**Vec3**

This module is a templated structure pertaining to three objects of the specified type along with specific mathematical operations.

**Dependencies**

* Access to the following:
  + None
* Accessed by the following:
  + Vec4
  + Matrix3
  + Object Manager
  + Animation Manager
  + Collision Library
  + AI Manager
  + Camera
  + World Manager

**Public Interface**

|  |  |  |  |
| --- | --- | --- | --- |
| **Return** | **Name** | **Parameters** | **Description** |
|  | Vec3 | Void | This is the default constructor for the struct. |
|  | ~Vec3 | Void | This is the destructor that will handle cleaning up the struct. |
|  | Vec3 | Type vx, Type vy, Type vz | This is an overloaded constructor that takes in 3 items of the specified type. |
|  | Vec3 | Const vec3<VT> &v | This is the copy constructor that takes in a reference to another vec3 |
| Vec3& | Make\_zero | Void | Zeroes out this vector. |
| Vec3& | Normalize | Void | Normalizes this vector. |
| Vec3& | Negate | Void | Negates this vector. |
| Type | Magnitude | Void | Gets the magnitude of this vector |
| Type | Dot\_product | Const vec3& vectorA, const vec3& vectorB | Returns the dot product between two vectors |
| Inline friend void | Cross\_product | Vec3& vectorO, const vec3& vectorA, const vec3& vectorB | Stores the result, of two vectors crossed, in the out vector. |
| Inline friend vec3<T> | Interpolate | Const vec3& vectorA, const vec3& vector, float lambda | Linear Interpolation between two vectors. |
| Type\* | array | Void | Returns the array which is used to access the other private members. |
| Const Type\* | Array | Void | Returns the array which is used to access the other private members, but without allowing for anything to be changed. |

**Time to complete estimate:**

* complete

**Module Author(s):**

* Engine Development 2 Lab Assistants (Joe and Clark)

**Vec4**

This module is a templated structure pertaining to four objects of the specified type along with specific mathematical operations.

**Dependencies**

* Access to the following:
  + Vec3
* Accessed by the following:
  + Object Manager
  + Animation Manager
  + Collision Library
  + AI Manager
  + Camera
  + Renderer
  + World Manager

**Public Interface**

|  |  |  |  |
| --- | --- | --- | --- |
| **Return** | **Name** | **Parameters** | **Description** |
|  | Vec4 | Void | This is the default constructor for the struct. |
|  | ~Vec4 | Void | This is the destructor that will handle cleaning up the struct. |
|  | Vec4 | Type vx, Type vy, Type vz, Type vw | This is an overloaded constructor that takes in 4 items of the specified type. |
|  | Vec4 | Const vec3<VT> &v, type vw | This is an overloaded constructor that takes in a vec3 and the w component |
|  | Vec4 | Const vec4<VT> &v | This is the copy constructor that takes in a reference to another vec3 |
| Vec4& | Make\_zero | Void | Zeroes out this vector. |
| Vec4& | Normalize | Void | Normalizes this vector. |
| Vec4& | Negate | Void | Negates this vector. |
| Type | Magnitude | Void | Gets the magnitude of this vector |
| Type | Dot\_product | Const vec4& vectorA, const vec4& vectorB | Returns the dot product between two vectors |
| Inline friend vec4<T> | Interpolate | Const vec4& vectorA, const vec4& vector, float lambda | Linear Interpolation between two vectors. |
| Type\* | array | Void | Returns the array which is used to access the other private members. |
| Const Type\* | Array | Void | Returns the array which is used to access the other private members, but without allowing for anything to be changed. |

**Time to complete estimate:**

* complete

**Module Author(s):**

* Engine Development 2 Lab Assistants (Joe and Clark)

**Matrix3**

This module is a templated structure with nine objects of the specified type along with its specified mathematical operations.

