Introduction

Mina Konaković

Prerequisite: C++

- C++ Tutorial:

  http://www.cplusplus.com/doc/tutorial/

- Transition from Java:

  http://cs.brown.edu/courses/cs123/docs/java_to_cpp.shtml

- Book: C++ Primer (5th Edition). Stanley B. Lippman, Josée Lavoie, Barbara E. Moo. Addison-Wesley, 2012.

Eigen

- Eigen is a C++ template library for linear algebra
  - matrices, vectors, numerical solvers, and related algorithms.
- More on http://eigen.tuxfamily.org/
- See slides at the end on how to compile/build/run

Eigen Tutorial : Vectors (1/2)

- Initializing a vector

```
Eigen::Vector3f u, v;
u(0) = 1.0;
u(1) = 0.0;
u(2) = 0.0;
// Another way of initializing a vector/matrix
v << 0.0, 1.0, 0.0;
```

- Display them to standard input

```
cout << "u: " << u.transpose() << endl;
cout << "v: " << v.transpose() << endl;
```

Eigen Tutorial : Vectors (2/2)

- Vector operations

```
auto w = u.cross(v);
cout << u.transpose() << " cross " << v.transpose() << " = " << w.transpose() << endl;
double dot = u.dot(v);
cout << u.transpose() << " dot " << v.transpose() << " = " << dot << endl;
```

- More vector operations

```
auto uHat = u.normalized();
u.normalize();
auto normU = u.norm();
auto normUHat = uHat.norm();
cout << "normU: " << normU << endl;
cout << "normUHat: " << normUHat << endl;
```

```
// Notice that these two now have the
// same value. Eigen normalizes in place.
// Note: be careful in performing
// A=A.something()! See "ALIASING" topic in Eigen
```

Eigen Tutorial : Matrix

- Matrix setup

```
Eigen::Matrix2f m;
m(0, 0) = 3;
m(1, 0) = 2.5;
m(0, 1) = -1;
m(1, 1) = m(1, 0) + m(0, 1);
cout << "\nMatrix m:\n" << m << endl;
```

```
// Again, inline initialization
Eigen::Matrix3f n;
n << 1, 2, 3,
     4, 5, 6,
     7, 8, 9;
cout << "\nMatrix n:\n" << n << endl;
```

- Matrix operations

```
auto m1 = m + n; // This will fail.
auto m2 = Eigen::Matrix2f::Identity();
auto m3 = m + m2; // This will not.
cout << "\nMatrix m3:\n" << m3 << endl;
```

Eigen Tutorial : Final Words

- Quick documentation (for MATLAB users)
  - http://eigen.tuxfamily.org/dox/AsciiQuickReference.txt
- Getting started
  - http://eigen.tuxfamily.org/dox/GettingStarted.html
- Longer tutorial
  - http://eigen.tuxfamily.org/dox/group__TutorialMatrixClass.html
- Reference
  - http://eigen.tuxfamily.org/dox/group__QuickRefPage.html

Building on Linux

- Download the zip for dgp17-exercise1 on Moodle
- Download g++, make and cmake from your favourite packet manager
- You can then compile the lab using a terminal:

```
~$ cd /path/to/dgp2017-exercise1/
~$ mkdir build/
~$ cd build/
~$ cmake ..
~$ make
~$ ./lab0_eigen/lab0_eigen
```

(see slide on Qt Creator if you need an IDE for C++)

Building on a Mac

- Install Xcode from the AppStore
- Install brew (see http://brew.sh/)
- Using brew install cmake
- You can then compile the lab using a terminal:

```
~$ cd /path/to/dgp2017-exercise1/
~$ mkdir build/
~$ cd build/
~$ cmake ..
~$ make
~$ ./lab0_eigen/lab0_eigen
```

