Learning Outcome

The final project is a venue for students to achieve the learning outcomes below:

- LO1. Analyze problem requirements by describing input specifications, processes, and target output.
- LO2. Design and implement algorithmic solutions from defined problems and requirements by applying knowledge of computing fundamentals using appropriate data types and constructs including expressions, conditional statements, iterative statements, and structured decomposition.
- LO3. Design, execute and document various classes of test cases and their corresponding results.
- LO4. Determine and apply proper debugging techniques using programming constructs and/or computing tools.
- LO5. Apply simple coding techniques such as inline comments, version documentation, and following coding standards for program readability.
- LO6. Exhibit intellectual honesty, responsibility, and punctuality, conforming to Christian principles.

Students will demonstrate their skills in logic formulation and its implementation in C programming language. They will use assignment statements, expression statements, conditional constructs, and iterative constructs to develop the program for the given scenario. Additionally, they are going to start writing test scripts and testing their own programs.

Secret Potions of the Geffen Witches

Images used in this document are edited from the game Ragnarok X,
and the cartoon series The Last Airbender.

In the town of Geffen, witches from all over the country have gathered to discover the secret behind the Ymir's Avatar Potion, the only potion that can help cultivate the Avatar's powers. The Ymir's Avatar Potion can only be crafted in Geffen, and it requires the following materials: Fire Chakra Potion, Water Chakra Potion, Earth Chakra Potion, and Air Chakra Potion. These potions can also be crafted in Geffen. The materials for each potion are shown in the Table 1. Materials and their locations for crafting each of the required potions for Ymir's Avatar Potion.

Table 1. Materials and their locations for crafting each of the required potions for Ymir's Avatar Potion.

Potion	Materials	Location
 Fire Chakra	Scaleless Blackfish	Taal Lake
	Gold	Holgrehenn Store
	Majestic Water	Holgrehenn Store
	Wondrous Vinegar	Holgrehenn Store
 Water Chakra	Mariana Snailfish	Galathea Deep
	Gold	Holgrehenn Store
	Majestic Water	Holgrehenn Store
	Wondrous Vinegar	Holgrehenn Store
 Earth Chakra	Mudskippers	Dagupan Mangrove Forests
	Gold	Holgrehenn Store
	Majestic Water	Holgrehenn Store
	Wondrous Vinegar	Holgrehenn Store
 Air Chakra	Hillstream Loaches	Mindanao Current
	Gold	Holgrehenn Store
	Majestic Water	Holgrehenn Store
	Wondrous Vinegar	Holgrehenn Store

The secret behind the Chakra potions crafted in Geffen are the extremely rare fishes that can be gathered from the 4 fishing sites: Taal Lake, Galathea Deep, Dagupan Mangrove Forests, and Mindanao Current. As these are extremely rare fishes, the chances of catching them from their respective locations are only **10%**. This chance can be increased to **30%** by using the Magical Bait which can be bought from Holgrehenn Store. The other materials for crafting the Chakra potions can also be bought from Holgrehenn Store. The currency used in Geffen is Ymir gold coins. A list of the items that can be bought from Holgrehenn Store is shown in Table 2.

Table 2. Items that can be bought from Holgrehenn Store, and its corresponding prices.

Item to buy	Price
Gold	750 Ymir
Majestic Water	100 Ymir
Wondrous Vinegar	150 Ymir
Magical Bait	300 Ymir

As you qualify and enter the town of Geffen, you will be given a magical fishing rod. Geffen is a prosperous town, so you don't need to worry about your living expenses. Housing and food are provided for free to all citizens of Geffen. If you want to earn Ymir coins to buy the materials that you need for the potions, you can sell the fishes that you can catch from the fishing sites. Holgrehenn buys these fishes too. Table 3 shows the fishing sites, the corresponding fishes that can be caught from the sites, and their prices when sold to Holgrehenn.

Images used in this document are edited from the game Ragnarok X,
and the cartoon series The Last Airbender.

Table 3. Fishing site locations, and the fishes that can be caught and its corresponding prices when sold.

