How to set up and run MS MPI using MS Visual Studios

With Lots of Pictures!

~Michael Puthawala

Preliminaries, references

- For a basic tutorial on coding with MPI, check this tutorial at LLNL.
- Here is another, shorter tutorial on compiling and running a simple MPI program using MS-MPI.
- Here is a link to the download page for MS-MPI.
- If you are just looking to download MS MPI and smpd, then you can download them here. If you are feeling saucy, then you can also download the entire MS HPC (high performance computing) pack cluster, but you won't need most of it if you just want to run and debug simple MPI programs.[1]
- Finally, here is a link to a forum where you can post questions/problems with your MPI code. The is the official support forum for people using MS MPI.

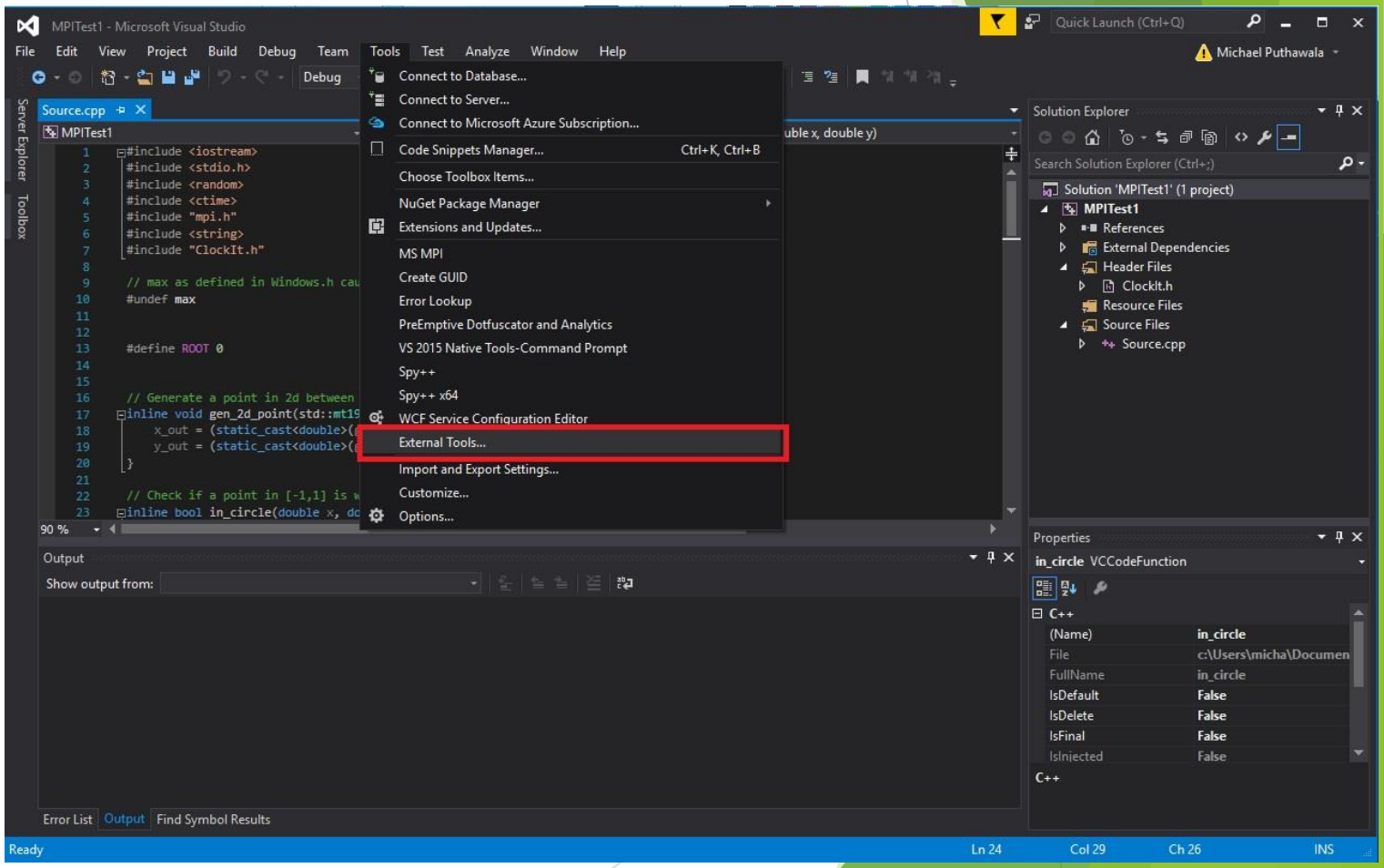
1 But make sure that you have EXACTLY 1 version of MS MPI installed in either way! If you install both MS MPI and MS HPC you risk having problem further down the line (see slide 7)

Setting up a native tools command prompt.

- In order for this to go smoothly, it’s useful to have access to a windows command prompt, with the proper configurations.
- This part will guide you through this process.
- You can try and use your own cmd.exe. If you don’t configure it properly, windows will think that you are a plebeian user, and won’t give you access to some powerful commands that you want.
- The following will let you open a command prompt that is properly configured for your super-user needs.

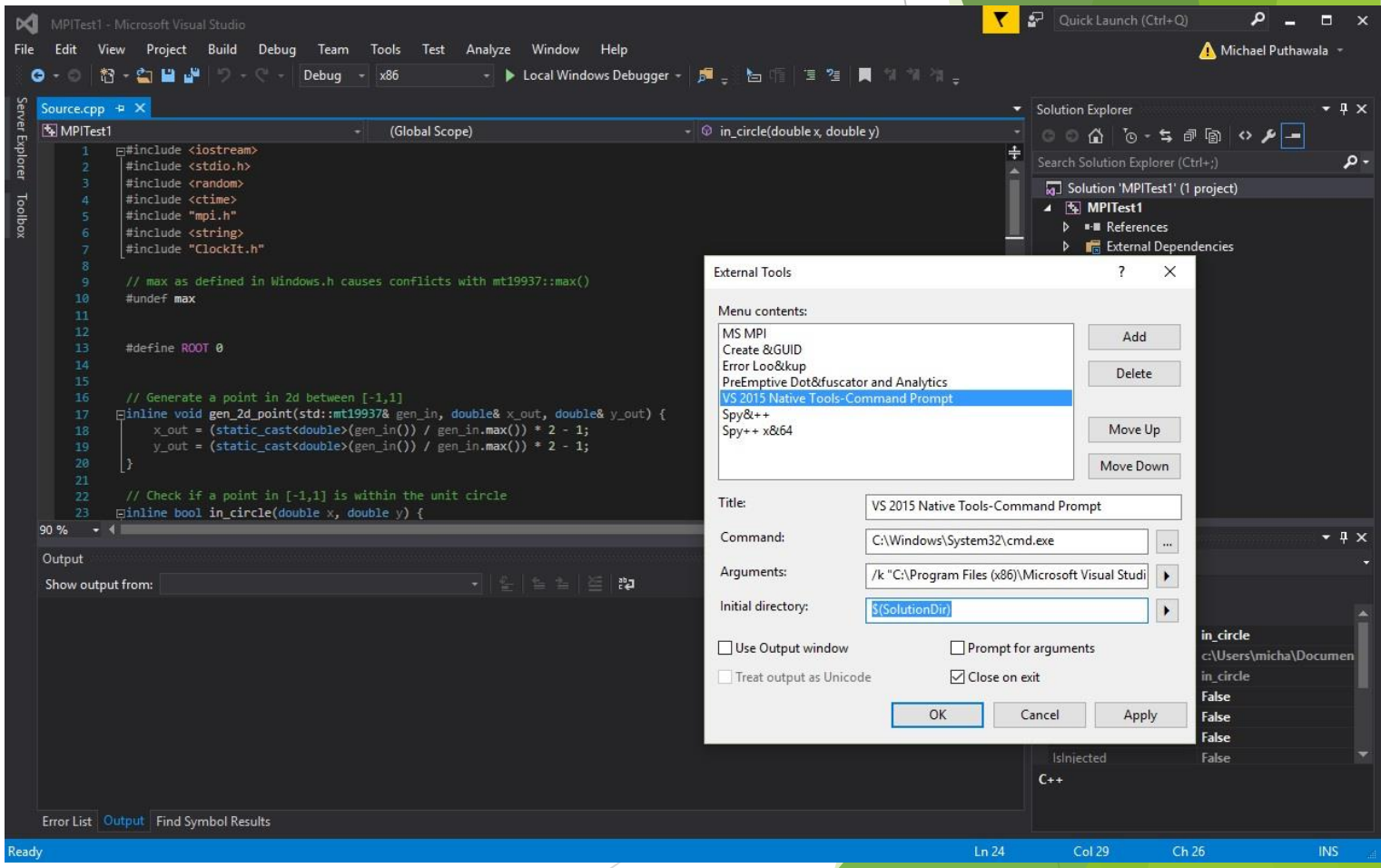
Setting up a native tools command prompt.