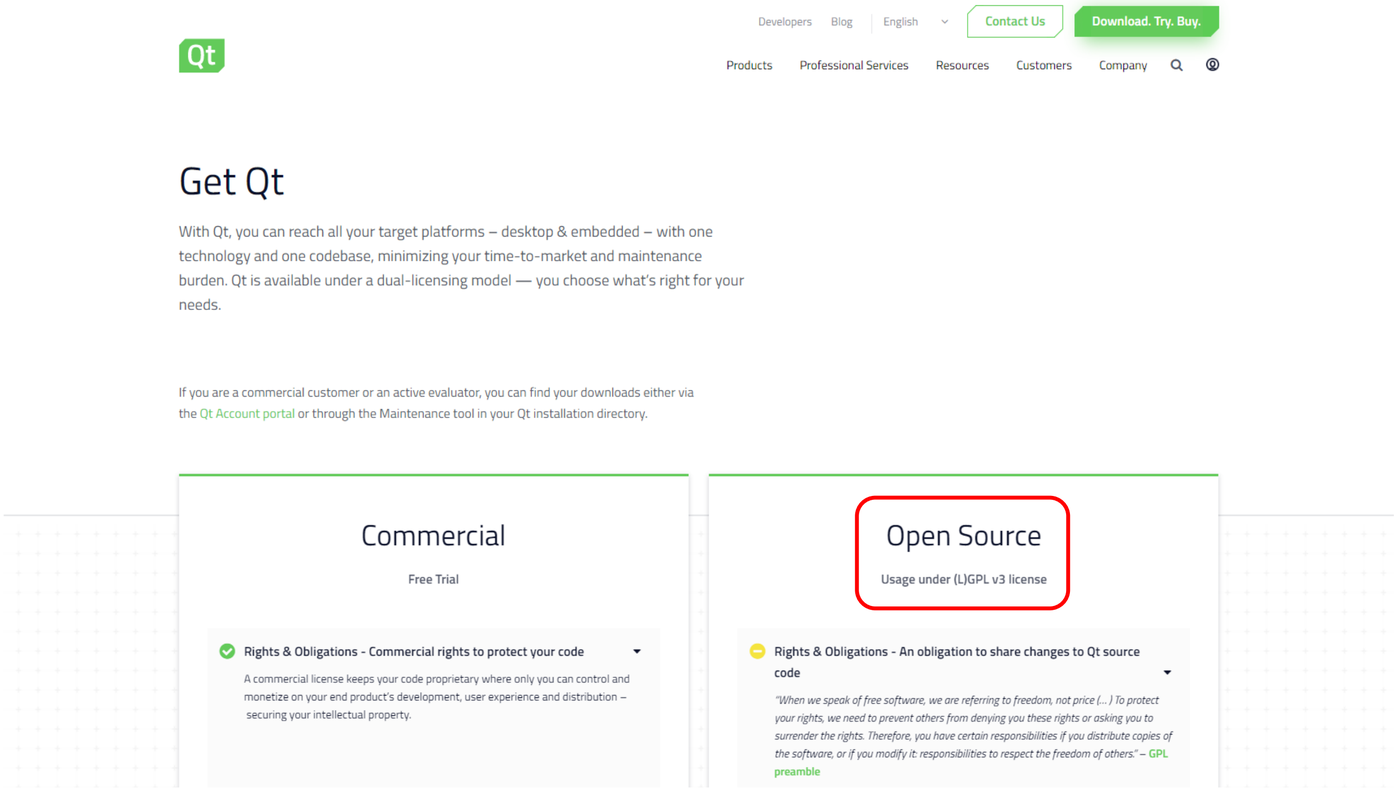
(see slide on Qt Creator if you need an IDE for C++)

Building on Windows

- Download and install Visual Studio Community Edition from https://www.visualstudio.com/vs/community/
  do not forget to tick Visual C++ when during installation
- Download and install CMake from http://www.cmake.org
- See next slide to build and run your program
  (we recommend you to use Qt Creator instead of the Visual Studio IDE has it has a better CMake integration)