Location	Item to sell	Price
 Taal Lake	Scaleless Blackfish	1000 Ymir
	Tilapia	35 Ymir
 Galathea Deep	Mariana Snailfish	1000 Ymir
	Sardines	40 Ymir
 Dagupan Mangrove Forests	Mudskippers	1000 Ymir
	Bangus	50 Ymir
 Mindanao Current	Hillstream Loaches	1000 Ymir
	Tuna	45 Ymir

Instructions

1. This project is to be done **individually**.
2. Make sure that you have the necessary tools for programming in C.
3. Understand the mechanics of the **Secret Potions of the Geffen Witches** and formulate the problem.
4. Create a C program for the **Secret Potions of the Geffen Witches**. The player should be able to do the following:
   a. **Select from Main menu** – The main menu must show the list of places the player can travel to. The player should be able to choose where they want to go. They can teleport to 6 places: Geffen, Holgrehenn Store, Taal Lake, Galathea Deep, Dagupan Mangrove Forests, and Mindanao Current. They should also have the option to check their bag and to exit the program.

   b. **Check Bag** – If the player chooses to check their bag, the items they have acquired, and its corresponding quantities should be displayed. An option to go back to the list of places is also displayed.

   c. **Craft Potion** – When the player is in Geffen, they should have the options to craft Water Chakra Potion, Air Chakra Potion, Wind Chakra Potion, Fire Chakra Potion, and Ymir's Avatar Potion. They should also have the option to go back to the list of places. If the player chooses to craft a potion, the materials needed and its location must be displayed, and they should have the options to proceed crafting, and to cancel crafting. If the player chooses to proceed and they have the required materials, the potion must be crafted and automatically placed in the bag. After which, the options for crafting will be shown again. If the player chooses to cancel crafting, they should also go back to the list of potions.

   d. **Catch Fish** – When the player is in Taal Lake, Galathea Deep, Dagupan Mangrove Forests, or Mindanao Current, they should have the option to catch a fish. The fish that they can catch depends on the location as shown in Table 3. The catch rate of the extremely rare fishes (i.e. Scaleless Blackfish, Mariana Snailfish, Mudskippers, and Hillstream Loaches) is **10%**, while the

Images used in this document are edited from the game Ragnarok X,
and the cartoon series The Last Airbender.

normal fishes (i.e. Bangus, Tilapia, Tuna, and Sardines) is **90%**. The player should also have the option to catch a fish with magical bait which increases the catch rate of extremely rare fishes to **30%**, while the normal fishes will be **70%**. (Think carefully about how you can implement the catch rates). When the fish is caught based on the catch rate, it must be automatically placed in the bag. There should also be an option for going back to the list of places.

e. **Buy Items** – When the player is in Holgrehenn Store, they should have the option to buy items, to sell items, and to go back to the list of places. If the player chooses to buy items, the program must display the list of items that can be bought, and its prices as shown in Table 2. Then, the player can specify which item they want to buy one type at a time and the quantity. If the player has enough Ymir coins, the item will be bought and automatically placed in the bag, then the price will be deducted from the current Ymir coins of the player. If the player doesn't have enough Ymir coins, the program should tell them. They may also cancel buying which leads them back to the options in Holgrehenn Store.

f. **Sell Items** - When the player is in Holgrehenn Store, they should have the option to buy items, to sell items, and to go back to the list of places. If the player chooses to sell items, the program must display the list of items that can be sold, and its prices as shown in Table 3. Then, the player can specify which item they want to sell one type at a time and its quantity. If the player has the item with the quantity specified, these will be removed from the bag, then the player will receive the corresponding Ymir coins for the items. They may also cancel selling which leads them back to the options in Holgrehenn Store.

5. Write a test script for the program. Test and debug your program.
6. Prepare for the demonstration of your project.

How to Approach the Machine Project

Step 1: Problem analysis and algorithm formulation

Read the MP Specifications again! Identify clearly what are the required information from the user, what kind of processes are needed, and what will be the output(s) of your program. Clarify with your professor any issues that you might have regarding the machine project.

When you have all the necessary information, identify the necessary functions that you will need to modularize the project. Identify the required data of these functions and what kind of data they will return to the caller. Write your algorithm for each of these modules/functions as well as the algorithm for your main program.

