PROG2004
Object Oriented Programming

Summary

Title	**Assessment 1**
Type	Programming
Due Date	**Monday 13 November 11:59 pm (Start of Week 3)**
Length	15 hours
Weighting	30%
Academic Integrity	Contract cheating and the use of GenAI, such as ChatGPT, in this assignment are **strictly prohibited**. **Any breach may have severe consequences.**
Submission	You will need to submit your code for tasks in Parts 1 - 7 of the assessment by writing your code in Java.
Unit Learning Outcomes	This assessment task maps to the following ULOs: • ULO1: explain, compare, and apply advanced class design concepts • ULO2: apply object-oriented programming principles to solve intermediate problems

Rationale

The purpose of this assessment is to test your ability to:

- Explain, compare, and apply advanced class design concepts; and
- Apply object-oriented programming principles to solve intermediate problems

Your work will demonstrate your learning over the first two modules of this unit.

Task Description

Your task is to complete various exercises using Java and to submit these via the MySCU link created for this purpose. There is a video associated with this assignment called Assessment One Specifications and Submission. Please do not start the assignment without watching the video, as it contains important information about the following points.

In this assessment, you MUST:

- Use **IntelliJ** and the **WakaTime plugin** to track the time spent on your project.
- Submit **screenshots of your WakaTime dashboard** for your Assignment One project showing the amount of time you worked on each one of your files over the last two weeks.
- Submit a **video explaining your code** as outlined in this document.

If you fail to do any of the above, you will lose marks and be required to attend an interview to explain your code. If you cannot explain your code, you will be submitted for academic misconduct.

Getting Started

In this assignment, you will write the code that manages the product categories on the **Amazon** website at https://www.amazon.com.au.

To get started:

- Create a new Java project called **username-A1** in IntelliJ.
- In the **src** directory, create a new class called **AssignmentOne**.
- In the AssignmentOne class, create the **main method**.

Module 1 - Advanced class design

The following part of the assessment covers the content in Module 1.

Part 1 – Creating abstract classes and interfaces

Go to https://www.amazon.com.au and explore all the different categories of products that they sell. While you are doing this, have a look at the different product categories as they go from more general to more specific. At the top level of the Inheritance hierarchy, you need a generic product. In your Java project:

- Create a new interface called **Product** with **at least two generic methods** that are suitable for ALL product categories on the Amazon website.

Amazon sells physical and digital products. Therefore, in the inheritance hierarchy, under products, you would need to handle physical products and digital products. In your Java project:

- Create an **abstract** class called **PhysicalProduct** that implements the interface called Product.
- Create an **abstract** class called **DigitalProduct** that implements the interface called Product.

The PhysicalProduct and DigitalProduct abstract classes must have the following at a minimum that are suitable for ALL physical OR digital product categories on the Amazon website:

- At least two instance variables
- A default constructor
- A second constructor that sets all the instance variables
- At least one abstract method
- At least one non-abstract method
- Any other methods or variables that you feel are necessary for the class to be usable in the program

Part 2 – Using abstract classes and interfaces

On the Amazon website, **choose one physical product category** and **one digital product category**. In your project:

- Create **one concrete class** that extends the **PhysicalProduct** abstract class.
- Create **one concrete class** that extends the **DigitalProduct** abstract class.

The classes **should match the physical product category and digital product category you chose on the Amazon site** and have the following at a minimum:

- At least two instance variables (one of these variables must be a boolean as a boolean instance variable is required in a later section of the assignment)
- A default constructor
- A second constructor that initialises all the instance variables (including the instance variables in the abstract superclass)
- A method that prints the Product details, e.g. "The product details are:" followed by all instance variables formatted for readability (including the instance variables in the abstract superclass)
- Any other methods or variables needed **so that your products match the physical product category and digital product category you chose on the Amazon site**.

For each class, in the class comments, you **MUST provide a link to the product category** on Amazon that you based your class on.

In the main method:

- Add the following comment - // Part 2 – Using abstract classes and interfaces
- Create an **object for the concrete class that extends the PhysicalProduct** abstract class using **the constructor that sets all instance variables**.
- Create an **object for the concrete class that extends the DigitalProduct** abstract class using **the constructor that sets all instance variables**.
- Add the following code - System.out.println("----------");

Part 3 – Polymorphism

In the **AssignmentOne class**, write a **method** called **demonstratePolymorphism** that takes one parameter. The parameter type can be either a:

- Product
- PhysicalProduct
- DigitalProduct

In the main method:

- Add the following comment - // Part 3 – Polymorphism
- Add a second comment that explains how you are going to use the method you just created to demonstrate your understanding of polymorphism
- Write the **code to create an object** and **use the method** you just created **to demonstrate your understanding of polymorphism**
- Add the following code - System.out.println("----------");

NOTE: Do not remove the code in the main method for Part 2 (or any other part of the assignment). You can comment it out when you are developing; however, when you submit your assignment, the main method must contain all the code required for all parts of the assignment, i.e., your marker should be able to run your main method, and all parts of your assignment should run in one go.