**Dependencies**

* Access to the following:
  + Vec3
* Accessed by the following:
  + Matrix4
  + Animation Manager
  + Renderer
  + Object Manager
  + Camera
  + World Manager

|  |  |  |  |
| --- | --- | --- | --- |
| **Return** | **Name** | **Parameters** | **Description** |
|  | Matrix3 | Void | This is the constructor for the struct. |
|  | ~Matrix3 | Void | This is the destructor that will clean up the struct. |
|  | Matrix3 | Const vec3<VT> &v0, const vec3<VT> &v1, vec3<VT> &v2 | This is an overloaded constructor that takes in three vec3s. |
|  | Matrix3 | Type m00, type m01, type m02, type m10, type m11, type m12, type m20, type m21, type m22 | This is an overloaded constructor that takes in 9 objects of the specified type. |
| explicit | Matrix3 | Const matrix3<MT>& m | This is the copy constructor that takes in a const reference of a matrix3 |
|  | Matrix3 | Const vec3<T> &diag, const vec3<T> &sym | This is an overloaded copy constructor that makes a symmetrical matrix |
| Matrix3& | Operator= | Const matrix3<MT> &m | This is the assignment operator, that allows you to assign this matrix to the one passed in. |
| Inline T | Operator[] | Size\_t index | This is the overloaded [] operator that allows access to the components array |
| Inline const T | Operator[] | Size\_t index | This is the overloaded [] that allows access to the components array but without allowing for change |
| Matrix3<T> | Operator\* | Const matrix3<T> &rhs | This is the overloaded \* operator which performs a matrix-matrix multiply and returns the result |
| Matrix3<T> | Operator\*= | Const matrix3<T> &rhs | This is the overloaded \*= operator which performs a matrix-matrix multiply and assigns the result to this matrix |
| Matrix3& | Make\_zero | Void | Zeros out this matrix |
| Matrix3& | Make\_identity | Void | Make this an identity matrix |
| Matrix3& | Make\_rotation\_x | Type radians | Makes this matrix a rotation matrix on the x axis |
| Matrix3& | Make\_rotation\_y | Type radians | Makes this matrix a rotation matrix on the y axis |
| Matrix3& | Make\_rotation\_z | Type radians | Makes this matrix a rotation matrix on the z axis |
| Matrix3& | Rotate\_pre\_x | Type radians | Makes this matrix a rotation matrix on the x axis times itself. |
| Matrix3& | Rotate\_pre\_y | Type radians | Makes this matrix a rotation matrix on the y axis times itself. |
| Matrix3& | Rotate\_pre\_z | Type radians | Makes this matrix a rotation matrix on the z axis times itself. |
| Matrix3& | Rotate\_post\_x | Type radians | Makes a rotation matrix with this times a rotation matrix on the x axis |
| Matrix3& | Rotate\_post\_y | Type radians | Makes a rotation matrix with this times a rotation matrix on the y axis |
| Matrix3& | Rotate\_post\_z | Type radians | Makes a rotation matrix with this times a rotation matrix on the z axis |
| Matrix3& | Invert | Void | Inverts this matrix |
| Matrix3& | Transpose | Void | Transposes this matrix |
| Matrix3& | Negate | Void | Negates this matrix |
| Matrix3& | Skew | Const vec3<T> &v | Skews this matrix by the vector passed in |

**Time to complete estimate:**

* Completed

**Module Author(s):**

* Engine Development 2 Lab Assistants (Joe and Clark)

**Matrix4**

This module is a templated structure with sixteen objects of the specified type along with its specified mathematical operations.

