GD1M01 – Project - Geometry for Games Header File

File name: geometry.h

```
enum EIntersections
{
    INTERSECTION_NONE,
    INTERSECTION_ONE,
    INTERSECTION_TWO
};

struct TVector2
{
    float m_fX;
    float m_fY;
};

struct TTriangle2
{
    TVector2 m_v2p1;
    TVector2 m_v2p2;
    TVector2 m_v2p3;
};

struct TRectangle
{
    TVector2 m_v2p1;
    TVector2 m_v2p2;
};

struct TVector3
{
    float m_fX;
    float m_fY;
    float m_fZ;
};

struct TTriangle3
{
    TVector3 m_v3p1;
    TVector3 m_v3p2;
    TVector3 m_v3p3;
};

struct T3DLine
{
    TVector3 m_v3q; //point on the line
    TVector3 m_v3v; //direction vector along the line
};

struct TCircle
{
    TVector2 m_v2center;
    float m_fRadius;
};
```

```
struct TPlane
{
    TVector3 m_v3normal;
    TVector3 m_v3point;
};

struct TSphere
{
    TVector3 m_v3center;
    float m_fRadius;
};

bool Equals(const TVector3& _krA, const TVector3& _krB);

TVector3& Add(const TVector3& _krA,
              const TVector3& _krB,
              TVector3& _rResultant);

TVector3& Subtract(const TVector3& _krA,
                   const TVector3& _krB,
                   TVector3& _rResultant);

TVector3& ScaleVector(const TVector3& _krA,
                      const float _kfScalar,
                      TVector3& _rResultant);

float Magnitude(const TVector3& _krA);

float DotProduct(const TVector3& _krA, const TVector3& _krB);

TVector3& CrossProduct(const TVector3& _krA,
                       const TVector3& _krB,
                       TVector3& _rResultant);

TVector3& Normalise(const TVector3& _krA, TVector3& _rResultant);

TVector3& Projection(const TVector3& _krA,
                     const TVector3& _krB,
                     TVector3& _rResultant);

float ComputeAngleBetween(const TVector2& _krA,
                          const TVector2& _krB);

float ComputeAngleBetween(const TVector3& _krA,
                          const TVector3& _krB);

float ComputeDistancePointToLine(const T3DLine& _krLine,
                                 const TVector3& _krPoint);

float ComputeDistancePointToPlane(const TPlane& _krPlane,
                                  const TVector3& _krPoint);

//Distance between point and center of the spheres
float ComputeDistancePointToSphere(const TSphere& _krSphere,
                                   const TVector3& _krPoint);
```

```
//Distance between center of the circles
float ComputeDistanceCircleToCircle(const TCircle& _krCircle1,
                                    const TCircle& _krCircle2);

//Distance between center of the circle and triangle
float ComputeDistanceCircleToTriangle(const TCircle& _krCircle,
                                      const TTriangle2& _krTriangle);


EIntersections ComputeLineSphereIntersection(const T3DLine& _krLine,
                                             const TSphere& _krSphere,
                                    TVector3& _rv3IntersectionPoint1,
                                    TVector3& _rv3IntersectionPoint2);

bool IsLinePlaneIntersection(const T3DLine& _krLine,
                             const TPlane& _krPlane,
                             TVector3& _rv3IntersectionPoint);


bool IsIntersection(const T3DLine& _krLine1,
                    const T3DLine& _krLine2);

TVector3& ComputeIntersectionBetweenLines(const T3DLine& _krLine1,
                                          const T3DLine& _krLine2,
                                          TVector3& _rIntersectionPoint);

bool IsInFieldOfView(const TVector2& _krCameraPosition,
                     const TVector2& _krCameraDirection,
                     const float _kfFieldOfViewInRadians,
                     const TVector2& _krObjectPosition);

TVector3& FindTriangleNormal(const TTriangle3& _krTriangle,
                             TVector3& _rNormal);

bool IsSurfaceLit(const TVector3& _krPointOnSurface,
                  const TVector3& _krLightSourcePosition,
                  const TTriangle3& _krSurface);


TTriangle2& RotateTriangleAroundPoint(const TTriangle2& _krTriangle,
                                      const float _kfRotAngleInRadians,
                                      const TVector2& _krRotAroundPoint,
                                      TTriangle2& _rRotatedTriangle);
```