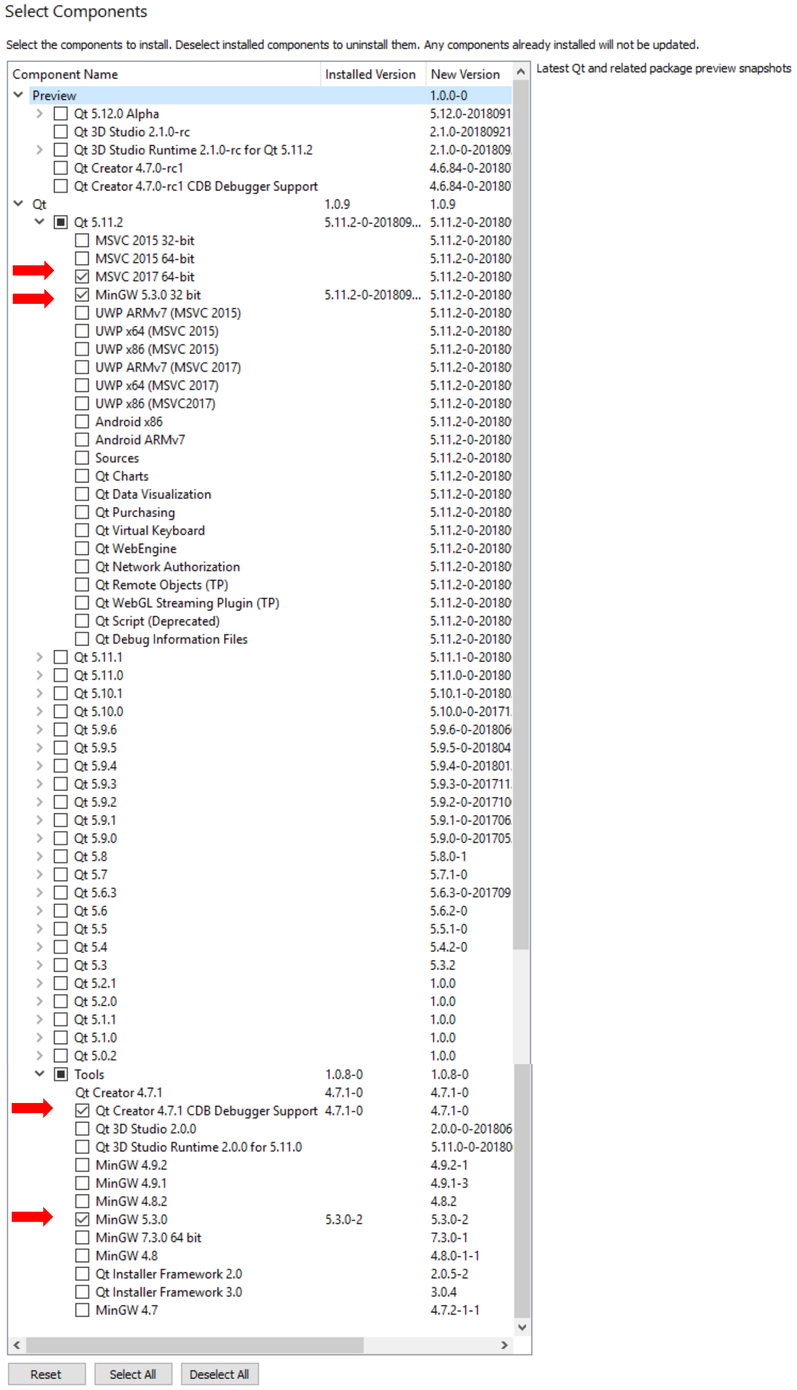
QT Creator (1/6)

- Download and install QT Creator (only, not the full Qt stack) from https://www.qt.io/download-open-source/#section-9
- Choose “Open Source”

