Assessment 1

Inheritance and Polymorphism

CSE1IOX Intermediate Object-Oriented Programming

Before you begin

Objectives

This is an individual assignment. Students are not permitted to work in a group when writing this assignment.

Copying and Plagiarism

This is an individual assignment. Students are not permitted to work in a group when writing this assignment. Plagiarism is the submission of another person's work in a manner that gives the impression that the work is their own. La Trobe University treats plagiarism seriously. When detected, penalties are strictly imposed.

Further information can be found on http://www.latrobe.edu.au/students/academic-integrity/explanation/plagiarism

Submission Guidelines

Your assignment submission should be typed, not written/drawn by hand.

Submit the electronic copy of your assignment through the subject LMS.

Submission after the deadline will incur a penalty of 5% of the available assignment mark per day capped at 5 days. No assignment will be accepted after 5 days. If you have encountered difficulties that lead to late submission or no submission, you should apply for special consideration.

Background

‘Green Guard’ is a garden maintenance company that is growing quickly. The company is struggling to keep track of all their equipment and have engaged you to develop a program to assist them with identifying when their equipment needs to be replaced.

‘Green Guard’ has three categories of equipment:

- Standard equipment (i.e. Rakes, Brooms and Shovels)
- Fuel powered equipment (i.e. Lawn Mowers, Chainsaws)
- Battery powered equipment (i.e. Edge Trimmers, Brush Cutters)

The program will need to be able to manage:

- The display of common information about all equipment
- The display of information relevant to specific equipment categories
- When a piece of equipment needs to be replaced

It is recommended that you read through all the tasks before commencing work on your assessment.

This assessment consists of the creation of one abstract class and three inherited classes. You may wish to consider implementing and testing the abstract class and one of its inherited classes before implementing all classes. The Battery Powered Equipment class is most straightforward to implement.

Requirements

In terms of when equipment needs to be replaced, that depends on the type of equipment:

- **Battery Powered Equipment** needs to be replaced when the equipment warranty runs out using the following formula:

  Replacement Year = Length of Warranty + Purchase Year

  i.e. for a battery powered item with a warranty of 2 years purchased in 2022

  Replacement Year = 2 + 2022
  Replacement Year = 2024

- **Fuel Powered Equipment** needs to be replaced after being used for more than 500 days. The number of days that a piece of equipment is used in a year is approximated to determine when it should be replaced using the following formula:

  Replacement Year = Maximum Days / Days Usage Per Year + Purchase Year

  i.e. for a fuel powered item that is used approximately 100 days in a year purchased in 2022

  Replacement Year = 500 / 100 + 2022
  Replacement Year = 2027

- **Standard equipment** needs to be replaced a maximum of 7 years after it was purchased, but this is also dependent on the durability of the equipment. A durability factor is used to assist in determining the replacement date. The durability factor is a number between 0 and 1. The overall calculation of the replacement year is made using the following formula.

  Replacement Year = Maximum Lifespan * Durability Factor + Purchase Year

  i.e. for a standard item with a durability of 0.6 purchased in 2022

  *Replacement Year = 5 * 0.6 + 2022*
  Replacement Year = 2025

  Replacement years should always be rounded up

Tasks

Task 0 – Creation of NetBeans Project

As a starting point create a new NetBeans project and give it the name ‘ReplacementTest’. Your main class for the project should (by default) have the same name. The aim of this class is to enable testing to determine if all aspects of the program are working as expected. You can use it to run testing as you develop your project and also once it is complete.

Task1 – Creation of Abstract Equipment Class

You have determined that the program would be best managed using an abstract class as a starting point.

The Equipment abstract class will feature standard properties and methods that are common between all equipment categories.

You will need to create the abstract class as outlined below:

- There are three class properties (make, model and purchaseYear) – you will need create methods to ‘get’ and ‘set’ each of these
- You are to create an abstract method called ‘replacementYear’. This is only the shell of a method that will be overridden in child classes i.e. there is no need to implement this method.
- You are to create a method called ‘showDetails’ that displays the ‘make’, ‘model’, ‘purchaseYear’ and ‘replacementYear’ of equipment as one string.

Abstract Class - Equipment

Properties

Name	Data Type	Description
make	String	The make (brand) of the equipment (private)
model	String	The model of the equipment (private)
purchaseYear	int	The year that the equipment was purchased (private)

Methods

Name	Return value	Input parameters	Description
Equipment (Constructor)		String, String, int	Initializing class by providing Model, Make and Year of Purchase.
setMake	Void	String	Set the equipment Make
getMake	String		Get the equipment Make
setModel	Void	String	Set the equipment Model

getModel	String		Get the equipment Model
setPurchaseYear	Void	int	Set the equipment Purchase Year
getPurchaseYear	int		Get the equipment Purchase Year
replacementYear	int		Abstract method
showDetails	String		Print equipment make, model, purchase year and replacement year.

Task 2 – Creation of Battery Powered Equipment Class

You are to create a class named ‘BatteryPoweredEquipment’. This class is designed to keep track of the replacement of battery powered equipment such as Edge Trimmers and Brush Cutters.