Step 2: Implementation

In this step, you are to translate your algorithm into proper C statements. While implementing, you are to perform the other phases of program planning and design (discussed in the other steps below) together with this step.

Follow the Linux Kernel coding standard.

You may choose to type your program in a text editor or an IDE (i.e. Dev-C IDE) at this point. Note that you are expected to use statements taught in class. You can explore other libraries and functions in C as long as you can clearly explain how these work. You may also use arrays, should these be applicable and you are able to properly justify and explain your implementation using these. For topics not covered, it is left to the student to read ahead, research, and explore by himself.

Note though that you are **NOT ALLOWED** to do the following:

- to declare and use global variables (i.e., variables declared outside any function),

Images used in this document are edited from the game Ragnarok X,
and the cartoon series The Last Airbender.

- to use goto statements (i.e., to jump from code segments to code segments),
- to use the break statement to exit a block other than switch blocks,
- to use the return statement or exit statement to prematurely terminate a loop or function or program,
- to use the exit statement to prematurely terminate a loop or to terminate the function or program, and
- to call the main() function to repeat the process instead of using loops.

It is best that you perform your coding "incrementally." This means:

- dividing the program specification into subproblems, and solving each problem separately according to your algorithm;
- coding the solutions to the subproblems one at a time. Once you're done coding the solution for one subproblem, apply testing and debugging.

Documentation

While coding, you have to include internal documentation in your programs. You are expected to have the following:

- File comments or Introductory comments
- Function comments
- In-line comments

Introductory comments are found at the very beginning of your program before the preprocessor directives. Follow the format shown below. Note that items in between < > should be replaced with the proper information. Items in between *[ ]* are optional, indicate if applicable.

```
/*
     Description: <Describe what this program does briefly>
     Programmed by: <your name here>  <section>
     Last modified: <date when last revision was made>
     Version: <version number>
     [Acknowledgements: <list of sites or borrowed libraries and sources>]
*/

<Preprocessor directives>

<function implementation>

int main()
{
     return 0;
}
```

Function comments precede the function header. These are used to describe what the function does and the intentions of each parameter and what is being returned, if any. If applicable, include pre-conditions as well. Pre-conditions refer to the assumed state of the parameters. Follow the format below when writing function comments:

```
/*     <Description of function>
```

Images used in this document are edited from the game Ragnarok X,
and the cartoon series The Last Airbender.

```
     Precondition: <precondition / assumption>
     @param <name> <purpose>
     @return <description of returned result>
*/
<return type>
<function name> ( <parameter list>)
     :
```

Example:

```
/* This function computes for the area of a triangle
   Precondition: base and height are non-negative values
   @param base is the base measurement of the triangle in cm
   @param height is the height measurement of the triangle in cm
   @return the resulting area of the triangle
*/
float
getAreaTri (float base,
            float height)
{
    ...
}
```

In-Line comments are other comments in major parts of the code. These are expected to explain the purpose or algorithm of groups of related code, esp. for long functions.

Step 3: Testing and Debugging

SUBMIT THE LIST OF TEST CASES YOU HAVE USED.

For each feature of your program, you have to fully test it before moving to the next feature. Sample questions that you should ask yourself are:

1. What should be displayed on the screen after a user input?
2. What would happen if inputs are incorrect? (e.g., values not within the range)
3. Is my program displaying the correct output?
4. Is my program following the correct sequence of events (correct program flow)?
5. Is my program terminating (ending/exiting) correctly? Does it exit when I press the command to quit? Does it exit when the program's goal has been met? Is there an infinite loop?
6. and others...

IMPORTANT POINTS TO REMEMBER:

1. You are required to implement the project using the C language (C99 and NOT C++). Make sure you know how to compile and run in both the IDE (DEV-C++) and the command prompt via
   gcc –Wall <*yourMP.c*> -o <*yourExe.exe*>
2. The implementation will require you to:
   - Create and Use Functions
     Note: Non-use of self-defined functions will merit a grade of 0 for the machine project.
   - Appropriately use conditional statements, loops and other constructs discussed in class (Do not use brute force solution. You are not allowed to use goto label statements, exit statements. You are required to pass parameters to functions and not allowed to declare global or static variables.)
   - Consistently employ coding conventions

Images used in this document are edited from the game Ragnarok X,
and the cartoon series The Last Airbender.

