CG语言基础：

(摘抄自《GPU编程与CG语言GPU-Programming-AndCgLanguage-Primer》)

概述：

Cg（C for Graphcis）语言，是NVIDIA 与Microsoft 合作研发，旨在为开发

人员提供一套方便、跨平台（良好的兼容性），控制可编程图形硬件的高级语言。

Cg 是一个可以被OpenGL 和Direct3D 广泛支持的图形处理器编程语言。

数据类型：

Cg 支持7 种基本的数据类型：

1. float，32 位浮点数据，一个符号位。浮点数据类型被所有的profile 支持

（但是DirectX8 pixel profiles 在一些操作中降低了浮点数的精度和范围）；

2. half，16 为浮点数据；

3. int，32 位整形数据，有些profile 会将int 类型作为float 类型使用；

4. fixed，12 位定点数，被所有的fragment profiles 所支持；

5. bool,布尔数据，通常用于if 和条件操作符（?:），布尔数据类型被所有的

profiles 支持；

6. sampler\*,纹理对象的句柄（the handle to a texture object）

7. string，字符类型，该类型不被当前存在的profile 所支持，实际上也没有

必要在Cg 程序中用到字符类型。

向量：

Cg 程序中可以声明float1、float2、float3、

float4 类型的数组变量，但是不能声明超过4 元的向量。

向量还可以通过较短的向量进行构建：

float2 a = float2(1.0, 1.0);

float4 b = float4(a, 0.0, 0.0);

Cg 中向量、矩阵与数组是完全不同，向量和矩阵是内置的数据类型

（矩阵基于向量），而数组则是一种数据结构，不是内置数据类型！

矩阵：

Cg 还提供矩阵数据类型，

不过最大的维数不能超过4\*4 阶。

例如：

float1x1 matrix1;//等价于float matirx1; x 是字符，并不是乘号！

float2x3 matrix2;// 表示2\*3 阶矩阵，包含6 个float 类型数据

float4x2 matrix3;// 表示4\*2 阶矩阵，包含8 个float 类型数据

float4x4 matrix4;//表示4\*4 阶矩阵，这是最大的维数。

矩阵的初始化方式为：

float2x3 matrix5 = {1.0, 2.0, 3.0, 4.0, 5.0, 6.0};

数组：

和C里基本一模一样。

要获取数组长度，可以调用“.length”，例如：

float a[10]; //声明一个数组

int length = a.length;//获取数组长度

结构体：

结构体：

目前的Cg 语言中的结构体以展现“封装”功能为主，并不支持继承机制。

struct myAdd

{

float val;

float add(float x)

{

return val + x;

}

};

myAdd s;

使用符号“.”引用结构体中的成员变量和成员函数。例如：

float a = s.value;

float b = s.add(a);

注意：在当前的所有的profile 版本下，如果结构体的一个成员函数使用了

成员变量，则该成员变量要声明在前。此外，成员函数是否可以重载依赖于使用

的profile 版本。

struct VertexIn

{

float4 position : POSITION;

float2 textureCoordinates : TEXCOORD0;

};

输入结构体，带有绑定语义：

( Binding Semantics)，所谓

绑定语义类型是为了与宿主环境进行数据交换的时候识别不同数据类型的。目前

Cg 支持的绑定语义类型包括POSTION 位置)，COLOR(颜色)，NORMAL(法向

量)，Texcoord(纹理坐标)等类型。

表达式与控制语句：

与C语言相同之处：

<,<=,!=,==,>=,>,&&,||,!，：\*乘法；

/除法；-取反；+加法；—减法；%求余；++；——；\*=；/=；+=；-=

注意：

1.逻辑操作符也可以对向量使用，返回的变量类型是同样长度的内置bool向量。）

2.：Cg中的逻辑与（&&）和逻辑或（||）不存在C中的短路现

象（short-circuiting，即只用计算一个操作数的bool值即可），而是参与运算的操

作数据都进行bool分析。

3.Cg语言对向量的数学操作提供内置支持，即所有数学操作都可以直接用在向量上。

4.求余操作符%。只能在int类型数据间进行

5.Cg语言中的移位操作符，功能和C语言中的一样，也可以作用在向量上，但

是向量类型必须是int类型

----

不同的是：

1.Cg允许在向量类型变量上使用操作符，例如>操作符可以用来比较两个向量各个分量的大小关系

例如：

float3 a = float4(0.5, 0.0, 1.0);

float3 b = float4(0.6, -0.1, 0.9);

bool3 c = a<b;

运算后向量c的结果为float3(true, false, true);

