Program:	**GDP1**
Course:	**INFO6025 – Configuration and Deployment – Fall 2023**
Professor:	**James Lucas**
Project # 1:	**Installer basics!**
Weight:	**25% of "project" mark**
	(or an even weight, given less or more than 4 total projects)
Due Date:	**Thursday, December 7, 10:00pm**

Note: This project must be done independently. No group submissions are allowed.

Description and Purpose

This project is meant demonstrate your ability to create a basic installer program using NSIS.

- Create a c++ console application called **hello.exe**
  - When run, it will read then output the contents of a file **hello.txt** to the console, then wait for any user input to close.
- Create an NSIS installer script **installer_script.nsi**
  - When compiled by NSIS, it will create an installer **hello_installer.exe**.
  - The installer will create a **Hello** folder on the desktop containing **hello.exe** and **hello.txt**.
  - The installer will write your student number to a registry location (specified below)
  - The installer will also create an uninstaller **hello_unsinstaller.exe**.
  - The uninstaller will clean up everything including itself – deleting the installer, deleting the hello folder (and contents), and deleting the registry addition.

Your Submission

- **Submit a single .zip containing the script file, your hello solution, and your hello.txt file.**
- No readme file or video is required for this project.

"Show-Stopper" Marks

While these aren't "worth" any marks, they could deliver large penalties, or in some cases a mark of zero.

- Does not compile (mark=0)
- Build fails, doesn't run, or runs with crashes/errors (I may investigate for a simple fix to something like a last-minute typo, but if it's not a super simple fix for me then… mark=0)
- No/Awful/Nonsensical documentation (-30%)
- No/Terrible/Painful to look at conventions/style (-50%)

Marking Scheme

This is how the marks are divvied up.

Item	Marks
Hello.exe Your hello solution compiles to create **hello.exe**, which reads a **hello.txt** file and outputs the contents to console, waits for any user input, then closes.	1
Hello_Installer.nsi (Installer) **Hello_Installer.nsi** creates **hello_installer.exe**, which: • Creates a **Hello** folder on the desktop. • Adds **hello.exe** to the **Hello** desktop folder. • Adds **hello.txt** to the **Hello** desktop folder. • There is an "AfterBuild" step that copies hello.txt and Hello_Installer.nsi to the output folder. Ie you will end up with Release/ hello.exe hello.txt Hello_Installer.nsi	10
Hello_Installer.nsi (unnstaller) **Hello_Installer.nsi** also creates **hello_uninstaller.exe**, which cleans up everything, including itself.	10
Harder Marks 1 • There is a Function call in your script that checks for a pre-existing installation of Hello, the user may choose to overwrite, or cancel the installation.	4
Harder Marks 2 • The installer writes your student # as a string to the registry: **Location:** SOFTWARE\Wow6432Node\Configuration2023\Project4 **Name:** StudentNumber • The uninstaller cleans up the above registry value and location.	10
TOTAL	35

Additional Requirements

- While you may freely "borrow" mine (or anyone other) code ***but*** your code should be "sufficiently" different from mine. See the "plagiarism" test, later in this document, for more details.
- Further, you *cannot* simply use an existing game engine (or part of a game engine), even if it's a "from source" engine (i.e. you have the entire source) to complete this assignment; it should be either completely new of significantly modified. This includes, but is *not* limited to: Unity, Unreal, Cry, Anarchy, XNA, Cocos, Ogre, the framework from the OpenGL text, etc. In other words, you are expected to have made the vast majority (essentially all) of the engine *in this term by yourselves, from "scratch"* - i.e. starting from something a rudimentary as the "OpenGL Book" code or the GLFW starter code (we started with that in class).
- You also may not use any "3rd party" physics code/libraries like bullet, havok, etc. The exception to this is code taken from the text book, of course, but you also can't just use Ian Millington's complete "cyclone" engine code – you can use code inspired from that, but it has to be almost completely "yours" and created this term.
- The most "engine" type code you can use is limited to GLFW, glad, and OpenGL Math (glm); anything more is almost certainly "too much engine" code.

75/10-year old "squinty eye" plagiarism test:

(Credit: Feeney)

I have very little tolerance for plagiarism, but many students are unclear about what it is.

Basically, it's submitting somebody else's work as your own.

There is sometimes some confusion over this because you could argue nothing is actually "unique" (see: http://everythingisaremix.info/ for a fascinating overview of this).

The whole point of assignments/tests/projects in this course (or any course, really) is to try to see if you are actually able to ***do*** the coding that's asked of you. In other words: How competent are you? Handing me someone else's code and/or making a trivial change isn't good enough.

Also, it's illegal:

- http://www.plagiarism.org/ask-the-experts/faq/
- http://definitions.uslegal.com/p/plagiarism/
- http://en.wikipedia.org/wiki/Plagiarism

- https://www.legalzoom.com/articles/plagiarism-what-is-it-exactly

In other words, I'm not going to be drawn into a giant debate over how "different" your code is from mine or anyone else's, if any sensible person (including me) would conclude that the code/application is pretty much the same thing, then it is. It is up to my discretion to decide this.

- While you may freely "borrow" mine (or anyone other) code ***but*** your code should be "sufficiently" different from mine (you might want to replace the word "sufficiently" with "significantly").
- In other words, you *cannot* simply use an existing game engine (or part of a game engine) to complete this assignment; it should be either completely new of **significantly** modified.
- How will I determine this?
    - If I showed your application and/or your source code to either a pragmatic 75-year-old mother, or a typical 10-year-old, or even some random person walking down the hallway (i.e. a non-expert), and they looked at it, tilted their heads, squinted their eyes, and said "you know, they look the same," then they ***are*** the same.
    - Another test would: How much time it would take for a competent programmer (for example, me) to make the changes you are submitting? The point here is that I don't "care" if you tell me "But it took me *weeks* to make the changes!" Fine, but if I can make those same changes in 10 minutes, then not a lot of work has been done (certainly **not** sufficient work – these projects should show take **days** of work having been done).