CSE 321
Project 1: Foundations Of Design and Implementation

Purpose

The project aims for you to set up the base elements that will be used in later projects.

You will be applying the following concepts: planning methods.

You will be utilizing MBed Studio, creating planning documents, reading documentation, practicing best practices with code and creating the necessary templates for future project elements.

It is understandable that you may not have your Nucleo yet, if you want to run the code, you can borrow one during recitation and try it out in the lab.

Experience Mapping

NACE Competency	Included in Assignment
Career and Self Development	
Leadership	
Communication	X
Professionalism	X
Critical Thinking	x
Teamwork	

Learning Objective	Included in Assignment
1. Characteristics	
2. Programming	X
3. Documentation	X
4. Use of real-time elements	
5. Security	
6. Design and implementation of systems	

Equity and Inclusion	
Technology	X

Policy reminders

- **Academic Integrity**
- **Collaborating is not allowed**
  - **Other than during the facilitated part of the midpoint submission**
- **Support is available through recitation sections, digital communication, office hours, and appointments. It is your responsibility to use those resources appropriately and communicate with your instructor if there are issues.**

Tasks

Overall while completing this project you will complete the following tasks:

- Create a repository to navigate resources - submit as part of opening plan
- Setup MBed Studio - submit as part of your midpoint assignment
- Create your template for programming files - submit as part of your midpoint assignment
- Develop an algorithmic plan on how to approach going from a problem through complete implementation - submit as part of your midpoint assignment
- “Fix my mess”
- Create a wrap up document to summarize what you implemented

Submission Requirements

Preliminary Submission Requirements:

- PDF of documentation repository created
- Link information to repository location
  - With permissions for course staff to access only, not public

Midpoint Submission Requirements:

Part 1 Submission requirements

- PDF of screenshot of MBed being set up
- Code Template

- PDF of the algorithm you developed to figure out how to approach problems
- Docx of the evaluation guide

Part 2 Submission Requirements

- Notes on the review of someone else's Code Template and Algorithm
    - PDF of the algorithm annotated
    - Code file with feedback added
    - Completion of the evaluation document

Final Submission

- Revised Algorithm, Template and "Fix my mess" documents
- Summary document
- Reflection

Preliminary Task - Opening Plan:

Navigating materials and documentation can be one of the most challenging tasks. To help face these challenges you will be constructing a reference index document that you will use to help you in the future.

This document needs to be constructed in a way that you can maintain as a live document as the semester progresses as more resources are needed. This document will need to include keywords, brief descriptions, and page references for key elements.

It is your choice where to host the repository as long as it is not publicly findable. For Google Docs, this would be the setting that only people with the link can view. You can also use Microsoft365, GitHub (permissions will need to be given to course staff to see it and it needs to stay private until the end of the term), or as a webpage on a site you manage (with the page not accessible without the actual link).

You will be downloading a copy of your repository as a PDF to upload for submission along with the link to the live repository for the submission.

Setup MBed Studio

We will be using MBed Studio as the IDE to support the development of the programs in collaboration with MBed OS when OS elements are integrated. You will need to demonstrate you are able to create a program in this environment.

MBed Studio is installed on all of the lab machines and you will have access to them during recitation and TA office hours.

There are some versions of Linux that are not compatible, if you have one of those the lab machines will be where you need to demonstrate you are set up. While working on the project outside of the lab space, there are options and you should discuss those with Dr. Winikus to reduce technical issues that can not be addressed.

Note - To make things more efficient on your machine… you want to set up one project and then use the option to share one set of the OS.

Code Template

Communication within your code is critical to legacy utilization and collaboration. A way to support this is to create a template that has comments in it to represent the necessary information to make the code easy to navigate and understand.

The template will have a header, a section denoted for files to include, a section for global variables, a section for the main function, and a section defined to place functions you create. These all will be put into the file as comments and saved as a .cpp file to submit.

We will be using the style guide that goes with Mbed OS, the OS we will be using. https://os.mbed.com/docs/mbed-os/v6.15/contributing/style.html This is based on the Kerninghan and Ritchie core style established in “The C Programming Language", the first edition is on Archive.org for your convenience tp use in addition to what is on the Mbed website.
https://archive.org/details/TheCProgrammingLanguageFirstEdition/page/n79/mode/2up

Dr. Germain’s commenting guide will come in handy as a reference
https://www.cs.utah.edu/~germain/PPS/Topics/commenting.html

Review Dr. Germain's site and some examples of code on the Mbed website to establish the content that should be included in each comment block that is going into the template.