- Open MSVS, and to go tools -> External Tools…



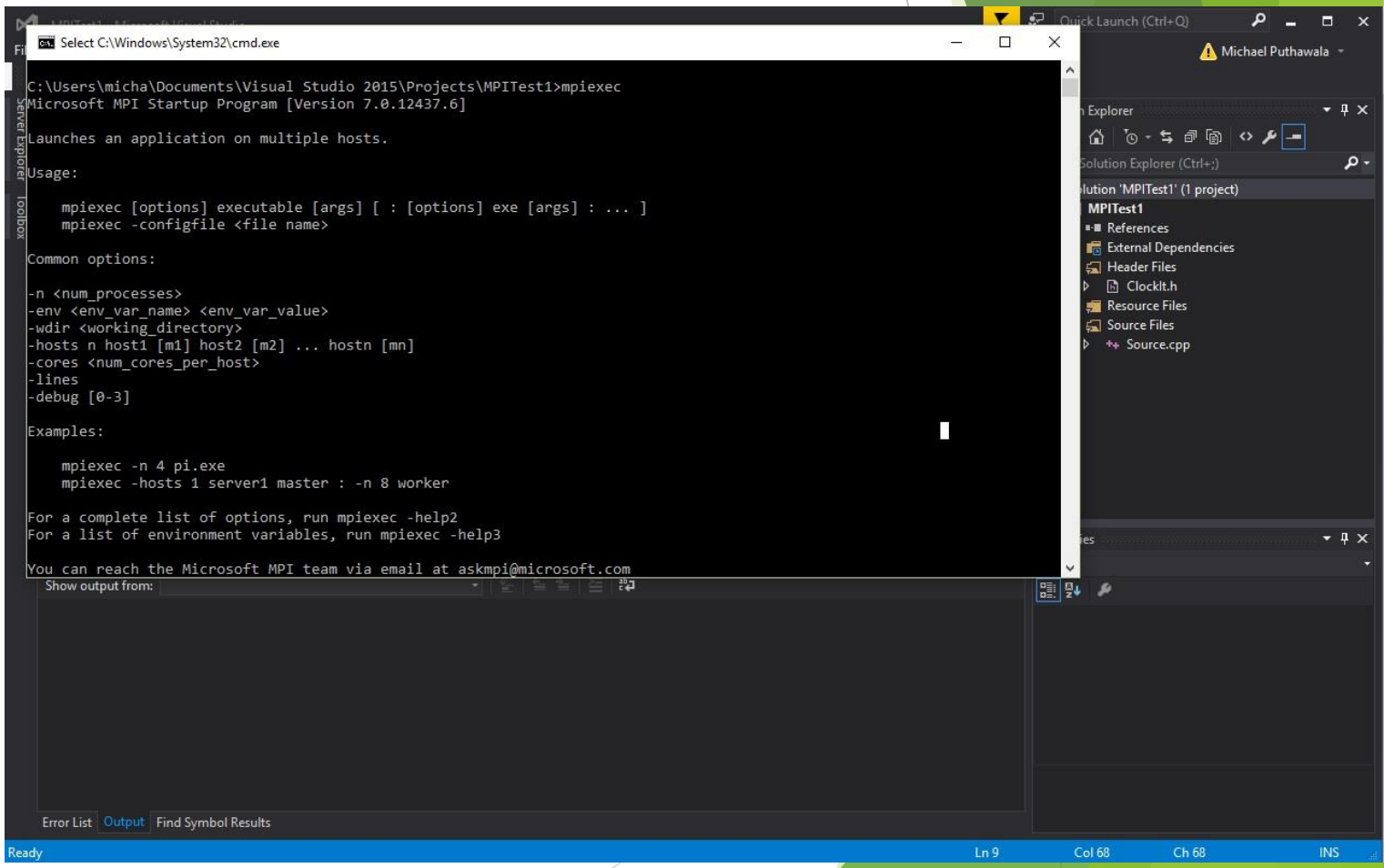
Setting up a native tools command prompt.

- Click Add, and fill in the following fields:
- Title: VS 2015 Native Tools-Command Prompt [or whatever else you want your tool to be called]
- Command: C:\Windows\System32\cmd.exe
- Arguments: /k "C:\Program Files (x86)\Microsoft Visual Studio 14.0\Common7\Tools\VsDevCmd.bat "
- Initial directory: $(SolutionDir)



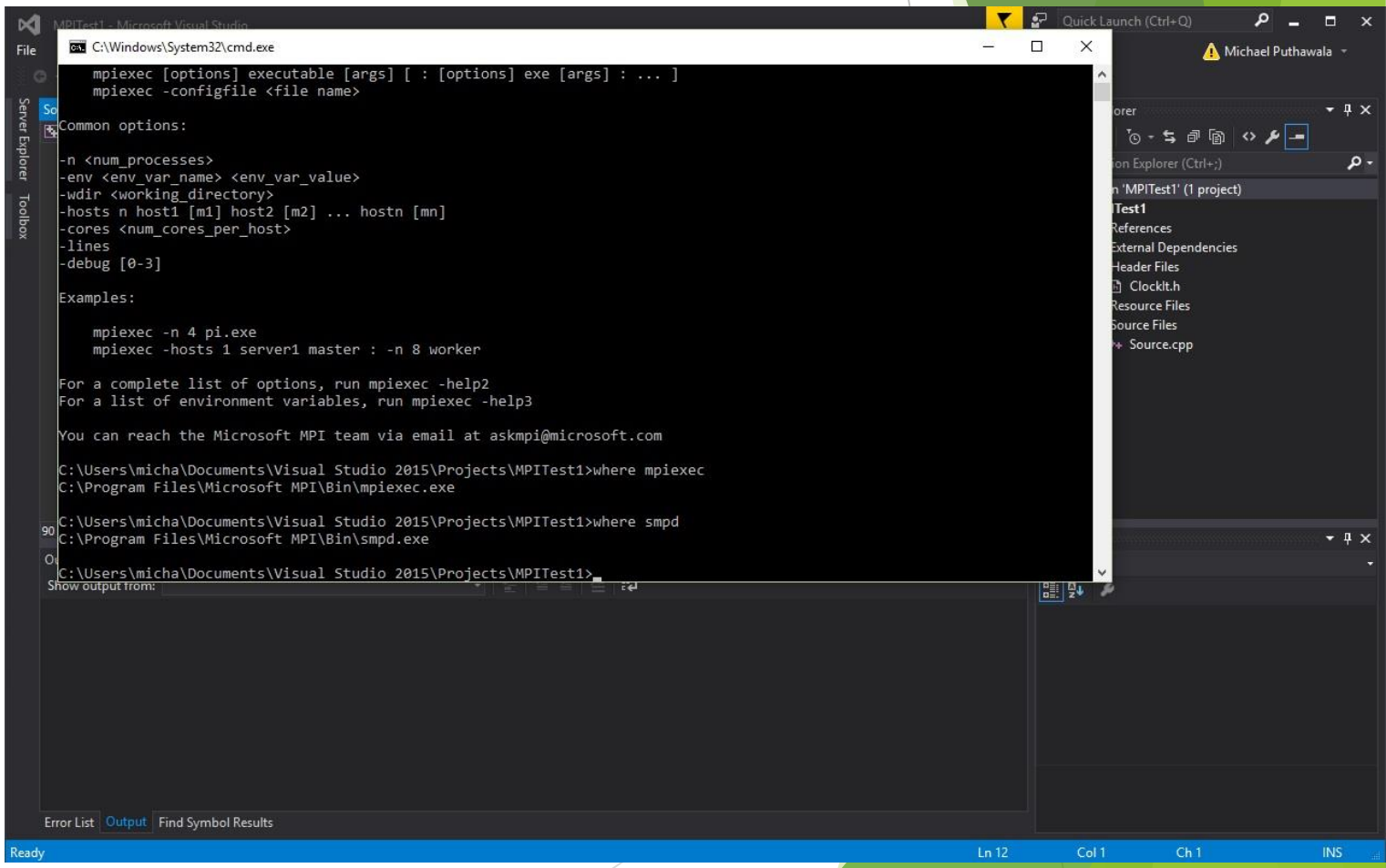
Setting up a native tools command prompt.

