决方案(lumen)：

安装：先下载epic games平台，在epic games平台中可以创建和下载不同版本的虚幻5引擎，初次安装时不需要选目标平台(文件很大)。创建后需要搜教程更改缓存路径

创建项目：创建项目时可以选蓝图或者c++，c++需要安装编程软件和环境

**快捷键**

F：视角跳转到当前选择的物体