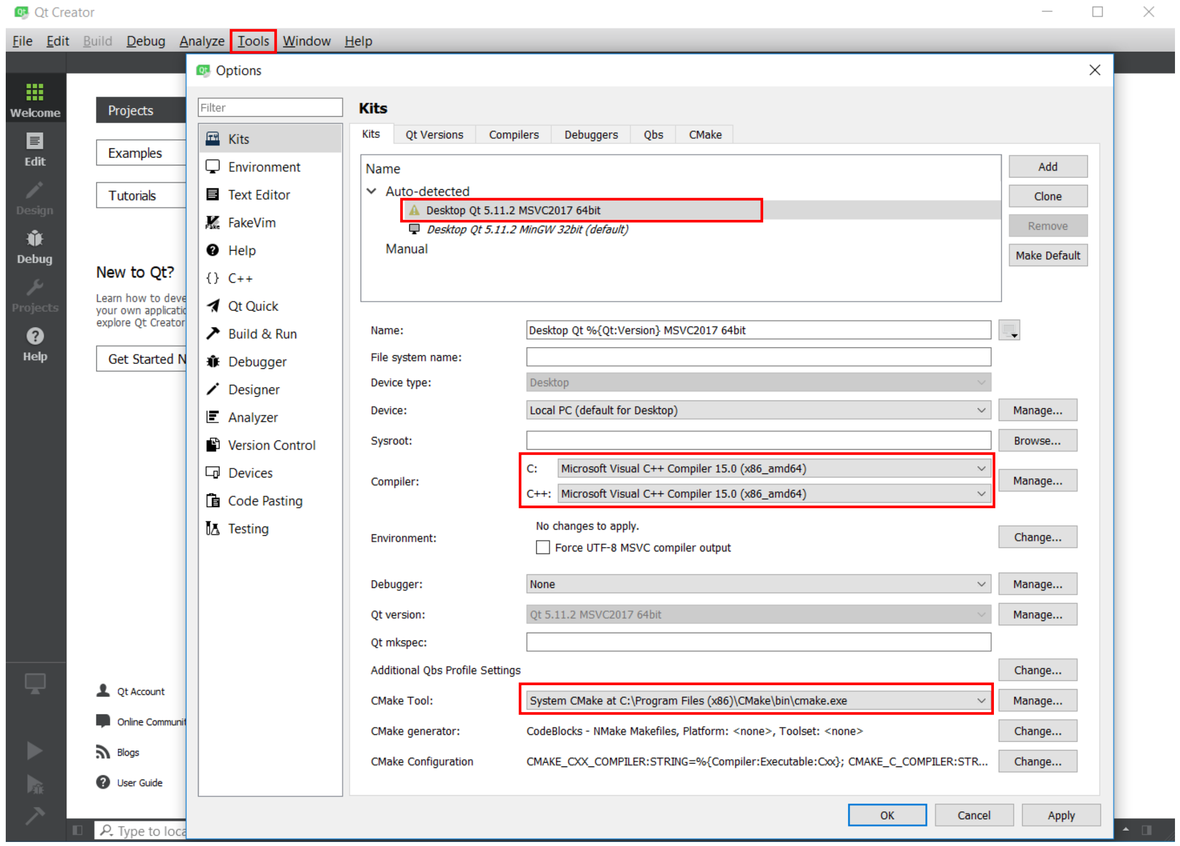


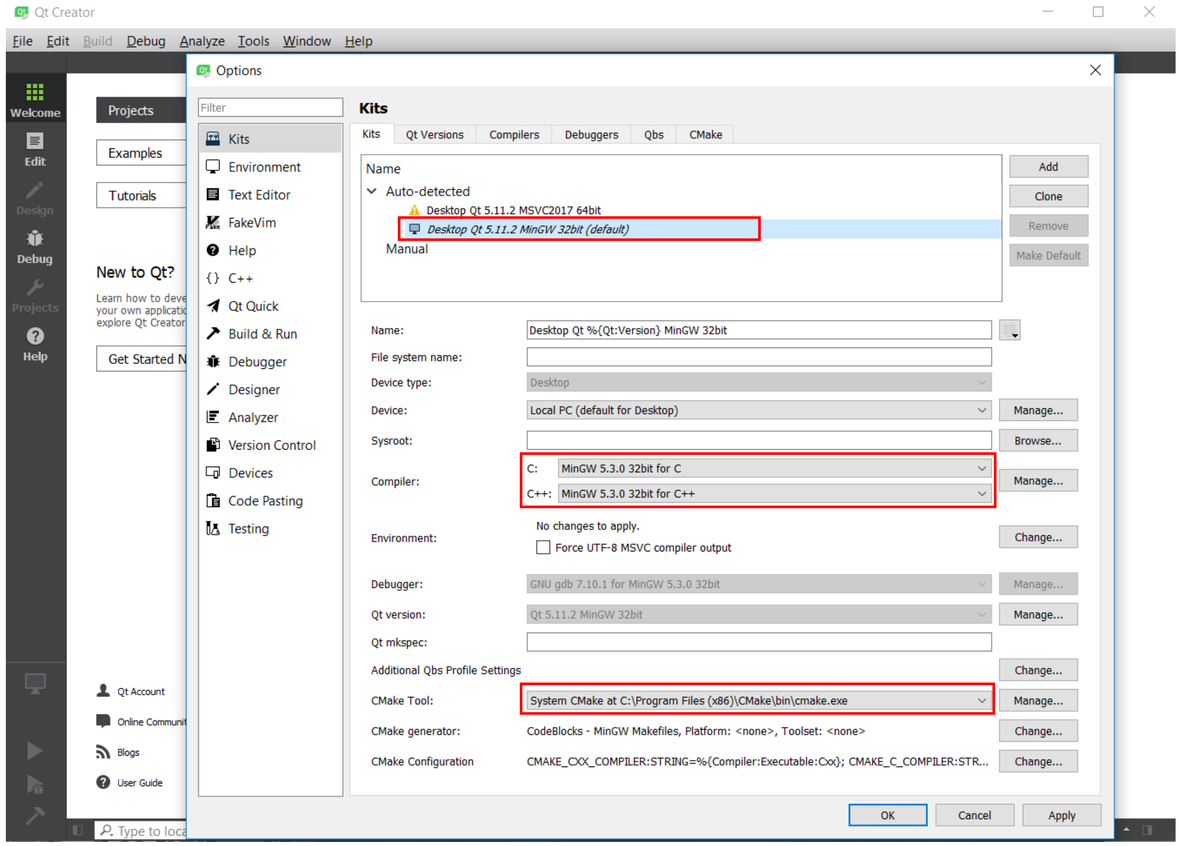
QT Creator (2/6)

- Select the following components to install
  - ❑ QT 5.11.2
    - ✓ MSVC 2017 64-bit
    - ✓ MinGW 5.3.0 32 bit