Module 2 – Generics and lambdas

The following part of the assessment covers the content in Module 2.

In Part 2 of this assessment, you **created a concrete class that extends the PhysicalProduct** abstract class. **For the remainder of this assessment, this class will be referred to as yourClass as, in theory, all students should have a different name for this class**.

Part 4 – Generic classes

In this part of the assignment, you are going to write the code for an ArrayList that can store instances (objects) of yourClass.
In the main method, write the code to:

- Add the following comment - // Part 4 – Generic classes
- Declare an **ArrayList**
- Add **at least 5 instances of yourClass to the ArrayList**
- Add the following code - System.out.println("----------");

Part 5 – Generic methods

In this part of the assignment, you are going to sort the ArrayList that we created in part 4.
In yourClass, implement the Comparable interface. When you implement the *compareTo()* method from the Comparable interface, **you must use at least two instance variables in the comparison.**
Once you have implemented the Comparable interface in the main method:

- Add the following comment - // Part 5 – Generic methods
- Write the code to sort the ArrayList that you created in Part 4.
- Add the following code - System.out.println("----------");

Part 6 – Wildcards

In the **AssignmentOne class**, create a **method** called **DemonstrateWildcards** that takes an **ArrayList generic parameter <? extends T>**. In the method, call one of the methods from **PhysicalProduct** abstract class.
In the main method:

- Add the following comment - // Part 6 - Wildcards
- Add a second comment that explains how you are going to use the method you just created to demonstrate your understanding of wildcards
- Write the code to **create an object** and **use the method** you just created **to demonstrate your understanding of wildcards**
- Add the following code - System.out.println("------------------------------");

Part 7 – Simple lambda expressions

In the **AssignmentOne class**, create a **method** called **DemonstrateLambdas** that takes an **ArrayList of yourClass** and a **Predicate** as a parameter. In the method, **test the Boolean instance variable from yourClass** and print a suitable message based on whether it is true or false.
In the main method:

- Add the following comment - // Part 7 - Simple lambda expressions
- Write the code to pass the Arraylist that you created in Part 4 to the DemonstrateLambdas method
- Add the following code - System.out.println("------------------------------");

Task Instructions

To complete the assignment tasks, you are required to follow the following steps:

1. Step 1: Access the assignment one link in MySCU;
2. Step 2: Read the task description carefully to clearly understand what you need to do. If you have questions, use the Discussion Board to ask.
3. Step 3: Read the marking rubric to understand how your work will be assessed. If you have questions, use the Discussion Board to ask.
4. Step 4: Study modules 1 and 2 and spend about 7.5 hours each week to complete the relevant assignment tasks.
5. Step 5: Write your code for each task while the WakaTime plugin is installed on your IntelliJ IDE.
6. Step 6: Record a video explaining your code.
7. Step 7: Submit your assessment.

Resources

To complete the task, you are recommended to:

- Study modules 1 and 2 materials and complete all learning activities
- Take an active role in the weekly tutorial and workshop.

Task Submission

You are required to submit three items for this assessment, including:

- Your Java project
- Screenshots of your WakaTime dashboard for your project, showing the amount of time you worked on each one of your files in your project, and the overall time you have worked on your project over the last two weeks.
- A short video as outlined below

These items are outlined below:

Java Project

You should now have the following in your main method:

```
// Part 2 - Using abstract classes and interfaces
Code demonstrating part 2
System.out.println("-----------------------------");
// Part 3 - Polymorphism
Code demonstrating part 3
System.out.println("-----------------------------");
// Part 4 - Generic classes
Code demonstrating part 4
System.out.println("-----------------------------");
// Part 5 - Generic methods
Code demonstrating part 5
System.out.println("-----------------------------");
// Part 6 - Wildcards
Code demonstrating part 6
System.out.println("-----------------------------");
// Part 7 - Simple lambda expressions
Code demonstrating part 7
```

```
System.out.println("------------------------------");
```

Make sure that none of the code demonstrating each part of the assignment is commented out. Your marker will run your main method and will be expecting to see the output for all parts of the assignment. If the output for any part is missing, you WILL lose marks for that part.
Zip your project into a file called username-A1.zip. Please note that the extension of the file must be zip. Any other compression, e.g., .rar, will NOT be accepted.

Video

Create a short video:

- Showing that the concrete classes that you created in Part 2 and the categories you based these classes on that you chose on the Amazon site
- Explaining how you demonstrated polymorphism in Part 3
- Explaining how you implemented the comparable interface in Part 5 and why you chose the 2 instance variables that you did for the comparison
- Explaining how you demonstrated wildcards in Part 6
- Explaining how you demonstrated lambda expressions in Part 7

Your video does not need to be long or go into a lot of detail. You should be able to do all the above in <= 5 minutes; however, the video must demonstrate that:

- You designed your concrete classes in Part 2 based on categories on the Amazon site
- You understand the code you are submitting in the remaining parts, and did not use ChatGPT or similar GenAI to generate it.

Upload your video to your SCU OneDrive and create a sharable link to it.