You will need to create the class as outlined below:

- The ‘BatteryPoweredEquipment’ class should inherit from the ‘Equipment’ class (take advantage of super)
- The class should have a property to store the warranty (measured in years) of individual items and should also have associated functions to get and set the usage.
- At class initialisation you should be able to set the make, model, purchase date and warranty of the item
- The replacementYear function should be overridden to return the replacement year (based on calculations outlined in the requirements section)

Class - BatteryPoweredEquipment

Properties

Name	Data Type	Description
warranty	int	Private

Methods

Name	Return value	Input parameters	Description
BatteryPoweredEquipment (Constructor)		String,string,int, int	Initializing class
replacementYear	int		Return year of replacement based on calculation
setWarranty		int	Pay per hour
getWarranty	int		

Task 3 – Creation of Fuel Powered Equipment Class

You are to create a class named ‘FuelPoweredEquipment’. This class is designed to keep track of the replacement of fuel powered equipment such as lawn mowers and chainsaws.

You will need to create the class as outlined below:

- The ‘FuelPoweredEquipment’ class should inherit from the ‘Equipment’ class (take advantage of super)
- The class should have a constant value that sets the maximum number of days an item can be used for.
- The class should have a property to store the usage (measured in days per year that an item is used) of individual items and should also have associated functions to get and set the usage.
- At class initialisation you should be able to set the make, model, purchase date and usage of the item
- The replacementYear function should be overridden to return the replacement year (based on calculations outlined in the requirements section)

Class - FuelPoweredEquipment

Properties

Name	Data Type	Description
usage	int	Private
MaximumDays	int	Constant

Methods

Name	Return value	Input parameters	Description
FuelPoweredEquipment (Constructor)		String, string, int, int	Initializing class
replacementYear	int		Return year of replacement based on calculation
setUsage		int	
getUsage	int		

Task 4– Creation of Standard Equipment Class

You are to create a class named ‘StandardEquipment’. This class is designed to keep track of the replacement of standard (non-powered) equipment such as rakes, brooms and shovels.

You will need to create the class as outlined below:

- The ‘StandardEquipment’ class should inherit from the ‘Equipment’ class (take advantage of super)
- The class should have a constant value that sets the maximum time an item can be retained
- The class should have a property to store the durability of individual items and should also have associated functions to get and set the durability
- At class initialisation you should be able to set the make, model, purchase date and durability of the item
- The replacementYear function should be overridden to return the replacement year (based on calculations outlined in the requirements section)

Class - StandardEquipment

Properties

Name	Data Type	Description
durability	double	Private
maximumRetention	int	Constant

Methods

Name	Return value	Input parameters	Description
StandardEquipment (Constructor)		String, String, int, double	
replacementYear	int		Return year of replacement based on calculation
setDurability		double	
getDurability	double		

Task 5 – Testing Using Replacement Testing Class

This class should have already been created as part of Task 0.

As part of the 'main' method of this class you should:

- Create two instances of each of the three types of Equipment classes using dummy data
- The output should be according to what has been defined in the 'showDetails' method in each class. Your output should be starting to look like that as outlined under *Figure 1 – Expected Output (*the exceptions are covered as part of the next Task)

Task 6 - Exceptions

You will need to run some checks on the validity of data been entered. It is expected that the program will throw an exception in the following circumstances:

- If the warranty of a Battery Equipment item is 0 or less
- If the durability of a Standard Equipment item is 0 or less OR greater than 1

You must have a catch block that catches these exceptions and displays an appropriate error message. (You can use the getMessage method of the Throwable class to achieve this)

