OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH
TECHNOLOGICAL UNIVERSITY DUBLIN

Technological University Dublin
Blanchardstown Campus

Academic term	2024-2025
Year of study	2

Programme code	Programme title	Module code
TU653	Higher Certificate in Science in Computing in Information Technology	COMP H2030
TU757	Bachelor of Science in Computing in Information Technology	COMP H2030
TU860	Bachelor of Science (Honours) in Computing	COMP H2030
TU863	Bachelor of Science (Honours) in Computing in Digital Forensics and Cyber Security	COMP H2030

Module title	**VERSION 5 – Musical Instruments GUI** Advanced Programming – Final Assignment 3rd April 2025

Internal Examiner(s): **Dr. Luke Raeside**
External Examiner(s): **Dr. Graham Healy**

Instructions to candidates (READ VERY CAREFULLY):

1) **Please check that the module and programme which you are following is listed in the table above.**
2) **DOWNLOAD AND COMPLETE THE CORRET VERSION OF THE ASSIGNMENT FROM BRIGHTSPACE (THERE ARE SIX VERSIONS):**
   - **VERSION 1 for students with student number ending with 2.**
   - **VERSION 2 for students with student number ending with 4 or 6.**
   - **VERSION 3 for students with student number ending with 3 or 8.**
   - **VERSION 4 for students with student number ending with 5 or 7.**
   - **VERSION 5 for students with student number ending with 0 or 1.**
   - **VERSION 6 for ALL OTHER students.**
3) **YOU MUST CHOSE THE CORRECT ASSIGNMENT VERSION.**
4) **Attempt ALL parts of this assignment.**
5) **This is a 1 day assignment – upload the assignment to BrightSpace before 23:59pm on April 3rd 2025.**
6) **Record <u>TWO</u> short video clips as part of this assignment (3 minutes approx.). Record a video <u>before you start to code the solution</u> AND record a video <u>prior to submission of the completed code</u>. VIDEOS MUST CONTAIN TECHNICAL JAVA PROGRAMMING DETAILS.**

- **The initial video will summarize your plans for the coded solution, any difficulties you may feel you will have to overcome and include any designs or diagrams you have sketched (show to the camera).**
- **The second video will summarize <u>what you managed to complete</u> and summarize any difficulties encountered. Show the completed programme running, if possible, in this video. Ensure that you submit BOTH videos with the assignment. Failure to enclose the videos could result in an additional viva-voce examination (with a -20% deduction), i.e., a follow-up one-to-one assessment. <u>If you need to upload a link to the videos</u> the links must be included in the submission and the lecturer must have download access.**

7) **IMPORTANT RULES FOR THE VIDEOS:**
    a. **I MUST <u>SEE YOUR FACE</u> TO INTRODUCE THE VIDEOS**
    b. **I MUST <u>HEAR YOUR VOICE</u> THROUGHOUT THE VIDEOS**
    c. **VIDEOS <u>MUST BE IN ENGLISH</u>.**

   Failure to follow the above rules will result in a viva-voce examination being required, i.e., a follow-up one-to-one assessment that will also incur a -20% penalty to your result.
8) **You MUST attempt to use the AssignmentLogger class correctly in your assignment. This class will log your assignment coding each hour. Failure to reasonably attempt to use the AssignmentLogger will result in a viva-voce examination being required, i.e., a follow-up one-to-one assessment that will also incur a -20% penalty to your result. Don't allow any possible logger problems prevent you from submitting an attempt for this assignment – try your best!.**
9) **Java 1.8 is a minimum requirement (Java 10+ recommended).**
10) **Self-assessment is optional for this assignment (rubric provided).**
11) **ALL THE SUBMITTED WORK MUST BE YOUR OWN. <u>THIS IS AN INDIVIDUAL ASSIGNMENT YOU CANNOT ASSIST OTHERS IN ANY WAY OR SEEK ASSISTANCE FROM ANYONE ELSE IN ANY WAY</u>. ZERO MARKS WILL BE AWARED TO ANY STUDENT THAT ASSISTS ANOTHER OR RECEIVES ASSISTANCE WITH THIS ASSIGNMENT.**
12) **USE OF GENERATIVE AI IS NOT PERMITTED WITH THIS ASSIGNMENT.**
13) **LATE SUBMISSION WILL INCUR PENALTIES:**
    - **Within first hour late will result in an automatic grade reduction of 20%, additional 20% reduction within every hour late thereafter, e.g., 1-2 hours late -40%, 2-4 hours late -60% etc.**
14) **YOU HAVE ALREADY SIGNED A COMMITMENT REGARDING ACADEMIC HONESTY THROUGH THE UNIVERSITY - HONOUR THAT COMMITMENT.**