- Now, you should have a new option under Tools. Running your new tool will open a command prompt which is properly configured.
- Take this opportunity to make sure that mpiexec is in your system's path
- Type mpiexec into the command prompt, and make sure that you see something like the following



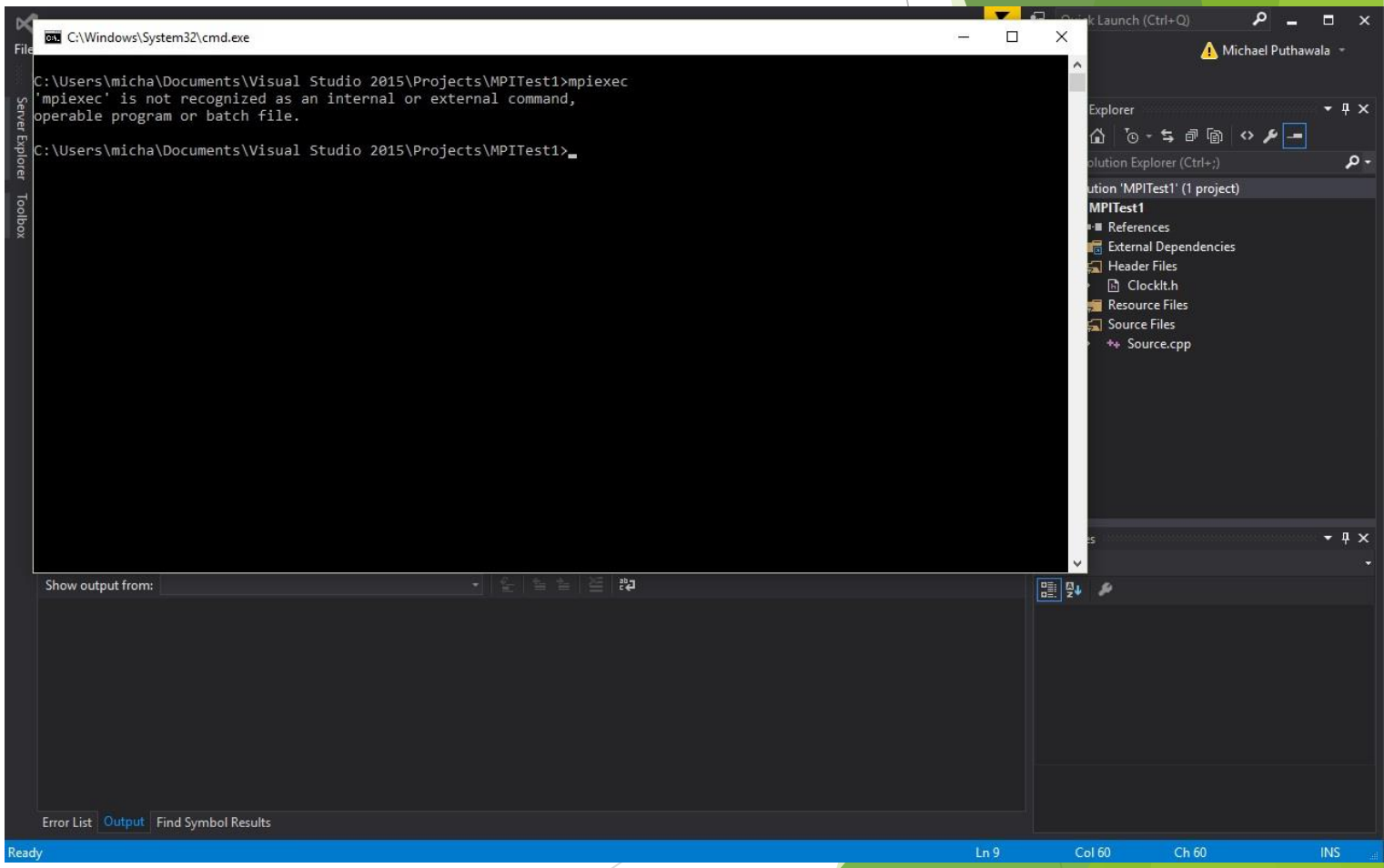
Setting up a native tools command prompt.

- You should also take this opportunity to make sure that you have exactly one version of mpiexec and smpd installed.
- Type where mpiexec and where smpd in the command prompt and make sure that you get 1 result.
- If you have more than 1 version installed, uninstall one or risk a version mismatch.



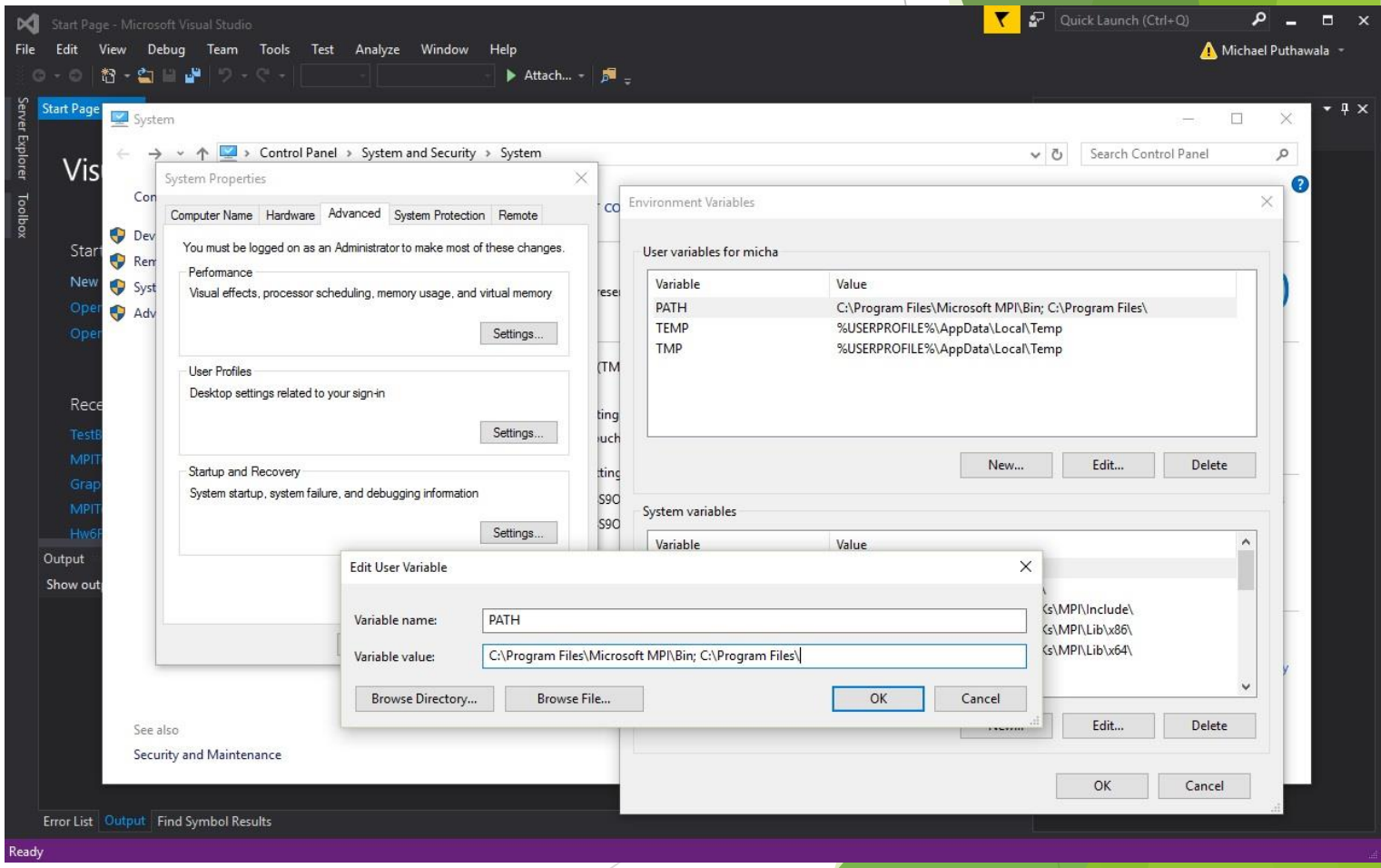
Configuring your system PATH variable

- If you instead see something like this on the right, then msmpi isn't in the default path for your command prompt.