  - ❑ Tools
    - ✓ Qt Creator 4.7.1 CDB Debugger Support 4.7.1-0
    - ✓ MinGW 5.3.0
- Launch Qt Creator and follow the screenshots on the next slides



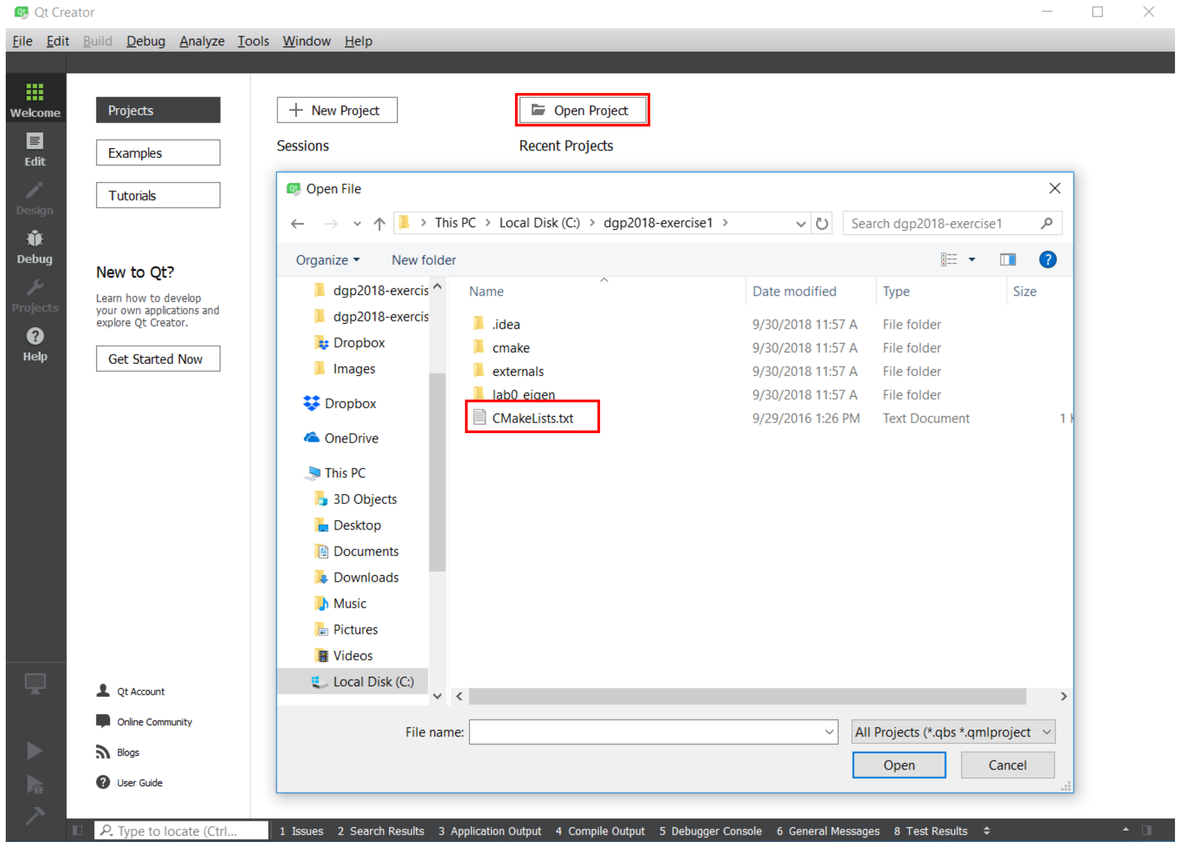
QT Creator (3/6)



