ECE 18847 Homework 1
Due 9/26/2024

September 10, 2024

Implement the Classes `Box.cpp` and `MDArray.cpp` corresponding to the headers `Box.H` and MDArray.H, respectively. Test the functions by compiling making them along with mdarray-Main.cpp, and run the code with command-line inputs 16 16, 32 32, 64 64.

The mathematical specification of the member functions are given as follows.

`Box.cpp`:

- `Box()`, `Box(const int a_lowCorner[DIM],const int a_highCorner[DIM])`, `Box(const Box& a_Box)` : constructors.
- `Box operator&&(const Box& a_rightBox) const` : intersection operator - if A and B are boxes, then `A&&B` returns `A` $\cap$ `B`.
- `Box shift(int a_direction, int a_offset)`: shift the location of the `Box`: $B.shift(i, d, s) = \{\boldsymbol{i} + s\boldsymbol{e^d} : \boldsymbol{i} \in B\}$, where $\boldsymbol{e}^d$ is the unit vector in the d direction, and s is the size of the offset.
- `Box grow(int a_numpoints)`: grow the box by `a_numpoints` in all directions.
- `void getLowCorner(int a_lowercorner[DIM]) const`,
  `void getHighCorner(int a_lowercorner[DIM]) const`: copy low, high corners of Box into input tuples.
- `int linearIndex(const int a_tupleIndex[DIM]) const`: Convert tuple location in the `Box` into a linear index, corresponding to where the data would be stored for an array defined over that `Box`. In 2D, for example, if $B = [i_L, i_H] \times [j_L, j_H]$, and arrays are to be stored in row-major order, then the linear index of (i, j) is given by $ind = i - i_L + (j - j_L)(i_H - i_L + 1)$.
- `void tupleIndex(int a_linearIndex, int a_tupleIndex[DIM]) const`: compute the tuple corresponding to the linear index and copies it into `a_tupleIndex[DIM]`. Inverse of the operator given above.
- `int sizeOf() const`: retuns the number of points in the `Box`.
- `bool operator==(const Box& a_rhsBox) const`: return `true` if the `Box` is equal to the input, otherwise `false`.

MDArray.cpp

- `MDArray()`, `MDArray(Box a_box)`: constructors.
- `void define(Box a_box)`: defines an `MDArray` that has been default-constructed.
- ∼`MDArray()`: destructor.
- `void operator+=(const MDArray& a_rhs)`, `void operator-=(const MDArray& a_rhs)`, `void operator*=(const MDArray& a_rhs)`, `void operator/=(const MDArray& a_rhs)`: pointwise operators. `A+=B` replaces the values $A_{\boldsymbol{i}}$ by $A_{\boldsymbol{i}} + B_{\boldsymbol{i}}$ for all tuples in `m_box`. Only defined if the `Boxes` for `A` and `B` are identical. Similarly for `-=,*=, /=`.
- `void shift(int a_dir, int a_len)`: shift operator. Translates `m_box` by `a_len` in the `a_dir` direction, leaving the data unchanged.
- `float& operator[](int a_index[DIM]) const`: indexing operator - return a reference to $A_{\boldsymbol{i}}$ for the input tuple $\boldsymbol{i}$.
- `float& indexShift(const int a_tuple[DIM], const int& a_dir,const int& a_shift) const`: `A.indexshift`$(\boldsymbol{i}, d, s)$ returns a reference to $A_{\boldsymbol{i}+s\boldsymbol{e}^d}$ for input tuple $\boldsymbol{i}$, direction d, and shift distance s.
- `float& operator[](int a_linearIndex) const` : return a reference to `m_data[a_linearIndex]`.
- `const Box& getBox() const`: return a const reference to `m_box`.
- `void operator+=(const float& a_rhs)`, `void operator-=(const float& a_rhs)`, `void operator*=(const float& a_rhs)`, `void operator/=(const float& a_rhs)`: pointwise scalar operators. `A+=a` replace all the values $A_{\boldsymbol{i}}$ by $A_{\boldsymbol{i}} + a$. Similarly for `-= , *=, /=`.

Expected output when compilation succeeds:

```
> make clean
rm *.o a.out *.exe
> make all
clang++ -g -DDIM=2 -c -g -o MDArray.o MDArray.cpp
clang++ -g -DDIM=2 -c -g -o Box.o Box.cpp
clang++ -g -DDIM=2 -c -g -o mdarrayMain.o mdarrayMain.cpp
clang++ -g -DDIM=2 MDArray.o Box.o mdarrayMain.o
clang++ -g -DDIM=2 -c -o boxTester.o boxTester.cpp
clang++ -g -DDIM=2 -o boxTester.exe Box.o boxTester.o
clang++ -g -DDIM=2 -DDIM=1 -Wno-macro-redefined -c -g -o Box1D.o Box.cpp
clang++ -g -DDIM=2 -DDIM=1 -Wno-macro-redefined -c -o boxTester1D.o boxTester.cpp
clang++ -g -DDIM=2 -o boxTester1D.exe Box1D.o boxTester1D.o
clang++ -g -DDIM=2 -DDIM=3 -Wno-macro-redefined -c -g -o Box3D.o Box.cpp
clang++ -g -DDIM=2 -DDIM=3 -Wno-macro-redefined -c -o boxTester3D.o boxTester.cpp
clang++ -g -DDIM=2 -o boxTester3D.exe Box3D.o boxTester3D.o
```

Expected output when execution succeeds:

```
 make run
./boxTester.exe
detected 0 errors in this test program
./boxTester1D.exe
detected 0 errors in this test program
./boxTester3D.exe
detected 0 errors in this test program
./a.out 16 16
max error:1.009445e+00
at location [4, 4]

./a.out 32 32
max error:2.532196e-01
at location [8, 8]

./a.out 64 64
max error:6.327820e-02
at location [16, 16]
```