**Dependencies**

* Access to the following:
  + Matrix3
* Accessed by the following:
  + Animation Manager
  + Renderer
  + Object Manager
  + Camera
  + World Manager

|  |  |  |  |
| --- | --- | --- | --- |
| **Return** | **Name** | **Parameters** | **Description** |
|  | Matrix4 | Void | This is the constructor for the struct. |
| explicit | Matrix3 | Type &n | This is an overloaded constructor that takes in a value of the specified type and initializes this matrix |
| Inline T | Operator[] | Size\_t index | This is the overloaded [] operator that allows access to the components array |
| Inline const T | Operator[] | Size\_t index | This is the overloaded [] that allows access to the components array but without allowing for change |
| Matrix4<T> | Operator\* | Const matrix4<T> &rhs | This is the overloaded \* operator which performs a matrix-matrix multiply and returns the result |
| Matrix4<T> | Operator\*= | Const matrix4<T> &rhs | This is the overloaded \*= operator which performs a matrix-matrix multiply and assigns the result to this matrix |
| Matrix4& | Make\_zero | Void | Zeros out this matrix |
| Matrix4& | Make\_identity | Void | Make this an identity matrix |
| Matrix4& | Make\_rotation\_x | Type radians | Makes this matrix a rotation matrix on the x axis |
| Matrix4& | Make\_rotation\_y | Type radians | Makes this matrix a rotation matrix on the y axis |
| Matrix4& | Make\_rotation\_z | Type radians | Makes this matrix a rotation matrix on the z axis |
| Matrix4& | Make\_scale | Type x, type y, type z | Makes this matrix a scale matrix based on the three values passed in. |
| Matrix4& | Make\_scale | Const vec3<VT> &v | Makes this matrix a scale matrix based on the vector passed in |
| Matrix4& | Make\_translation | Const vec3<VT> &v | Makes this matrix a translation matrix based on the vector passed in |
| Matrix4& | Rotate\_pre\_x | Type radians | Makes this matrix a rotation matrix on the x axis times itself. |
| Matrix4& | Rotate\_pre\_y | Type radians | Makes this matrix a rotation matrix on the y axis times itself. |
| Matrix4& | Rotate\_pre\_z | Type radians | Makes this matrix a rotation matrix on the z axis times itself. |
| Matrix4& | Rotate\_post\_x | Type radians | Makes a rotation matrix with this times a rotation matrix on the x axis |
| Matrix4& | Rotate\_post\_y | Type radians | Makes a rotation matrix with this times a rotation matrix on the y axis |
| Matrix4& | Rotate\_post\_z | Type radians | Makes a rotation matrix with this times a rotation matrix on the z axis |
| Matrix4& | Scale\_pre | Type x, type y , type z | Makes this matrix a scale matrix times itself based off the three components |
| Matrix4& | Scale\_pre | Const vec3<VT> &v | Makes this matrix a scale matrix times itself based off the vector passed in |
| Matrix4& | Scale\_post | Type x, type y , type z | Makes a scale matrix with this times a scale matrix, based on the three components |
| Matrix4& | Scale\_post | Const vec3<VT> &v | Makes a scale matrix with this times a scale matrix, based on the vector passed in |
| Matrix4& | Translate\_pre | Type x, type y , type z | Makes this matrix a translation matrix times itself based off the three components |
| Matrix4& | Translate\_pre | Const vec3<VT> &v | Makes this matrix a translation matrix times itself based off the vector passed in |
| Matrix4& | Translate\_post | Type x, type y , type z | Makes a translation matrix with this times a scale matrix, based on the three components |
| Matrix4& | Translate\_post | Const vec3<VT> &v | Makes a translation matrix with this times a scale matrix, based on the vector passed in |
| Matrix4& | Transpose | Void | Transposes this matrix |
| Matrix4& | Negate | Void | Negates this matrix |
| Matrix4& | Normalize3x3vectors | void | Normalizes the x, y and z axis |
| Friend void | Multiply | Matrix4<T> &matrixO, const matrix4<T> &matrixA, const matrix4<T> &matrixB | Performs a matrix-matrix multiply and stores the result in the out matrix. |

**Time to complete estimate:**

* Completed

**Module Author(s):**

* Engine Development 2 Lab Assistants (Joe and Clark)