- Include internal documentation (i.e., comments)

3. Deadline for the project is the 7:30AM of January 24, 2022 (Monday) via submission through AnimoSpace. After this time, the submission facility is locked and thus no MP will be accepted anymore and this will result to a 0 for your machine project.
4. The following are the deliverables:

Checklist:

- ☐ Upload in **AnimoSpace** by clicking **Submit Assignment** on Machine Project and adding the following files:
  - ☐ source code*
  - ☐ test script**
- ☐ **email the softcopies of everything as attachments to YOUR own email address on or before the deadline**

Legend:
* Source Code also includes the internal documentation. The **first few lines of the source code** should have the following declaration (in comment) **BEFORE** the introductory comment:

```
/*****************************************************************
This is to certify that this project is my own work, based on my personal efforts in studying and applying
the concepts learned. I have constructed the functions and their respective algorithms
and corresponding code by myself. The program was run, tested, and debugged by my own efforts. I
further certify that I have not copied in part or whole or otherwise plagiarized the work of other students
and/or persons.

<your full name>, DLSU ID# <number>
*****************************************************************/
```

** Test Script should be in a table format, with header as shown below. There should be at least 3 distinct test classes (as indicated in the description) per function. There is no need to test functions which are only for screen design.

Function Name	#	Test Description	Sample Input *(either from the user or passed to the function)*	Expected Result	Actual Result	P/F
checkMaterials	1	The player has no materials at all.	Scaleless Blackfish = 0 Gold = 0 Majestic Water = 0 Wondrous Vinegar = 0	Return False	Returned False	P
	2	The player lacks some of the materials.	Scaleless Blackfish = 1 Gold = 0 Majestic Water = 2 Wondrous Vinegar = 3	Return False	Returned True	F

Images used in this document are edited from the game Ragnarok X, and the cartoon series The Last Airbender.

	3	The player has all the materials.	Scaleless Blackfish = 1 Gold = 3 Majestic Water = 2 Wondrous Vinegar = 1	Return True	Returned True	P
…	…	…	…	…	…	…

5. MP Demo: You will demonstrate your project on a specified schedule during the last weeks of classes. Being unable to show up on time during the demo or being unable to answer convincingly the questions during the demo will merit a grade of 0 for your machine project. The project is initially evaluated via black box testing (i.e., based on output of running program). Thus, if the program does not compile successfully using gcc and execute in the command prompt, a grade of 0 for the project will be incurred. However, a fully working project does not ensure a perfect grade, as the implementation (i.e., correctness and compliance in code) is still checked.
6. Any requirement not fully implemented and instruction not followed will merit deductions.
7. Any topic/s applied on the implementation that is/are **outside the scope of CCPROG1 must be clearly explained** during the demo. Make sure that you understand the topic/s and can explain its use, and how it works. Failure to explain can lead to deductions or even a grade of 0 for the final project.
8. This is an **individual project**. Working in collaboration, asking other people's help, borrowing or copying other people's work or from books or online sources (either in full or in part) are considered as cheating. Cheating is punishable by a grade of 0.0 for CCPROG1 course. Aside from which, a cheating case may be filed with the Discipline Office.
9. Bonus points: You can earn up to **at most 10 points** bonus for additional features that your program may have. Sample features and implementation that may allow you to gain bonus points include: (a) use of data structures substantially, appropriately, and properly, (b) incorporation of weather as additional factor for catch rate. Some of the indicated additional features may require self-study. Also, any additional feature not stated here may be added but should not conflict with whatever instruction was given in the project specifications. Bonus points are given upon the discretion of the teacher, based on the difficulty and applicability of the feature to the program. Note that bonus points can only be credited if all the basic requirements are fully met (i.e., complete and no bugs).

Images used in this document are edited from the game Ragnarok X,
and the cartoon series The Last Airbender.