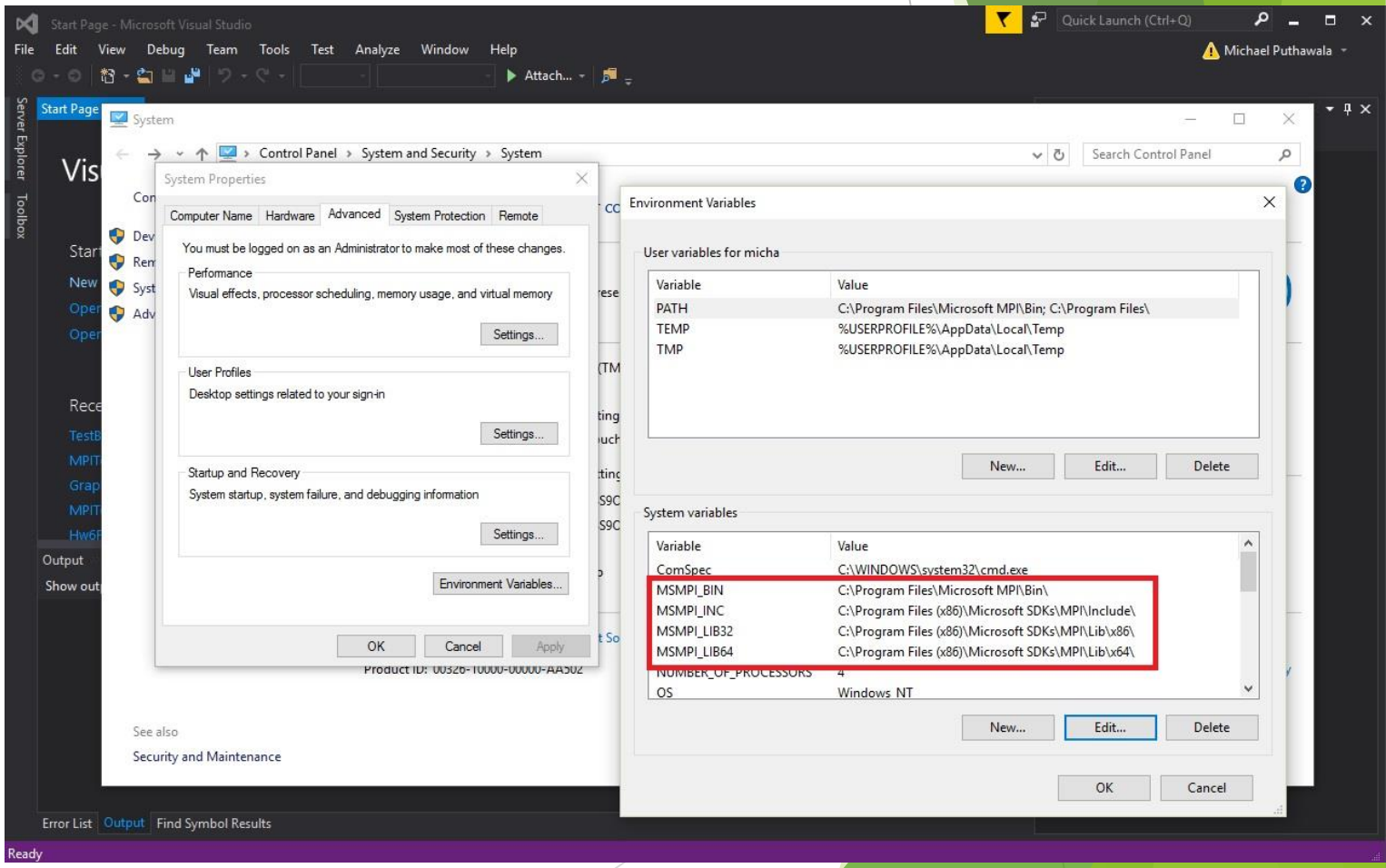
Configuring your system path

- There are lots of ways to fix this, but one easy (and permanent way) to fix this, is to go to control panel -> system -> advanced system properties -> environment variables and add C:\Program Files\Microsoft MPI\Bin to your PATH variable.
- If you don't have a PATH, then you will need to make a new one
- If you already have a PATH variable, and don't want to change it, you can add a new path with a ;
- Then, restart MSVS and try running mpiexec again.



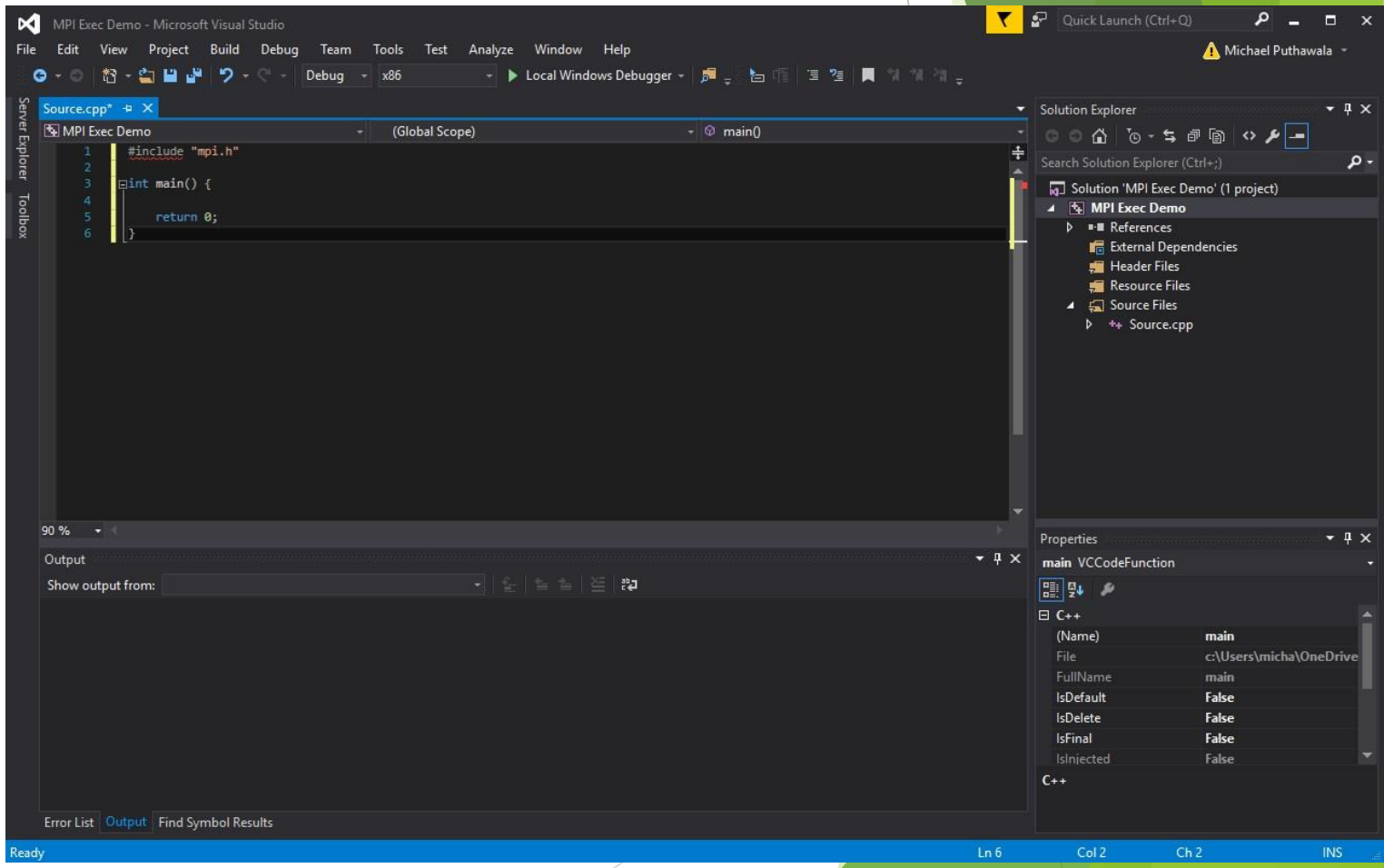
Configuring your system path

- You should have some important system variables set, namely MSMPI_BIN, MSMPI_INC, MSMPI_LIB32 and MSMPI_LIB64.
- These should have been set when you installed MSMPI.