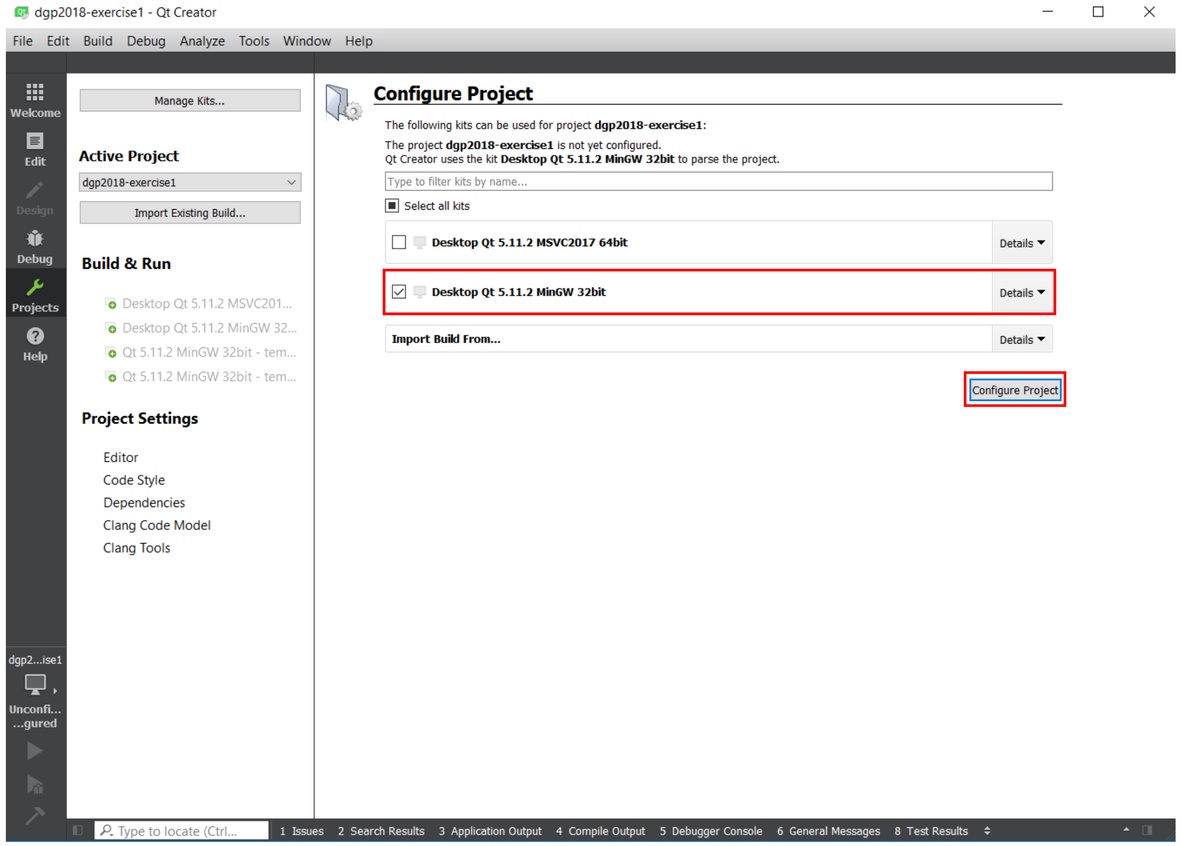


On ***Windows only***, be sure that your compiler is set to be the Microsoft Visual C++Compiler or MinGW (click “Tools” -> “Options”)