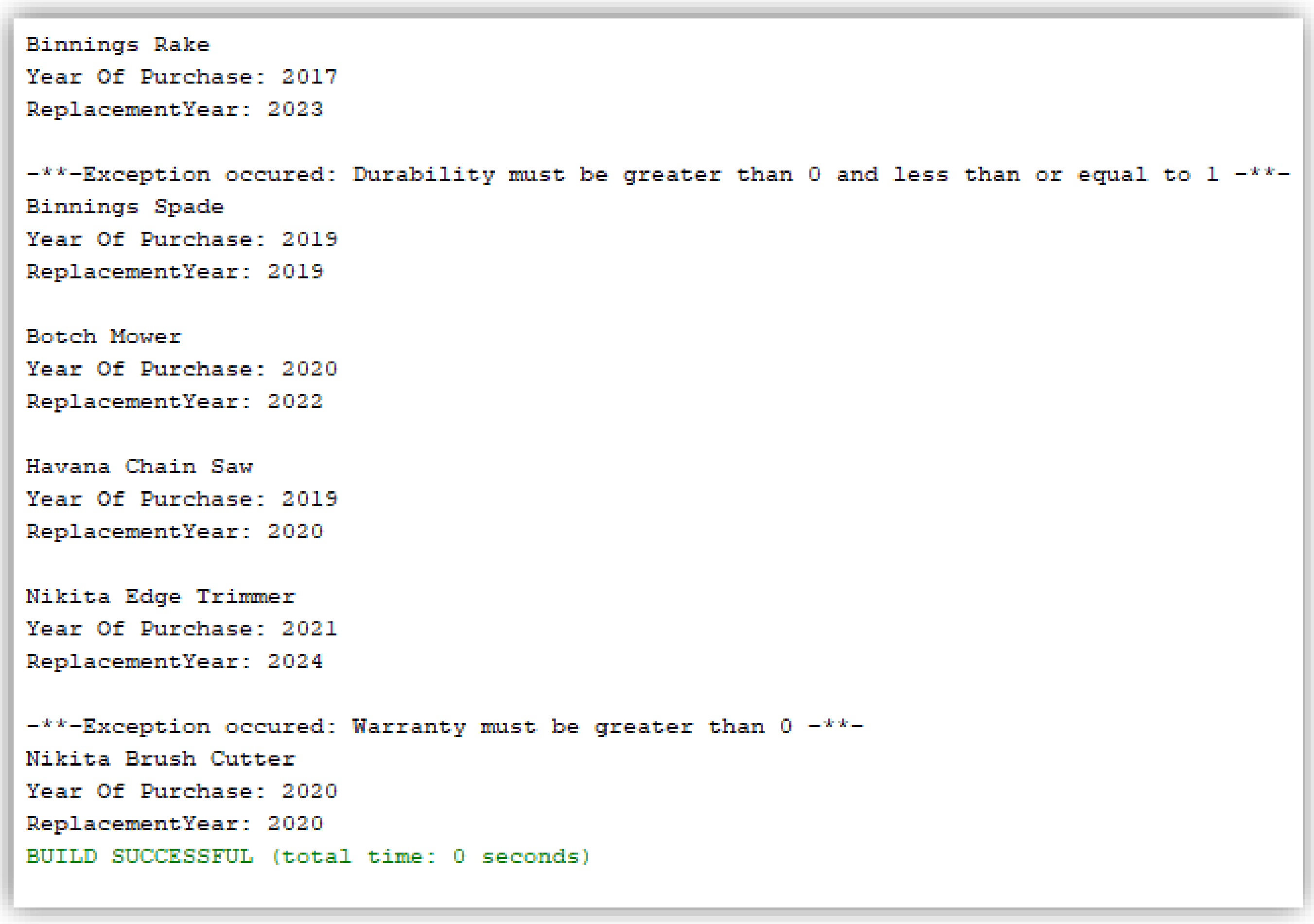
You only need to alert the user to there being an error with the entered data. You do not need to manage this in any other way at this point.

Your output should be similar to that outlined under *Figure 1- Expected Output* below

Task 7 - Comments

Ensure you include appropriate comments in your code including:

- Purpose of classes
- Purpose of methods
- Explanatory notes for any potentially confusing items



```
Binnings Rake
Year Of Purchase: 2017
ReplacementYear: 2023

-**-Exception occured: Durability must be greater than 0 and less than or equal to 1 -**-
Binnings Spade
Year Of Purchase: 2019
ReplacementYear: 2019

Botch Mower
Year Of Purchase: 2020
ReplacementYear: 2022

Havana Chain Saw
Year Of Purchase: 2019
ReplacementYear: 2020

Nikita Edge Trimmer
Year Of Purchase: 2021
ReplacementYear: 2024

-**-Exception occured: Warranty must be greater than 0 -**-
Nikita Brush Cutter
Year Of Purchase: 2020
ReplacementYear: 2020
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 1 Expected Output

Submission:

When you have completed, submit the solution to these tasks via the link on the LMS.

You should submit the following:

- xxx_cse1iox_assessment1.zip containing the java project

Assessment Marking Criteria

Requirement	**Criteria**	**Points**
Inheritance	• The following classes have been correctly implemented to extend from the abstract Equipment class ○ StandardEquipment ○ BatteryPoweredEquipment ○ FuelPoweredEquipment • The constructors in the above classes make use of the super keyword to call their parent constructor	/2
Polymorphism	• The replacementYear method is overridden as expected in each of the following classes ○ StandardEquipment ○ BatteryPoweredEquipment ○ FuelPoweredEquipment	/2
Constructors	• Constructors for the following classes have been implemented as expected: ○ Equipment class ○ StandardEquipment ○ BatteryPoweredEquipment ○ FuelPoweredEquipment	/2
Methods	• Getter and Setter methods have been implemented for all non-constant variables in the following classes ○ Equipment class ○ StandardEquipment ○ BatteryPoweredEquipment ○ FuelPoweredEquipment • The showDetails method has been correctly implemented in the Equipment class	/2
Logic	• The logic for the calculation of ReplacementYear in each of the following classes is correct ○ StandardEquipment ○ BatteryPoweredEquipment ○ FuelPoweredEquipment	/2
Output	• Output is laid out in a format that is easy to interpret	/1
Exception Handling	• Exceptions for the following are managed as expected ○ Warranty of Battery ○ Durability of Standard Equipment • Test cases that trigger exceptions are included in output	/2
Comments	• Comments are included that cover ○ Purpose of classes ○ Purpose of methods ○ Explanatory notes for any potentially confusing items	/2

Total: 15 points