1 Introduction

During this practical assignment you will be required to implement the Abstract Factory and Prototype design patterns.

- Creating UML diagrams from your classes
- Understanding the Abstract Factory Design Pattern
- Understanding the Prototype Design Pattern
- Understand how polymorphism works with virtual functions in c++

2 Constraints

1. You must complete this assignment individually.
2. You may ask the Teaching Assistants for help but they will not be allowed to give you the solutions.

3 Submission Instructions

You are required to upload all your source files and your exported class diagram as a single tar.gz archive to the CS website before the deadline. You are required to implement all makefiles, headers and source files yourself. You should create a Main.cpp to test your code.

4 Mark Allocation

<table>
<thead>
<tr>
<th>Task</th>
<th>Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Unit Hierarchy</td>
<td>18</td>
</tr>
<tr>
<td>Abstract Factory</td>
<td>13</td>
</tr>
<tr>
<td>Prototype</td>
<td>12</td>
</tr>
<tr>
<td>Main</td>
<td>10</td>
</tr>
<tr>
<td>UML Diagram</td>
<td>12</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>65</strong></td>
</tr>
</tbody>
</table>
5 Assignment Instructions

You are making a game and based on your newfound Design-Pattern knowledge you have decided to use the Abstract Factory and Prototype design patterns to build your army.

Your army will consist of Archers and Swordsmen of Common, Skilled and Elite varieties. You should create one instance of a specific unit using the abstract factory and clone it to populate the rest of your army.

This practical consists of five tasks that build on each other:

1. The unit hierarchy
2. The abstract factory pattern
3. The prototype design pattern
4. A Main function to illustrate the use of the design patterns
5. UML Class diagrams

Task 1: The Unit Hierarchy ........................................ (18 marks)
In this task you will create the classes for all your Units.

- An abstract Unit Class.
- Archer and Swordsman classes that inherit from Unit.
- Classes for the concrete units: CommonArcher, SkilledArcher, EliteArcher, CommonSwordsman, SkilledSwordsman and EliteSwordsman.

1.1 The Unit Class
The Unit class should have at least the following members:

Private:
- skill - a string eg. Common
- damage - an integer
- type - a string eg. Archer
- id - initialize this based on a static counter in the Unit class that is incremented with each call to the constructor.

Public:
- A constructor that takes 3 arguments to initialize the skill, damage and type.
- A getDescription function that returns a string describing the Unit. This string should contain all the member variables.

1.2 The Archer
The Archer Inherits from Unit.
- It should have a private member variable ‘range’ (an integer)
- The constructor should take 3 arguments: skill, damage and range
- You should override the description function to include the range. Your overridden function should include a call to the parent class description function.

Hint: Make sure the Archer’s description function is called even when you store one of your archers in a Unit*

1.3 The Swordsman
Similarly to the Archer, the Swordsman should inherit from Unit.
- It should have a private member variable ‘armor’ (an integer)
The constructor should take 3 arguments: skill, damage and armor.

As in the Archer class, you should override the description function.

### 1.4 The concrete Units

The CommonArcher, SkilledArcher, EliteArcher, CommonSwordsman, SkilledSwordsman and EliteSwordsman classes should each have only a constructor. This constructor takes no arguments and calls the constructor of the base class with the correct values. You can choose appropriate values for the damage, armor and range of the different units.

#### Task 2: Abstract Factory

In this task you will need to use the abstract factory design pattern to create your units. You will need the following classes:

- An abstract UnitFactory class with functions to create an archer and a melee unit.
- Concrete factories for Common, Skilled and Elite Units.

#### Task 3: Prototype

In this task you will modify your unit classes to use the Prototype design pattern.

- Add a pure virtual clone function to the unit class.
- Implement the clone function in Archer and Swordsman. You are allowed to create additional helper functions if you want to.
- **Important:** The clone functions should copy all member variables including the id.

#### Task 4: Main

Create a main function that can create your army based on user input for type, skill level and amount. It should illustrate the usage of both the abstract factory and the Prototype design patterns.

In your main you should store your army in a `Unit**`. All dynamic memory should be deallocated so that your program does not cause memory leaks.

**Hint:** Place output statement in your factory, constructor and/or clone function to clearly show when units are created vs. cloned and output a Unit’s details when you add it to your army.

An output example is included at the end of the document. You do not need to follow it exactly.

#### Task 5: UML Diagram

Create a complete class diagram from all the classes of this practical. Annotate the diagram with the roles the different classes have in the Abstract Factory and Prototype design patterns. Export the class diagram to an png image and upload it with your code.

**Software:** You can use Visual Paradigm to create your class diagrams. The license details will be made available on the COS121 discussion board, but you can also use the Community Edition.
Choose Type: Archer=1, Swordsman=2 > 1
Choose Skill: Common=1, Skilled=2, Elite=3 > 3
Enter size of army > 4
Creating -- Unit #1: Elite Archer, damage: 75, Range: 20
Cloning -- Unit #1: Elite Archer, damage: 75, Range: 20
Cloning -- Unit #1: Elite Archer, damage: 75, Range: 20
Enter 1 to create a new army or 0 to stop > 1

Choose Type: Archer=1, Swordsman=2 > 2
Choose Skill: Common=1, Skilled=2, Elite=3 > 3
Enter size of army > 6
Creating -- Unit #2: Elite Swordsman, damage: 80 Armor: 15
Cloning -- Unit #2: Elite Swordsman, damage: 80 Armor: 15
Cloning -- Unit #2: Elite Swordsman, damage: 80 Armor: 15
Cloning -- Unit #2: Elite Swordsman, damage: 80 Armor: 15
Cloning -- Unit #2: Elite Swordsman, damage: 80 Armor: 15
Enter 1 to create a new army or 0 to stop >