**How to Compile for Dummies**

**CPU Scheduler**

Version 1.0

This program primarily focuses on the basic API’s provided, with this in mind – it is possible to run this program across platforms. Though in-order to run on the desired platform, the program must be compiled *in* that desired platform. In this document, I will provide instructions to compile this program in both *GNU/Linux* and *Microsoft Windows*.

**Microsoft Windows**

Microsoft Visual Studio 2017

Despite that *Visual Studio 2017* is covered in this document, it should be understood that any compliant C/C++ compiler and IDE’s for *Microsoft Windows* based Operating Systems will just fine.

|  |  |
| --- | --- |
| **Step** | **Task** |
| #1 | Open Visual Studio 2017 |
| #2 | Create a new project from the file menu |
| #3 | With Visual C++ templates installed, select ‘Empty Project’ from the ‘General’ templates. |
| #4 | Fill in the Project properties to your liking or you can use the following:  Name: CPU Scheduler  Location: %UserProfile%\Desktop\  Solution: Create a new solution  Solution Name: CPU Scheduler  Then click ‘OK’ when ready |
| #5 | Once solution has been created, import the ‘main.c’ file into the project. To do this, perform the following:   * Open ‘Solution Explorer’ * Right-click on ‘Source Files’ * Add > Existing Item… * Navigate to the ‘main.c’ file (our CPU Scheduler) * Click ‘Add’ once the file has been selected |
| #6 | Due to ‘unsafe’ functions being used, we must change the preprocessor for this project. To do this, perform the following:   * Open ‘Solution Explorer’ * Right-click on the solution name (or CPU Scheduler solution name) * Left-click on Properties * From the tree on the left side, be sure to locate and select the following: Configuration Properties >> C/C++ >> Preprocessor |
| #7 | With the Preprocessor selected, find ‘Preprocessor Definitions’ and then include the two flags:   * \_CRT\_SECURE\_NO\_DEPRECATE\_ * \_CRT\_NONSTDC\_NO\_DEPRECATE\_   Without these flags, Visual Studio’s compiler will refuse to compile this project due to unsafe thread functions – despite we are not using threads in this program.  Click ‘OK’ to add the flags into the Preprocessor Definitions. |
| #8 | To compile and run the program, merely click on the green play button. |
| #9 | To generate a release build, merely expand the compiler settings and change the compiler setting from ‘Debug’ to ‘Release’. |
| #10 | Run the program! |

**GNU/Linux**

Makefile

There exists an easy script to help compile this program easily with minimal interaction needed, just be sure that the *makefile* that was provided in this program - is available to you. However, if not – I will also include the steps to compile the program without the makefile.

**DEPENDENCY NOTE:** Please insure that the following dependencies has already been installed on the host system:

* make
* cc

|  |  |
| --- | --- |
| **Step** | **Task** |
| #1 | Download or duplicate the program files anywhere in your filesystem. |
| #2 | Open the terminal |
| #3 | Navigate to the project files that was downloaded or duplicated to your host system; for example: cd ~/desktop |
| #4 | With being in the same working directory, first assure that the make and cc dependencies has been installed. |
| #5 | Compile the program by typing the following the command: make |
| #6 | To run the program, merely type: UltraMega\_Scheduler in the terminal. |

**GNU/Linux**

Without Makefile

If you already installed the program using the *makefile*, then ignore this section. This section is dedicated to compiling the program if incase the *makefile* either was not included or does not simply work as intended.

**DEPENDENCY NOTE:** Please insure that the following dependencies has already been installed on the host system:

* cc

|  |  |
| --- | --- |
| **Step** | **Task** |
| #1 | Download or duplicate the program files anywhere in your filesystem. |
| #2 | Open the terminal |
| #3 | Navigate to the project files that was downloaded or duplicated to your host system; for example: cd ~/desktop |
| #4 | With being in the same working directory, first assure that the cc dependencies has been installed. |
| #5 | Compile the program by typing the following the command: cc main.c -w |
| #6 | To run the program, merely type: a.out in the terminal. |