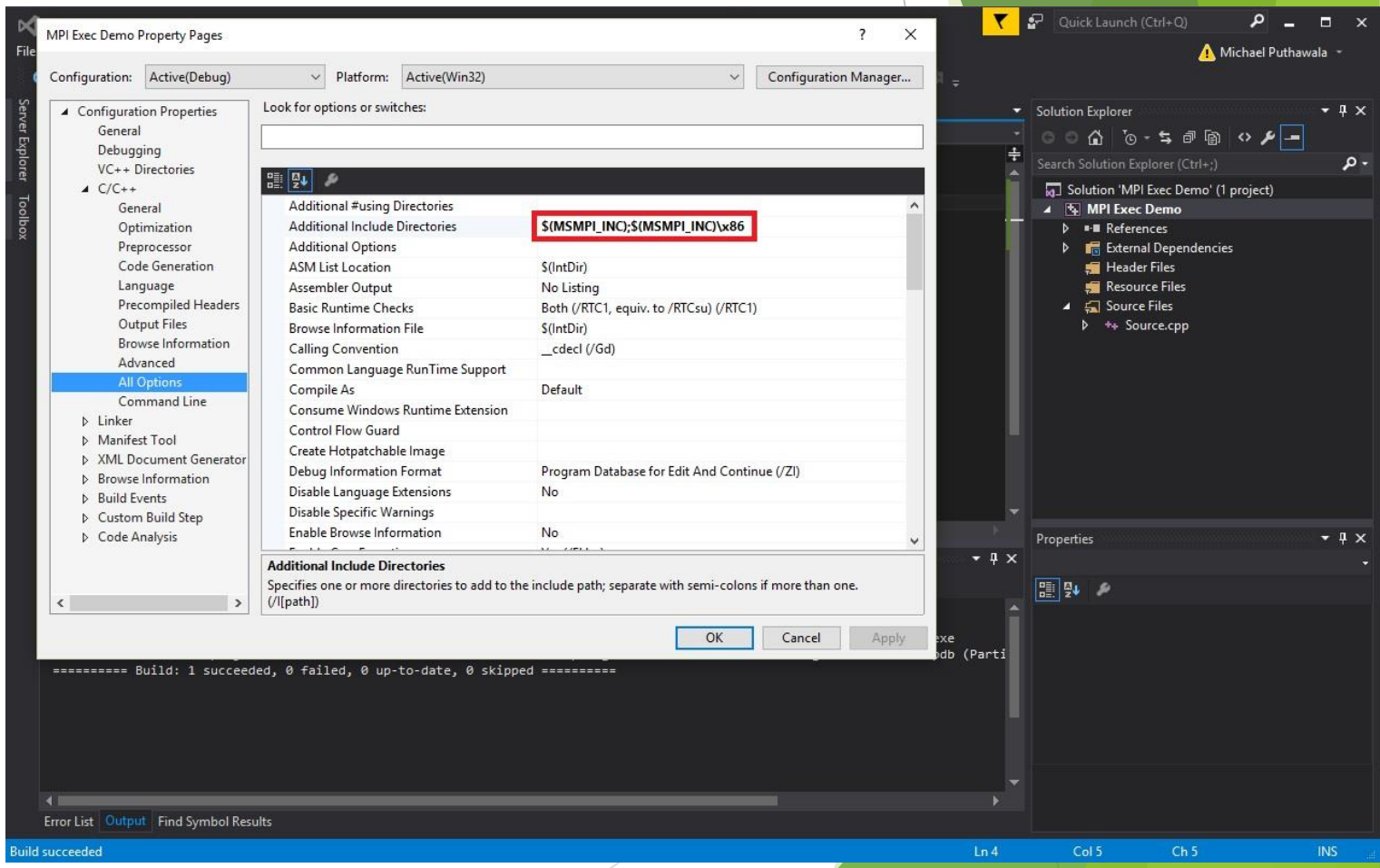
Setting up your first MPI Program

- Make sure to #include “mpi.h” at the top of your program.



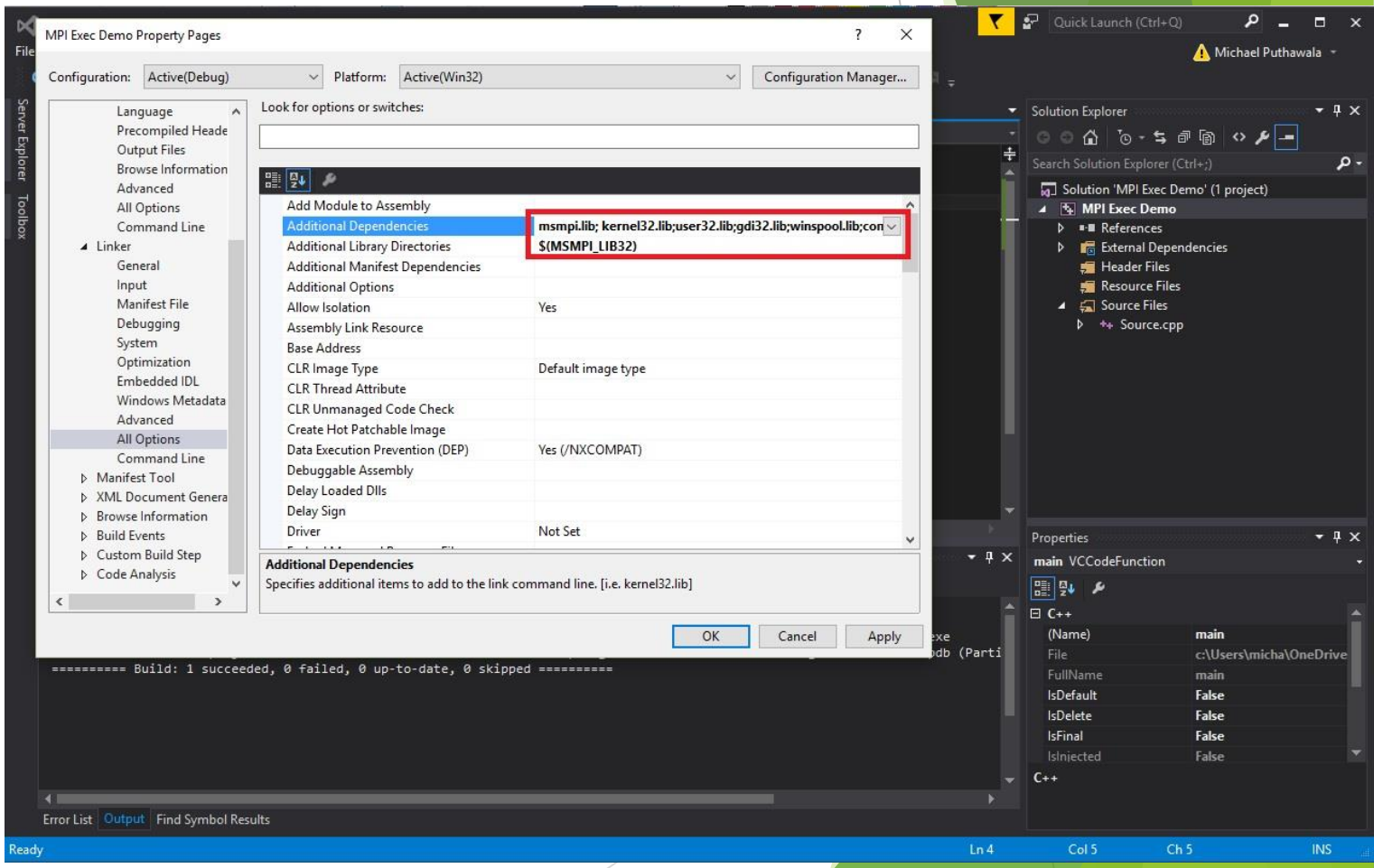
Setting up your first MPI Program

- Next, go into Project -> properties -> c/c++ -> All options and add $(MSMPI_INC); $(MSMPI_INC)\x86 to your addition include directories.
- Note, if you are using a 64 bit,then replace $(MSMPI_INC)\x86 with $(MSMPI_INC)\x64