QT Creator (4/6)

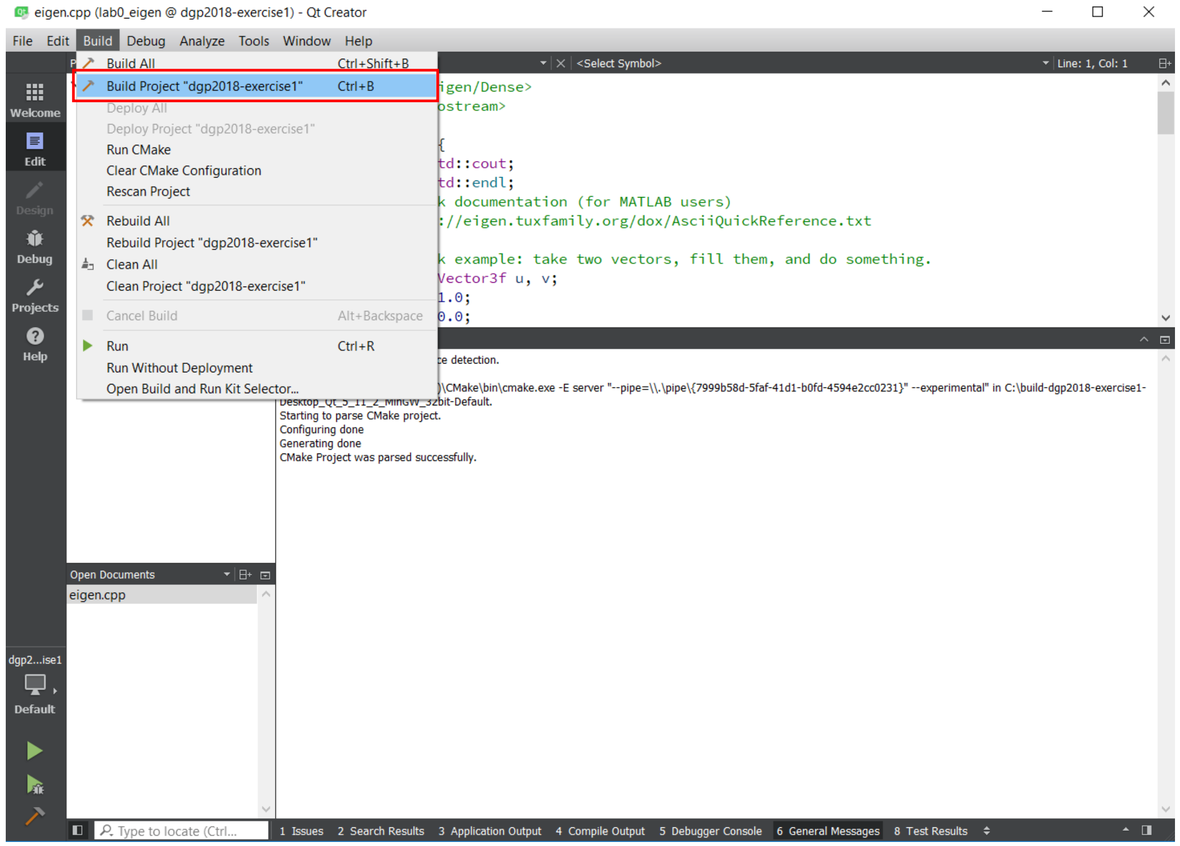


Click on “Open Project” and select the CMakeLists.txt file at the root of the dgp2018-exercise1/ directory

