Kvectors.c

Code documentation
1.12.2021
Release 1.0.0.

---

1. Introduction

This is a library written in programming language C to handle both spatial vectors and tuples of different types. The project is implemented through data polymorphism. The following will describe the api.

2. API

This part will describe the functions provided by the interface.

2.1 _GET_FUNCTION(__vectorN, T)* _GET_NAME(vectorN_create, T)();

This function allocates memory for the requested vector and returns a pointer to the allocated memory. If the memory could not be allocated, it returns zero and writes an error.

2.2 void _GET_NAME(vector2_init, T)(_GET_NAME(__vector2, T)* left, ...param...);

This function initializes the vector with values. Where parameters are a set of data. If n less than 5 you should pass individual variables, if n more than five you should pass an array of data. For example:

```
// N <= 5
void _GET_NAME(vector2_init, T)(_GET_NAME(__vector2, T)* left, T param1, T param2);
// N > 5
void _GET_NAME(vectorN_init, T)(_GET_NAME(__vectorN, T)* left, T* array);
```

2.3 double _GET_NAME(vectorN_norm, T)(_GET_NAME(__vectorN, T)* left);

This function returns the norm of the vector.

2.4 _GET_FUNCTION(__vectorN, double)* _GET_NAME(vectorN_normalize, T)(_GET_NAME(__vectorN, T)* left);

This function normalizes the vector and returns a pointer to it or zero in case of an error

2.5 void _GET_NAME(vectorN_add, T)(_GET_NAME(__vectorN, T)* result, _GET_NAME(__vectorN, T)* left, _GET_NAME(__vectorN, T)* right);

This is a function of the addition of two vectors. It takes a pointer to the resulting vector as the
first parameter. The second and third are the vectors to be added.

2.6 void _GET_NAME(vectorN_free, T)(_GET_NAME(__vectorN, T)* left);

This function frees previously allocated memory by the pointer “left”.

---

3. Example program

```
// Create three vectors of the float type

    __vector2_float* vec1 =  vector2_create_float();
    __vector2_float* vec2 =  vector2_create_float();
    __vector2_float* vec3 =  vector2_create_float();
// Initialize two of them with some numbers
    vector2_init_float(vec1, 12.0, 1.5);
    vector2_init_float(vec2, -1.0, 1.5);
// Add vector 1 and vector 2 and save the result in vector 3
    vector2_add_float(vec3, vec1, vec2);
// Print the result on the screen
    printf("%f %f\n", vec3->x, vec3->y);
```

Look file test.c for more examples program.