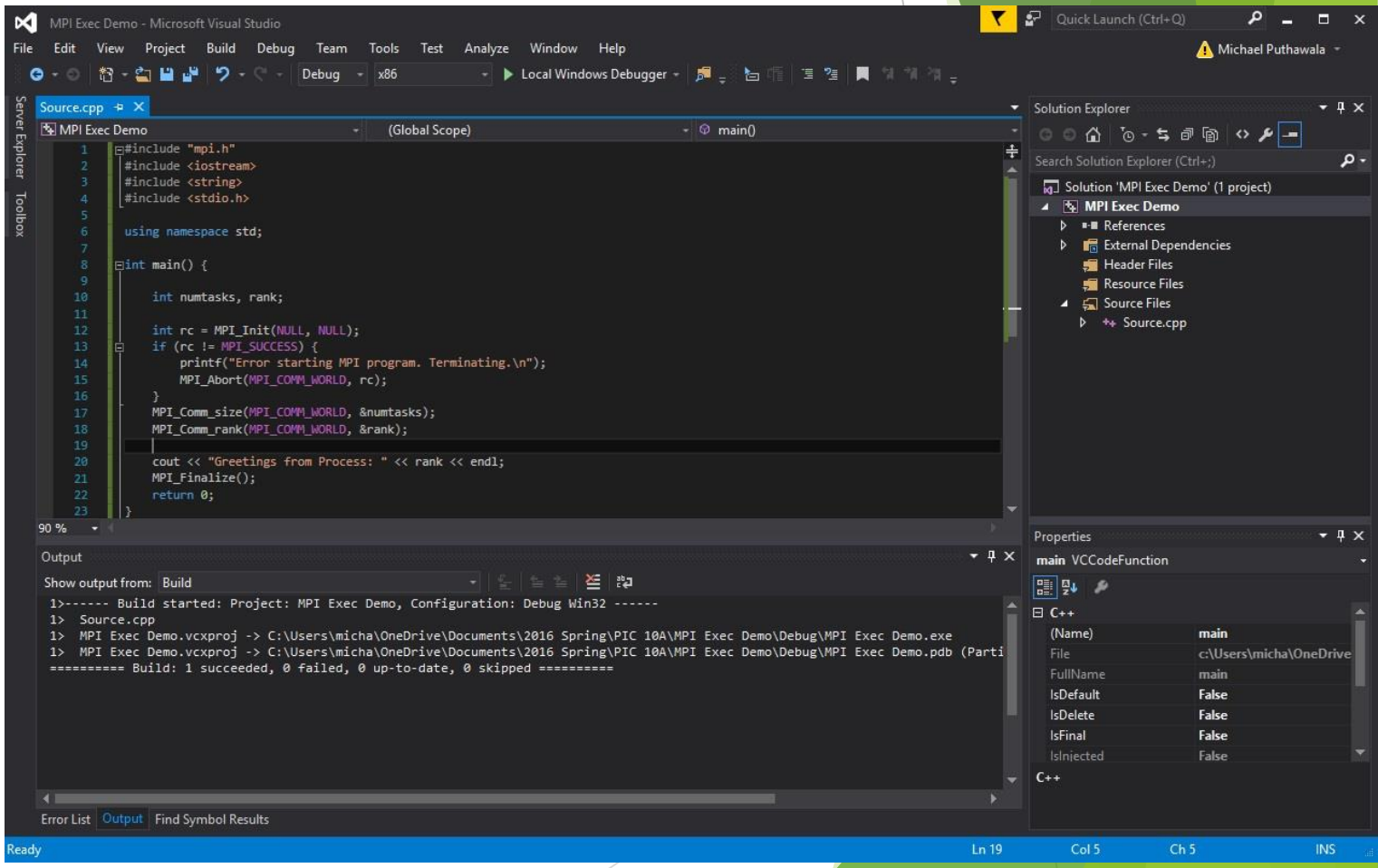
Setting up your first MPI Program

- Next, go to Project -> Properties -> Linker -> All options and add msmpi.lib; to the Additional Dependencies and $(MSMPI_LIB32) to Additional Library Directories.
- Again, if you are using 64 bit, then replace $(MSMPI_LIB32) with $(MSMPI_LIB64).



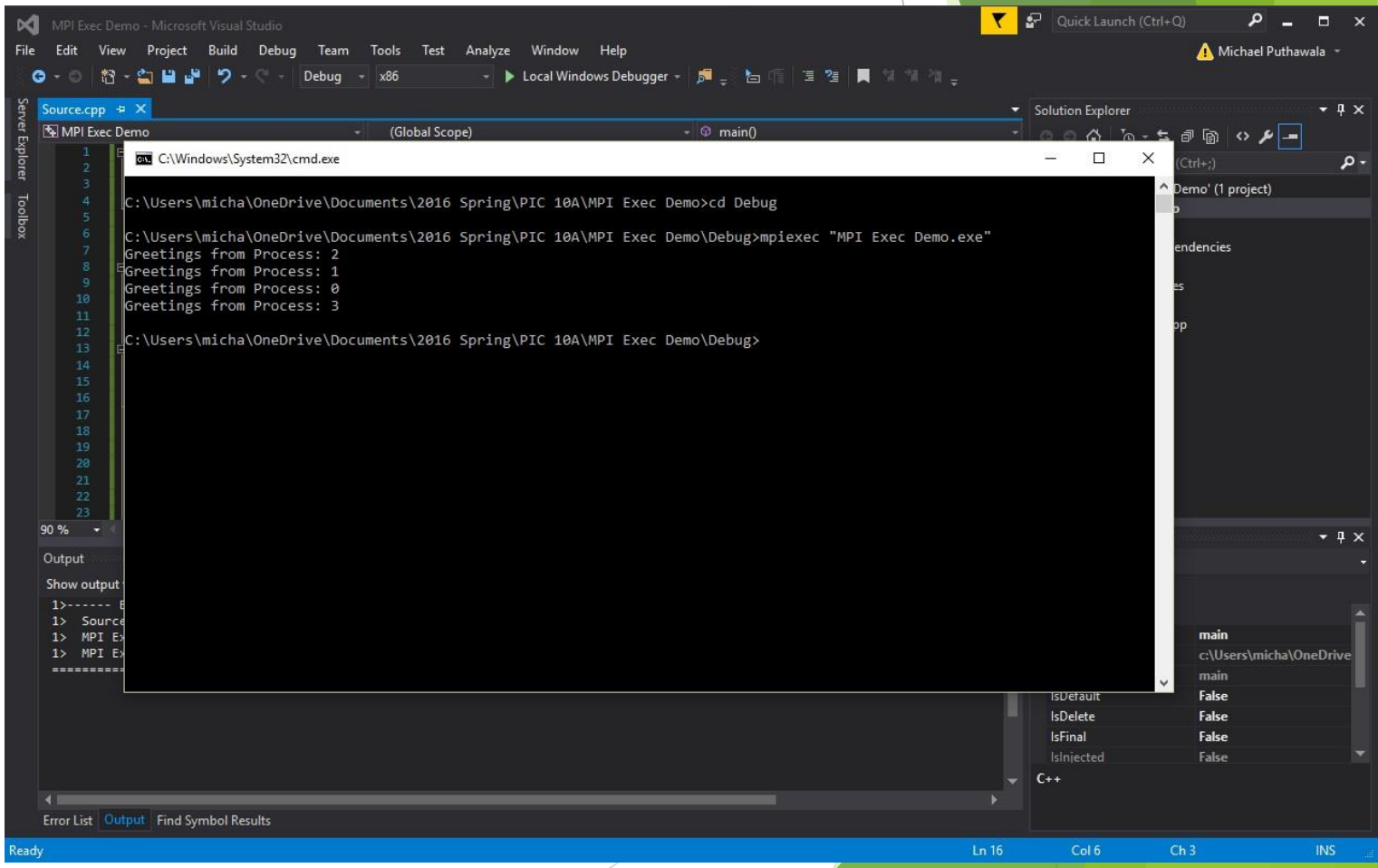
Setting up your first MPI Program

- Copy the code on the right into your empty project, and make sure that it builds.