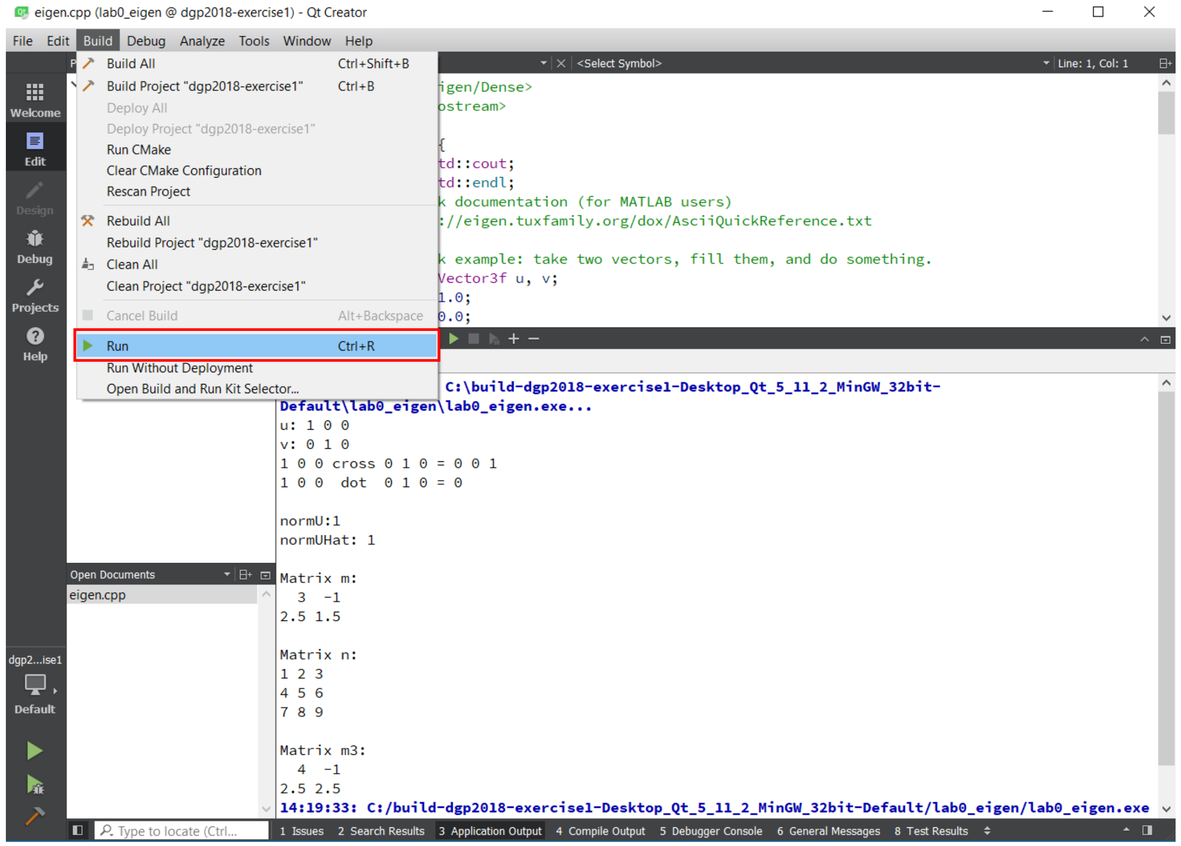


Select “Desktop QT 5.11.2 MinGW 32 bit” (or “Desktop QT 5.11.2 MSVC 2017 64bit”) and click on “Configure project”

QT Creator (5/6)



Click on “Build dgp2018-exercise1”



Click on “Run” and see the output of the program

QT Creator (6/6)

- You can also use QT Creator on a Mac
- When install QT creator on a Mac, you only need to select the following component

❑ QT 5.11.2

✓ macOS