Best practices in programming establish that **all files have a commented segment at the top** that provides information about the file. This should contain the following information for the reader:

- Who wrote the code
- The file's purpose
- Any modules/subroutines
- If it corresponds to any assignments
- The inputs
    - *Note: for the main file this will be for the peripheral connections. Function-specific inputs should be noted in the comments with the function.*
- The outputs
    - *Note: for the main file this will be for the peripheral connections. Function-specific outputs should be noted in the comments with the function.*
- Any constraints
- Any sources or references used (**with links**)

Algorithm Plan to Approach Problems

An algorithm is a set of instructions to follow to accomplish a task. Your task is how to take a word problem and go through the steps to implement it. The steps include decomposing it into the elements that need to be developed, how the elements interact with each other, constraints, testing. Each of these steps have multiple steps needed to accomplish the goal associated with it.

For Projects 2 and 3 you will be expected to document many of the parts expected during the process of implementing, so your plan will need to reflect those key stages (and the appropriate steps in each stage). The required stages/elements that need to be included are the defining of the objective/specifications, the creation of a flowchart/visual to define the behavior, schematic, testing, analysis.

Create a numbered list to demonstrate the order of how you plan to approach solving a problem. Make a PDF with this list and submit during Part 1 of the Midpoint.

Guide to Evaluating

In order for things to be reviewed, it has to have guidance to find the required elements. You will complete the provided table. This needs to be submitted as a docx so that the reviewer can add to your document.

Provided to you	To be completed by you	To be completed by the reviewer		
Required Project Element	Where to look in the submission	Is the element found based on the instructions	If the element is found, does it meet the requirements for that element?	Explain
Screenshot of MBed Studio Setup				
Header in the template file				
Scaffold in the template file				
Complete Algorithm to accomplish implementation				
Algorithm methodology validity				

Peer Review

To consider different viewpoints and approaches, reviewing the work of others can be very helpful in understanding concepts further.

You will be given a submission from part 1 created by another student. There are two elements you will do in the review: 1. The implementation elements like spelling; 2. The logic elements are present and correct.

For the implementation part, you will annotate directly on the provided files. Meaning you will type in the file. To make this clear, since we aren't using the file to execute a program, type your comments in as plain text, do not type them as comments. For the algorithm, you will use the comment feature in a PDF program to add on to the PDF to provide feedback.

For the logic of the work you are reviewing, you will use the provided table from the student for feedback.

You will submit these 3 files as your part 2 submission.

"Fix My Mess"

Legacy code and collaborations on projects involving code are common, but the other individuals may not be actively working on the project or code with you. Sometimes, they might not even be at the same company anymore. A good exercise for understanding the importance of good communication in programming is fixing a disastrous bit of code with a poorly written README file.

To process through and rectify horrible code is a challenge. As a way of practicing this and considering the elements in the code for use with the Nucleo, you will take the provided code and clean it up. While cleaning it up, you will also have to complete some blanks related to the functionality of the hardware that is being controlled in the code.

Open the provided code and clean it up. This means you may need to address the following issues.

- Professional Free Field layout
- Create an appropriate header
- Add comments
    - Any line that is not a very clear self-commenting operation must have a comment
    - Fix comments
- Make variable names appropriate for a professional setting

    - Choose self-commenting names if possible
- Remove unnecessary and inappropriate elements

Some additional details on comments for when you clean things up

- Some comments are correct
- Some comments need updating/fill in the blank
- Some are badly worded
- Some are totally inappropriate
- Some tell you what to do to the code to make the appropriate correction
- Some do not change the content of! The line numbers are based on the initial provided code. You can rearrange them as it makes sense.

The program must still run when you are done with the same behavior it had when you started.

Save this file as **CSE321_project1_*username_corrected_code.cpp***

README files are critical to collaboration, deployment, and continuous improvement. There are many variations on the content, but it needs to be appropriate for the project. For this assignment you will be doing a plain text README rather than the markdown style which is often seen (you will do markdown in the future). You can reference some tips on README contents and formatting from the University of Oklahoma and this guide from Akash Nimare.

Update the README file. This means you may need to address the following issues.

- Professional Free Field layout
- Overall grammar and language
- Relevant and correct content
- Clear and correct specifications
- Correct constraints, parameters, and names

This program makes use of some elements (specifically API functions) that have not been covered yet. These are tools with specific purposes. You will need the information here to create appropriate comments in the code and the **readme.md**

- `Thread`
    - Allows things to be configured with scheduling and priority management. It is a block of code that will execute based on an event.