2.可以使用Cg语言中的swizzle操作符（.）将一个向量的成员取出组成一个新

的向量。swizzle操作符被GPU硬件高效支持。swizzle操作符后接x、y、z、w，

分别表示原始向量的第一个、第二个、第三个、第四个元素；swizzle操作符后接

r、g、b和a的含义与前者等同。不过为了程序的易读性，建议对于表示颜色值的

向量，使用swizzle操作符后接r、g、b和a的方式。

例如：

float4(a, b, c, d).xyz 等价于 float3(a, b, c)

float4(a, b, c, d).xyy 等价于 float3(a, b, b)

float4(a, b, c, d).wzyx 等价于 float4(d, c, b, a)

float4(a, b, c, d).w 等价于 float d

注意 ：swizzle操作符只能对结构体和向量使用，不能对数组使用。

条件操作符 ：

expr1 ? expr2 : expr3;

Cg中的条件操作符一个独特的性能是：支持向量运算。即，expr1的计算结果可以是bool型向量，expr2和expr3必须是与expr1长度相同的向量

控制流语句：

Cg中的控制流语句和循环语句与C语言类似：条件语句有：if、if-else；循环语句有：while、for。break语句可以和在for语句中使用。

Cg语言中的控制流语句要求其中的条件表达式返回值都是bool类型，这一点是与C语言不同之处

因此，如果没有确切的把握，不要在低级的profiles中使用循环控制语句。

函数的递归调用（recursion）在Cg语言中是被禁止的。

Switch 、case和default在Cg中作为保留关键字存在，但是它们目前不被任何profile所支持。

绑定语义词（binding semantics）：

一系列独特的关键字，这些关键字不但用于指定输入图元的数据含义（是位置信息，还是法向量信息），本质也则对应着这些图元数据存放的硬件资源（寄存器或者纹理）。

除语义词外，Cg 中还提供了三个关键字，in、out、inout，用于表示函数的输入参数的传递方式，称为输入\输出关键字，这组关键字可以和语义词合用表达硬件上不同的存储位置，即同一个语义词，使用in 关键字修辞和out 关键词修辞，表示的图形硬件上不同的寄存器。

Cg 语言还提供两个修辞符：uniform，用于指定变量的数据初始化方式；const

关键字的含义与C\C++中相同，表示被修辞变量为常量变量。

图形硬件处理过程中，数据通常暂存在寄存器中，故而在Cg 语言中，通过引入语

义绑定（binding semantics）机制，指定数据存放的位置

记住这一点：语义，是两个处理阶段（顶点程序、片段程序）之间的输入\输出数据和寄存器之间的桥梁，同时语义通常也表示数据的含义，如POSITION一般表示参数种存放的数据是顶点位置。语义，只对两个处理阶段的输入\输出数据有意义，也就是说，语义只有在入口函数中才有效，在内部函数（一个阶段的内部处理函数，和下一个阶段没有数据传递关系）的无效，被忽略。

语义词POSITION0 等价于POSITION，其他的语义词也有类似的等价关系。至于为什么要将三元向量便为四元向量，又称齐次坐标，具体请看附录A。

下面这些语义词适用于所有的Cg vertex profiles作为输出语义和Cg fragmentprofiles的输入语义：POSITION，PSIZE，FOG,COLOR0-COLOR1,TEXCOORD0-TEXCOORD7。

顶点着色程序必须声明一个输出变量，并绑定POSITION语义词，该变量中的数据将被用于，且只被用于光栅化！

你也可以选择不使用struct结构，而直接在函数形参中进行语义绑定。无论使用何种方式，都要记住vertex program中的绑定语义（POSITION除外）的输出形参中的数据会传递到fragment program中绑定相同语义的输入形参中。

顶点入口函数的声明后带有“COLOR”语义词，表示该函数需要反馈一个颜色值，所以函数的返回类型为float4，函数体也必须以return语句结束。

参数传递方式：

1. in: 修辞一个形参只是用于输入，进入函数体时被初始化，且该形参值的改变不会影响实参值，这是典型的值传递方式。

2. out: 修辞一个形参只是用于输出的，进入函数体时并没有被初始化，这种类型的形参一般是一个函数的运行结果；

3. inout: 修辞一个形参既用于输入也用于输出，这是典型的引用传递。举例

举例如下：

void myFunction(out float x); //形参x，只是用于输出

void myFunction(inout float x); //形参x，即用于输入时初始化，也用于输出数据

void myFunction(in float x); //形参x，只是用于输入

void myFunction(float x); /等价与 in float x，这种用法和C\C++完全一致

函数：

Cg 语言中的函数声明形式与C\C++中相同。

注意：如果函数没有返回值，函数的返回类型一定要是void，否则编译会出

现大量的错误

数组形参：

Cg 语言中不存在指针机制（图形硬件不支持），数组作为函数形参，传递

的是数组的完整拷贝。数组类型变量作为函数形参，可以是一维的也可以是多维的，并且不必声明数组长度。

函数重载：

Cg 语言支持函数重载（Functon Overlaoding），其方式和C++基本一致，通过形参列表的个数和类型来进行函数区分。