
1 The Vector Class

Design and implement a class named `Vector` that stores three `double` values (for example x , y , and z coordinates). The class should also have appropriate constructors and accessor functions for managing vector coordinates. In addition, the class must properly overload the following member functions.

- `operator==` return true if each elements of each `Vector` are equivalent. Be careful when comparing double precision values (never directly compare).
- `operator!=` return true if each elements of each `Vector` are **not** equivalent.
- `operator=` assign the `Vector` object on the RHS to the `Vector` object on the LHS.
- `operator+` return a `Vector` object representing the sum of two `Vectors`
- `operator+` return a `Vector` object representing the sum of a `const double` and each element of a `Vector`. This operation must be **symmetrical**.
- `operator[]` return or assign the element of a `Vector` object at a specified index. Indices **must** start with 1 (not zero). If the index is illegal, give an error message and return the value of the *closest* valid element. This function must be overloaded **twice**, providing a `const` and non-`const` version. Example operations include,

```
void printFirst(const Vector& v)
{ cout << v[1] << '\n'; }
Vector v;
v[1] = 99.0;           // assign first location 99
cout << v[1] << '\n'; // print first element
printFirst(v);       // print first element, passing as const reference
```

- `operator<<` stream insertion operator that **neatly** outputs the `Vector` in the form `[1.0, 2.0, 3.0]`

2 Programming Points

You **must** adhere to all of the following points to receive credit for this program.

- Turn-in (print-outs and electronically) the files for this program.
- The program must be broken into the following 5 files
 1. `vector.h` contains the class declaration
 2. `vector1.cpp` contains about one-half of the member function definitions
 3. `vector2.cpp` containing the other one-half of the member function definitions
 4. `driver.cpp` is program that tests all the functionality of your `Vector` class
 5. `makefile` is the `makefile` for the `Vector` class and `driver` program, must be commented and have `make clean` option