THE NEXT SECTIONS PROVIDE DETAILED INSTRUCTIONS RELATING TO THE ASSIGNMENT PLEASE ENSURE THAT YOU READ ALL OF THE INSTRUCTIONS CAREFULLY.

VERSION 5: Musical Instruments [Student number ending in 0 or 1]

1. Assignment Description:

You are required to develop an international GUI that will display an image of AT LEAST THREE musical instruments and play a sample sound of each instrument. The sample sound should be small and simple (no full songs please! – as data upload size is limited). The GUI will be constructed using a Java JFrame. The GUI MUST provide an input FIELD (an input field is **required**) for the instrument you want to display, e.g., type in "Guitar" and a Guitar appears in the Frame with a description and the sound will play when activated. The GUI must be capable of operating in TWO languages (ONE of the languages must be English). The translation of the GUI must include ALL displays and input (i.e., inputting the instrument into the field works in two languages). Use thread programming to delay the appearance of the display image by a small interval (e.g., at least half a second). You will also provide a brief translated description of the type of instrument being displayed next to the image (e.g., "This is a guitar it is a small, hand-held stringed instrument"). Provide a component (e.g., a button) to play a sound to provide an example of how the instrument sounds (sound clip max 30 seconds but preferably less than 5 seconds).

2. IMPORTANT!: Use of the Logging package:

USE THE AssignmentLogger CLASS THROUGHOUT YOUR SOLUTION BY CALLING THE FOLLOWING METHODS AT THE SPECIFIED POINTS OF YOUR PROGRAM (You can import using Logger.jar or paste the logging package with AssignmentLogger.java in the package in to your project, find it in BrightSpace):

- AssignmentLogger.logConstructor(this) – at the entry of every constructor (pass 'this' as parameter). Note: If you are using **super** in the constructor you need to move the log call to the after the **super** call (**super** call must be first line). **YOU MUST CALL THIS METHOD FROM A CONSTRUCTOR FOR THE LOGGER TO WORK.**
- AssignmentLogger.logMethodEntry(this) and AssignmentLogger.logMethodExit(this) at entry (first line) and exit (last line) of every non-static method (pass 'this' to the method as a parameter and it will register method and class details, e.g., AssignmentLogger.logMethodEntry(this))
- AssignmentLogger.logStaticMethodEntry() and AssignmentLogger.logStaticMethodExit() at entry (first line) and exit (last line) of every static method (no parameter should be passed – include main methods that are static, e.g., AssignmentLogger.logStaticMethodEntry())

DO NOT DELETE THE GENERATED LOG FILE(S). INCLUDE ALL OF THE LOG FILES WITH YOUR SUBMISSION. THE LOGGER WILL GENERATE ONE LOG FILE PER HOUR BY DEFAULT. THIS IS AN ANTI-PLAGIARISM MECHANISM AND NOT INCLUDING THE LOGGER MAY RESULT IN LOSS OF MARKS OR A FOLLOW-UP VIVA-VOCE EXAM (i.e., YOU MAY HAVE TO DISCUSS YOUR SUBMISSION WITH -20% REDUCTION OF YOUR GRADE).