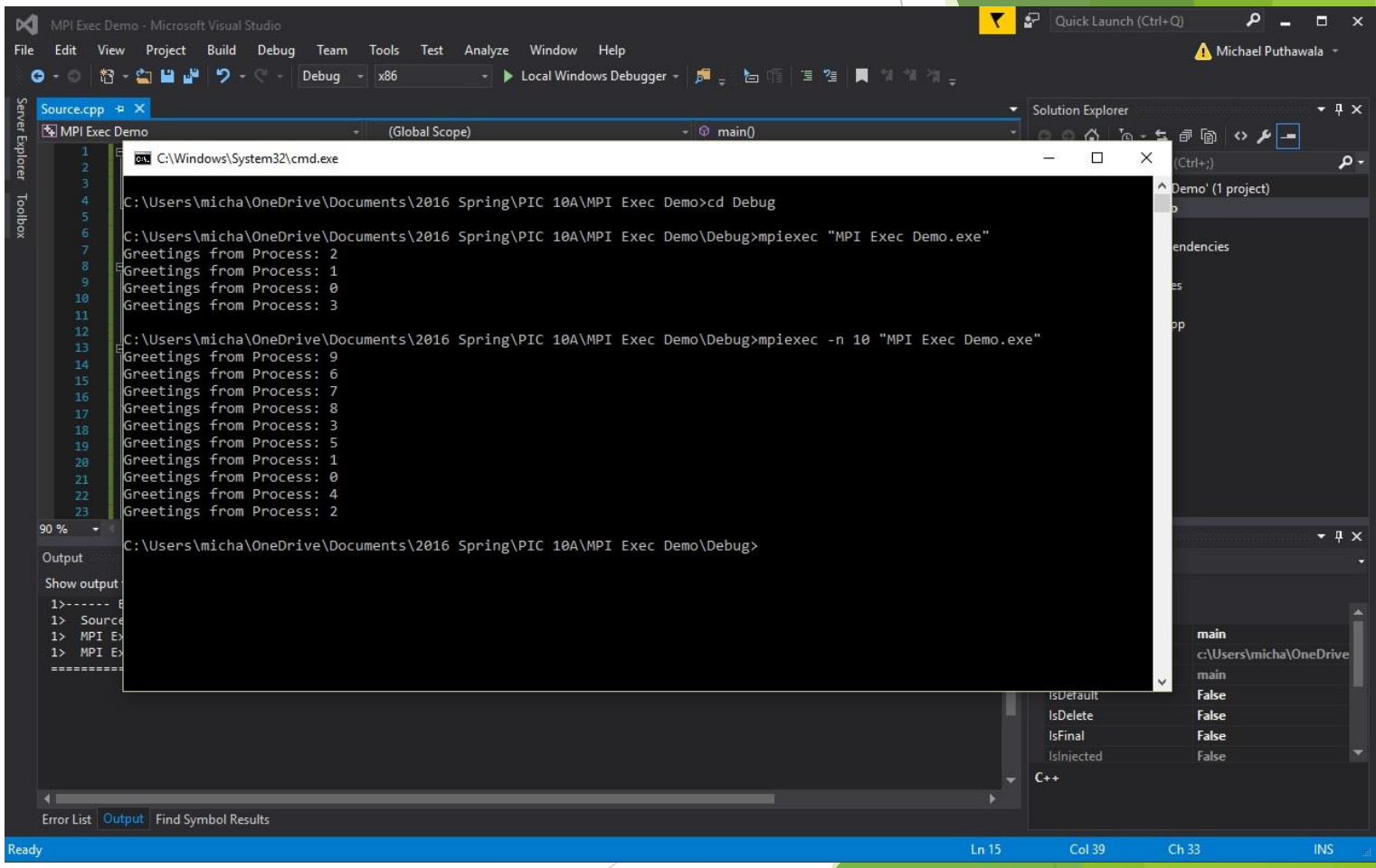
Setting up your first MPI Program

- Open up for fancy command prompt, cd to the project directory, and run mpiexec on your new program!



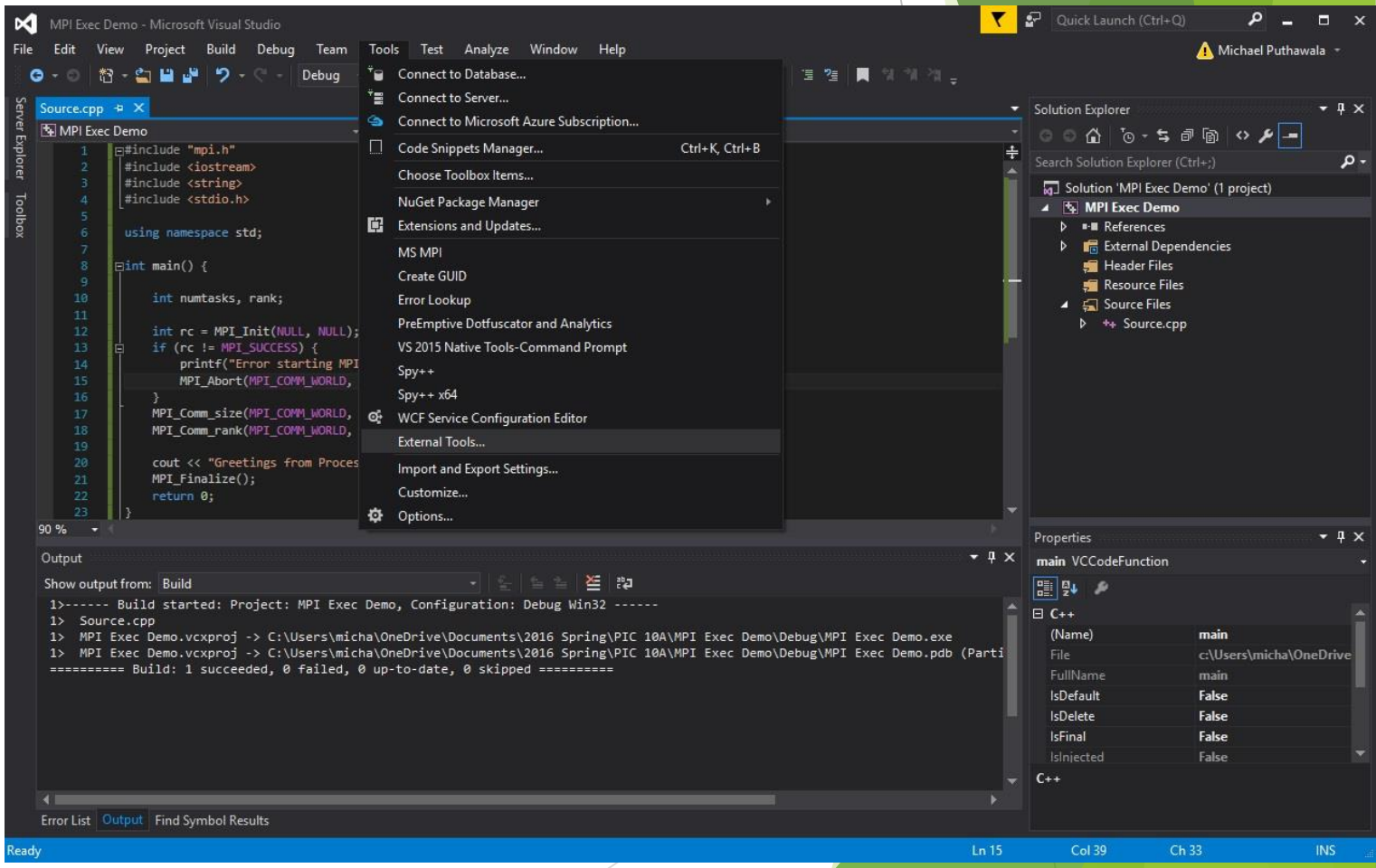
Setting up your first MPI Program

- You can run your code with more processes by including –n # where # is any natural number.
- If you don't specify how many processes to create, MPI will automagically choose the number of processes, to be the number of cores available to you.



