1 Linking

The following instructions are made painstakingly detailed with the hope that even a person with zero linking experience will manage to link successfully. They describe how to dynamically link the Long Integers library to a MSVC 2012 project.

1) Download and unpack the appropriate archive to a certain directory of your choice. Once unpacked it should include the following files

```
longInt.h
longIntNonMembers.h
stdafx.h
targetver.h
LongInt.lib
LongInt.dll
```

2) Create a new solution and project; (**File→New Project**: keep **Win32 Console Application**, fill in **Name** and **Solution Name**; hit **Ok** and then **Next**; under **Application type** check **Console application** and, optionally, under **Additional options** uncheck **Precompiled header**; hit **Finish**);

3) Remove all automatically generated .h and .cpp files file from your project; (In the **Solution explorer** (a vertical panel at the right-hand side) right-click on the file name and select **Remove→Delete**);

4) Add the downloaded header files, i.e., `longInt.h`, `longIntNonMembers.h`, `stdafx.h`, and `targetver.h`, to the project; these files do not need to be in your solution or project directory; they can be anywhere; (**Project→Add** existing item: select the files and hit **Add**); make sure that all .h files are under **Header files** and not under **Source files**, and all .cpp files are under **Source files** and not under **Header files**; if you find a file that is at the wrong place, drag it to its proper location);

5) Copy or create any other .h and .cpp files that are to be used and compiled into your console application to the project's working directory which by default is the project subdirectory of your solution directory (these directories should be already automatically created by MSVC 2012); (To find the current project's working directory, in the "Solution explorer" (a vertical panel at the right-hand side) right-click on the project (not any of the files but the very project) and go **References→Configuration properties→Debugging**. On the right side, look to the right of **Working directory** to see or change the currently specified working directory.)

6) Copy the `LongInt.dll` file to the project's working directory; (see step 5 for finding the project's working directory);

7) Add the directory (or directories) where the header files from step 4 are to your include list; (In the **Solution explorer** (a vertical panel at the right-hand side) right-click on the project (not any of the files but the very project) and go **References→Configuration properties→C/C++→General** (**Configuration properties** and **C/C++** would require double-clicks to open their lists of subcategories); click on the field **Additional Include Directories** to the right; press the **down** pointer at the right end of the edit line and select **Edit** from the drop down menu; click the **New line** icon (yellow folder) and then click the **Browse button** (the three dots ... button) to browse for the

directory; repeat that procedure if more than one directory need to be included; hit **Ok**, **Apply**, and **Ok** again);

8) Specify the LongInt.lib file and its directory (In the **Solution explorer** (a vertical panel at the right-hand side) right-click on the project (not any of the files but the very project) and go **References→Configuration properties→Linker→General** (**Configuration properties** and **Linker** would require double-clicks to open their lists of subcategories); click on the field **Additional Library Directories** to the right; press the **down** pointer at the right end of the edit line and select **Edit** from the drop down menu; click the **New line** icon (yellow folder) and then click the **Browse button** (the three dots ... button) to browse for the directory; repeat that procedure if more than one directory need to be included; hit **Ok**, **Apply**; this specified the directory; to specify the .lib file, on the same window, click the **Input** (i.e., **References→Configuration properties→Linker→Input**); click on the field **Additional Dependencies** to the right; press the **down** pointer at the right end of the edit line and select **Edit** from the drop down menu; type in LongInt.lib; hit **Ok**, **Apply**, and **Ok** again);

9) Build the library (**Build→Build solution**);

10) Start the executable to check if it runs (**Debug→Start Debugging**).

2 Use

Long Integers can be constructed from regular integers, strings of digits, or vectors of digits. For example,

```
LongInt a(56);
LongInt b("-435476956123402938740981238 4");
```

One can use expressions like the following with predictable results:

```
a++
++a
a--
--a

a + b
a - b
a * b
a / b
a % b
a = pow(b, 26)

a = b
a += b
a -= b
a *= b
a /= b
a %= b

a == b
a != b
a > b
a < b
a >= b
a <= b

cout << a
cin >> a
```

Note that the exponent in the power function can be only a "regular" non-negative integer. For other available, but less common functions, please, consult the header files `longInt.h` and `longIntNonMembers.h`.