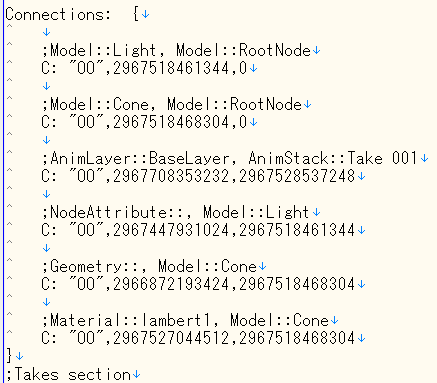
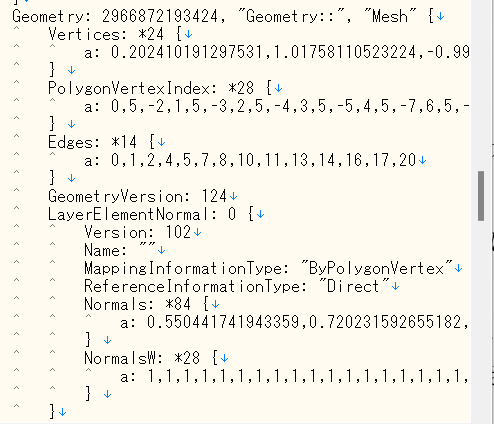
0６ – FBX②

**３Dフォーマット**

　FBXの内部データの最小単位はノードで表される。ゲームエンジンのように、一つのオブジェクトが一つのノードを指しているのではなく、一つのデータが一つのノードを表す。一つのモデルには、頂点座標、結線情報、法線情報、マテリアルなどのデータがノードで紐づいており、それらを組み合わせて一つのモデルを表示している。ノードはルートノードから始まり、カメラやライトなども含みつつ、モデルの細かい情報まで網羅する。

**座標情報**

　テキストフォーマットで読み込みを行っているため、一つずつ読み込んでいけば問題ない。頂点のデータは[Vertices]にまとめられている。

[\*24]は頂点の数ではなく、データの数のため、頂点一つでx,y,zを表し、頂点8つなら24個のfloatのデータとなる。

-- 文字探索関数 --

|  |
| --- |
| char\* FindKeyword(char\* ptr, const char\* keyword) {  ptr = strstr(ptr, keyword);  return ptr;  } |

-- 頂点データの読み込み --

|  |
| --- |
| ptr = FindKeyword(ptr, "Vertices:");  if (ptr != NULL) {  ptr = FindKeyword(ptr, "\*");  if (ptr != NULL) {  int num = atoi(ptr + 1);  m\_vtx.resize(num / 3);  ptr = FindKeyword(ptr, "a:");  if (ptr != NULL) {  ptr += 2;  for (int i = 0; i < m\_vtx.size(); ++i) {  m\_vtx[i].x = atof(ptr);  ptr = FindKeyword(ptr, ",") + 1;  m\_vtx[i].y = atof(ptr);  ptr = FindKeyword(ptr, ",") + 1;  m\_vtx[i].z = atof(ptr);  ptr = FindKeyword(ptr, ",") + 1;  }  }  }  } |

**インデックス情報**

　DirectXのポリゴン表示は三角形が基本の表示となる。インデックスを利用した表示では、複雑なモデルを表現しようとすれば必須となる。

　FBXのポリゴン表示は必ずしも三角形というわけではなく、多角形のポリゴンもサポートしている。そのため、インデックスデータは３つだったり、７つだったりと面ごとに必要な数のインデックスが異なる。各面の区切りを表すために、面を構成する最後のインデックスは「　　１の補数　　」で保存される。

例えば、[2,1,0]というインデックスならば[2,1,　　]というように保存される。C言語では、この計算を「　　~(NOT)　　」演算子で行う。

-- インデックスデータの読み込み --

|  |
| --- |
| ptr = FindKeyword(pGeometry, "PolygonVertexIndex:");  if (ptr != NULL) {  ptr = FindKeyword(ptr, "\*");  if (ptr != NULL) {  int num = atoi(ptr + 1);  ptr = FindKeyword(ptr, "a:");  if (ptr != NULL) {  ptr += 2;  int idxCnt = 0;  for (int i = 0; i < num; ++i) {  int idx = atoi(ptr);  ++idxCnt;  if (idxCnt >= 4) {  m\_idx.push\_back(m\_idx[m\_idx.size() - 3]);  m\_idx.push\_back(m\_idx[m\_idx.size() - 2]);  }  if (idx < 0) {  idx = ~idx;  idxCnt = 0;  }  m\_idx.push\_back(idx);  ptr = FindKeyword(ptr, ",") + 1;  }  }  }  } |

-- クラスで定義した各種取得関数 --

|  |
| --- |
| Vertex\* GetVertex() { return m\_vtx.data(); }  int GetVertexNum() { return m\_vtx.size(); }  unsigned long\* GetIndex() { return m\_idx.data(); }  int GetIndexNum() { return m\_idx.size(); } |

-- モデルの表示 --

|  |
| --- |
| if (g\_pFBX) {  glVertexPointer(3, GL\_FLOAT, 0, g\_pFBX->GetVertex());  glDrawElements(GL\_TRIANGLES, g\_pFBX->GetIndexNum(), GL\_UNSIGNED\_INT, g\_pFBX->GetIndex());  } |