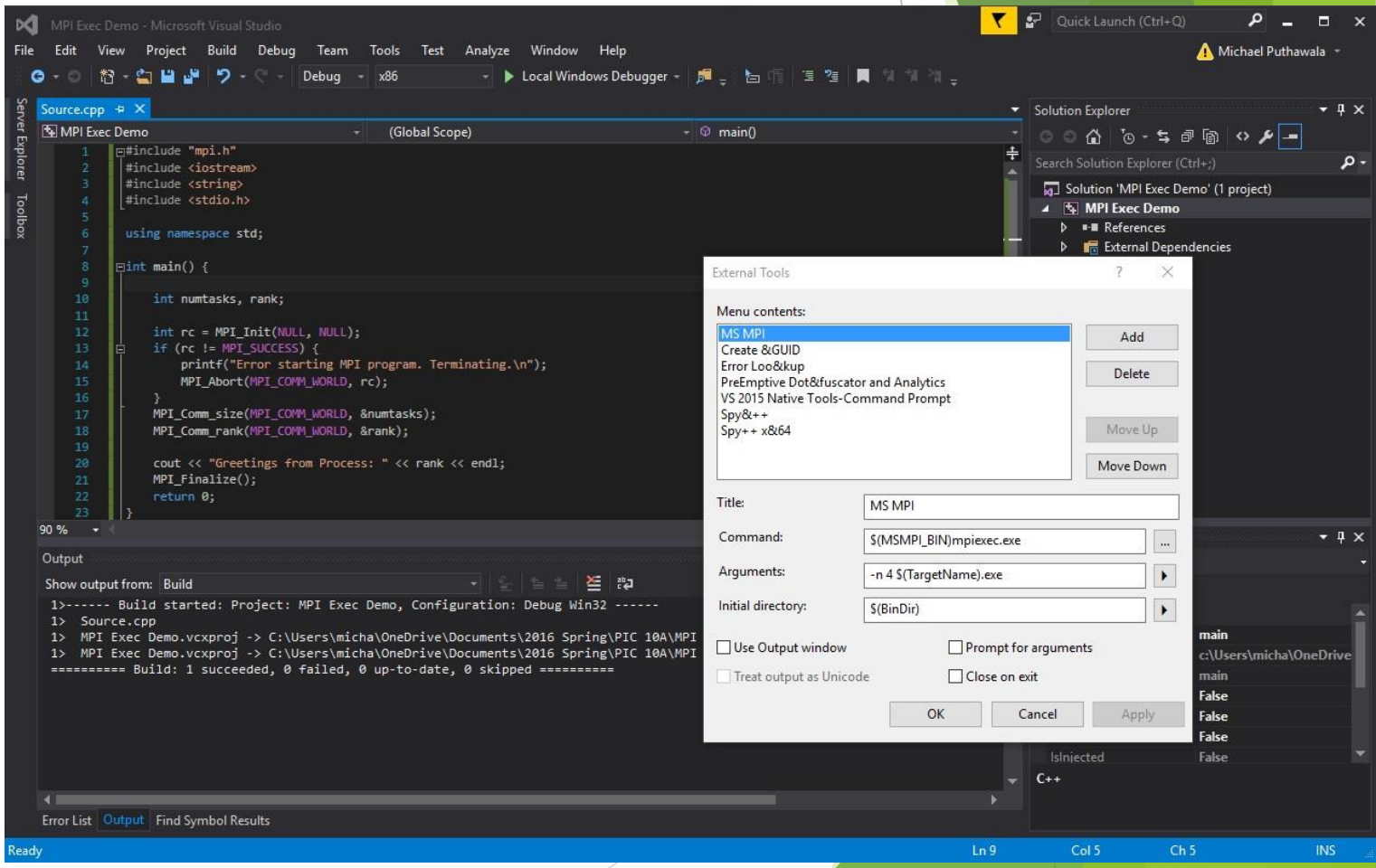
Setting up 2 click MPI

- If that process of opening up a new command prompt is too much work for you, then you can add another external tool which will automatically run any of your projects with mpiexec.



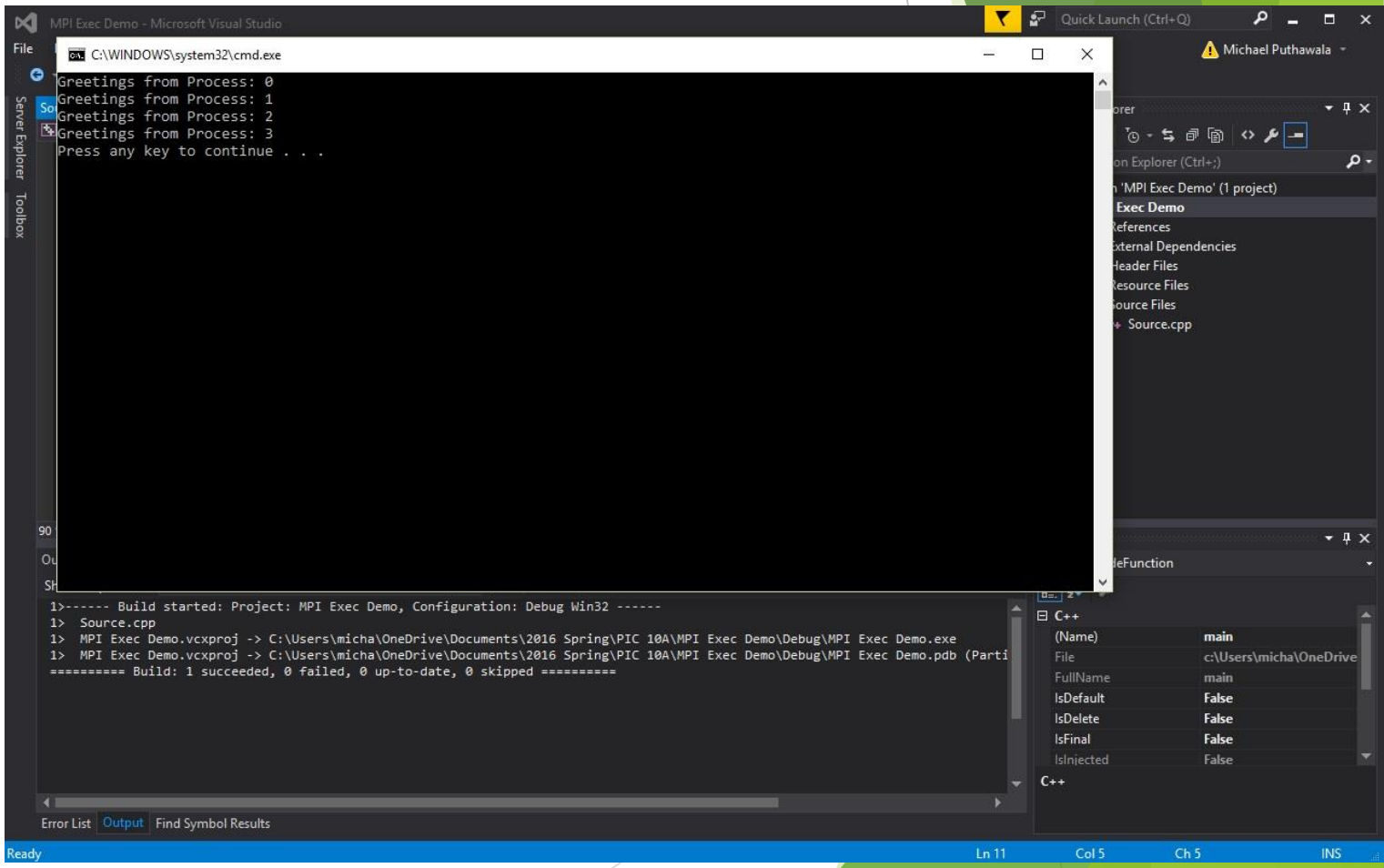
Setting up 2 click MPI

- Add a new tool, and specify it using the following:
- Title: MS MPI
- Command: $(MSMPI_BIN)mpiexec.exe
- Arguments: $(TargetName).exe
- Initial Directory: $(BinDir)
- Also, make sure that the close on exit button is unchecked.



Setting up 2 click MPI

- Now if you want to run your MPI Progam, you just have to build it, and then click Tools -> MS MPI



Conclusion

- MPI is very powerful, fun to program with, but a pain in the ass to install and get working on your first project.
- Hopefully this tutorial make the process of installing/configuring/using MSMPI slightly less terrible.
- If you have any questions, or this tutorial helped you in some way, don't be afraid to let me know at michaelputhawala@gmail.com