WakaTime Screenshots

Login to your account on WakaTime and **take two screenshots of your WakaTime dashboard** for your project, showing the amount of time you worked on each one of your files over the last 2 weeks. **Your screenshot MUST show the information as outlined in the assessment submission video.**

Submission

- Using the Assignment 1 link on MySCU, **submit the username-A1.zip file and the screenshots of your WakaTime dashboard** by the due date. These MUST be separate files, i.e., do not put the screenshot in the zip file. Please leave enough time for it to upload, so do not submit it at the last minute!
- **In the notes field, provide the link to your video.**

Please watch the instruction video available in the Assignment One link in MySCU.

Assessment Criteria

Please refer to the rubric provided in the assessment folder for the assessment criteria. Marking criteria include:

- Java code compiles with Java 17 LTS
- Use of correct coding style, including the use of comments
- Accuracy of coding
- Use of suitable coding structures
- Correct submission and naming conventions of assessment items as required

Academic Integrity

At Southern Cross University, academic integrity means behaving with the values of honesty, fairness, trustworthiness, courage, responsibility and respect in relation to academic work.

The Southern Cross University Academic Integrity Framework aims to develop a holistic, systematic and consistent approach to addressing academic integrity across the entire University. For more information, see: SCU Academic Integrity Framework.

NOTE: **Academic Integrity breaches include** unacceptable use of generative artificial intelligence (GenAI) tools, the use of GenAI has not been appropriately acknowledged or is beyond the acceptable limit as defined in the assessment, poor referencing, not identifying direct quotations correctly, close paraphrasing, plagiarism, recycling, misrepresentation, collusion, cheating, contract cheating, fabricating information.

Use of Gen AI such as ChatGPT

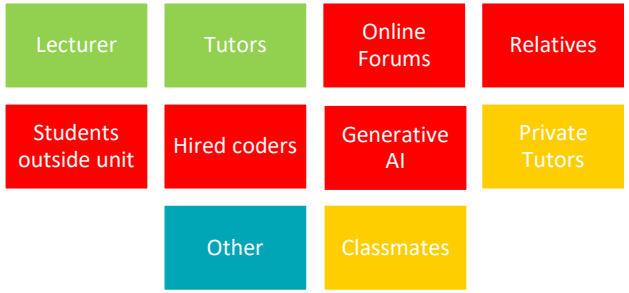
Generative artificial intelligence (GenAI) tools, such as ChatGPT, **must not be used** for this assessment task. You are required to demonstrate that you have developed the unit's skills and knowledge without the support of GenAI. If you use GenAI tools in your assessment task, it may result in an academic integrity breach against you, as described in the Student Academic and Non-Academic Misconduct Rules, Section 3.

Please note that your assignment will be submitted to a similarity detection service by your marker. Advanced plagiarism detection software will be used that compares the answers for all students to all questions and flags any similarities in assessments. If your marker has any suspicion that you had help with your

code or that your work is not your own, you will be asked to come to a meeting with your marker to explain your code. Any student who is unable to explain their code will be submitted for academic misconduct.

Getting Help

This diagram will help you understand where you can get help:



Encouraged **Attribution Required**
Not acceptable **Ask tutor**

Be aware that if you do get help from one of the red sources, you will be reported for academic misconduct, which may have serious penalties. Please visit the following link for the guidelines: **https://bit.ly/scuAcadMisconduct**

Special Consideration

Please refer to the Special Consideration section of Policy. https://policies.scu.edu.au/document/view-current.php?id=140

Late Submissions & Penalties

Please refer to the Late Submission & Penalties section of Policy. https://policies.scu.edu.au/view.current.php?id=00255

Grades & Feedback

Assessments that have been submitted by the due date will receive an SCU grade. Grades and feedback will be posted to the 'Grades and 'Feedback' section on the Blackboard unit site. Please allow 7 days for marks to be posted.

Description of SCU Grades

High Distinction:

The 'student's performance, in addition to satisfying all of the basic learning requirements, demonstrates distinctive insight and ability in researching, analysing and applying relevant skills and concepts, and shows exceptional ability to synthesise, integrate and evaluate knowledge. The 'student's performance could be described as outstanding in relation to the learning requirements specified.

Distinction:

The 'student's performance, in addition to satisfying all of the basic learning requirements, demonstrates distinctive insight and ability in researching, analysing and applying relevant skills and concepts, and shows a well-developed ability to synthesise, integrate and evaluate knowledge. The 'student's performance could be described as distinguished in relation to the learning requirements specified.

Credit:

The 'student's performance, in addition to satisfying all of the basic learning requirements specified, demonstrates insight and ability in researching, analysing and applying relevant skills and concepts. The 'student's performance could be described as competent in relation to the learning requirements specified.

Pass:

The 'student's performance satisfies all of the basic learning requirements specified and provides a sound basis for proceeding to higher-level studies in the subject area. The 'student's performance could be described as satisfactory in relation to the learning requirements specified.

Fail:

The 'student's performance fails to satisfy the learning requirements specified.