    - There are methods to access information with this controller and to control it. Specifically in this project you will see: `get_state` and `start`
- `DigitalOut`
    - Allows you to set up a GPIO pin as an output, doing all that initialization stuff.
- `InterruptIn`
    - This creates a reference to an interrupt with a variable name to allow additional interaction. This interrupt is triggered by the button, with actions on both rise and fall of the signal.
    - There are methods to configure the interrupt, specifically: `rise` and `fall`.

Do not change these terms/lines in the code. You can change other elements that are adjacent to them if appropriate (like the names of variables and functions). You can rearrange and move them still as appropriate.

```
#include "mbed.h"
Thread
DigitalOut
InterruptIn
.start
.rise
.fall
.get_state
thread_sleep_for(500)
thread_sleep_for(2000)
printf
```

Save this updated readme as **CSE321_project1_*username_readme.md***

Push/store these files to your repository and submit them on UB Learns with your final submission.

Wrap Up

Make the appropriate changes from the review before your final submission.

As part of completing the project, in addition to demonstrating you will need to communicate your work in a report. This sharing of technical design is good practice for future careers. For practice in conciseness, you will be building your report into a slide deck. The slide deck forces focus and the note field requires explanations.

The “report” will need to contain the following elements:

- Introduction presentation and discussion
- Goal/Purpose presentation and discussion
- Design presentation and discussion
- Implementation discussion
- Reflection

Each section and the overall report will consider general communication standards. This quality would be applied as a penalty to the appropriate portion of the report (if it is only an issue in section X, then only section X gets a penalty, but if it appears in multiple sections then the penalty is for the overall report).

Code Review in Final Submission

Due to the nature of careers, collaboration and legacy elements are common, some of the requirements for the project can only be evaluated with the review of your code.

The code template and “fixed” codes will manually be reviewed to make sure all the elements are included, that commenting is appropriate, the overall methods of implementation, and the overall appearance of the code.

The documents evaluated with comparison to best practices and requests.

Reflection

Then you will need to complete a reflection to help you be more successful in the remaining parts of the project. You will enter your answers in UB Learns.

1. Based on your feedback, what improvements are needed/suggested based on the table and from the recommendations paragraph? Identify those improvements and explain why you have those recommendations.

2. Based on your review, are there elements that you could consider improving after seeing the work of your peer? If there is nothing learned to help you improve things, then discuss how your peer's work validates your work.
3. Based on your answers to questions 1 and 2, what changes did you plan to make? Explain how you approached making those changes.
4. When reviewing code to clean it up, what approach/method did you take?
5. Based on your experience, what value does the quality of commenting, organization and documentation for code have to you?

Grading

The tasks that make up the project have the following contributions to the grade.

- 5% Opening Plan
- 15% Midpoint Check In
  - 7.5% submission of work
  - 7.5% submission of the feedback on work reviewed
- 20% "Fix my mess"
- 30% Wrap up
  - This is a written report containing the following elements
  - 2.5% Introduction presentation and discussion
  - 7.5% Goal/Purpose presentation and discussion
  - 10% Design presentation and discussion
  - 5% Implementation discussion
  - 5% Reflection
- 30% Code Review
  - This is the evaluation of the template and algorithm

While it is important to have the experience working with others, please keep in mind that doing your own work is even more important. As you share your projects when the course is done, they need to represent the work you have done. As a reminder, the collaboration is restricted to reviewing and exchanging feedback. If a choice is made to not follow the expectations of doing your own work, then it means that the choice was made to go through the academic integrity violation process (and no one wants to do that).

Criteria for Success

The project is broken down into multiple tasks to support the process of working with projects. Most of the elements in each task will be evaluated on completion, with the relevance and appropriateness of the response considered. If there is anything irrelevant or inappropriate present in addition to appropriate answers then that section will be subject to a penalty. Correctness-based parts will be evaluated in a similar manner, but partial credit will be possible based on the following rubric criteria. These partial credit elements will be applied to the section or overall depending on the situation, and they can be accumulated if multiple parts have issues.

Rubrics

All rubrics will be scaled to the appropriate point value.

Grading in general will be based on three levels of mastery unless otherwise stated. % in this rubric corresponds to the % of points allocated for that element.

Correctness based:

Beginning	Developed	Accomplished
Significant issues or not done or not correct file format [0%]	Some fields missing or inappropriate [40%]	Appropriate and complete [100%]

This segment is attempt based.

Beginning	Developed	Accomplished
Significant issues or not done or not correct file format [0%]	Most of the required elements seem to be present with a reasonable attmept [40%]	All expected elements appear present and attempted fully. [100%]