3. Modularization:

Your code will be split into at least five packages. One of the packages is already pre-written (the package called *logging,* add this to the project). The other four packages will be:

- **exceptions** – contains implemented custom exception(s)
- **gui** – contains all of the GUI code.
- **internationalization** – contains any classes or files associated with internationalization
- **instruments** – contains code relating to the warning system, e.g., object oriented hierarchical classes to define the instruments attributes and behaviours

There will also be TWO directories (already supplied) called *images* and *sounds* where the images to display and sample sounds will be supplied (these are samples you may wish to use: you can also use alternative appropriate images and sounds of a similar size if you wish – please include them with your submission\upload!).

4. Internationalization:

You will provide a way to dynamically switch the entire running program from one language to another and back again. One of the languages is **English** (required for assessment purposes!): you may choose the second language. The translation must include warning messages, all components and hard-coded strings and input strings. Use the Java ResourceBundle class to achieve the translations. Include all translation files related in the submission.

5. Threads:

When the GUI displays the musical instrument image, using **thread programming** implement a small delay before displaying the image to the GUI (minimum half a second delay, i.e., 500 milliseconds).

6. Instruments (hierarchy):

Create a hierarchy of classes to represent the Musical Instruments. Implement a base\super class and several sub-classes for each instrument type. The base\super class will define the generic attributes and behaviours of instruments in the system. The base\super class will be an abstract class or interface. The subclasses will implement any specialized attributes and\or behaviours. Marks will be awarded for the design and use of polymorphism and dynamic binding within the system (see marking scheme).

7. Exception handling and robustness:

Exception handling must be used to error check and control the input to the field. Use of try and catch is expected. At least ONE Custom Exception class must be created and used within the system, e.g., if the input for the input field is not a recognized musical instrument you could catch\throw an unrecognized instrument exception.

8. Marking Guideline Overview (400 marks in total – see detailed marking guideline for breakdown of marks):

1. Exceptions and exception handling: (40 marks)
2. The GUI and general program success: (170 marks)
3. Internationalization (2 languages): (80 marks)
4. Musical Instruments (Object Oriented): (70 marks)
5. Comments, Naming\Formats, Logger (40 marks)

9. Supporting artifacts\material:

- Logging package and AssignmentLogger class
- Images sub-directory for the assignment (with images used)
- Sound sub-directory for the assignment (with sounds used)

10. Self-assessment (optional):

You have been supplied with a self-assessment sheet (within this document and as a separate Excel spreadsheet). You MAY use the self-assessment sheet to indicate to the assessor how you feel you have performed and enter comments. Submit the completed self-assessment sheet **optionally** with the assignment upload.

11. Detailed marking guideline (Self-assessment sheet available in separate Excel file):

Exceptions and exception handling	**Available Marks**
Robustness of system (e.g. Input checked)	15
Custom exception class created and used	15
Use of try and catch	10
	40

GUI and general program success	
Layout of GUI (including design\look)	10
Use of Panel to control layout	10
Images coded and appear	20
Threaded delay of image\info	10
Performance of GUI	10
Switching of language (performed fully using components)	20
Modularized listener code (e.g. inner class)	20
Sounds playing when listener invoked	20
Integrated use of Instrument classes in GUI	20
Use of methods to achieve goals and separate tasks	20
Display of message on GUI	10
	170
Internationalization	
Localized strings first language	10
Localized strings second language	10
Localized class\classes or ResourceBundle	20

Switch of language strings works completely (both ways, i.e., to and from English)	20
Use of Locale classes	10
Logic of internationalisation (classes/approach)	10
	80
Musical Instruments (OO)	
Base class with general attributes\methods for sub-classes (use of abstract or interface)	15
Three sub-classes with specific information\behaviours	45
Use of relationships and logic of classes	10
	70
Comments, Naming\Formats, Logger	
Javadoc (use of tags and comments throughout classes)	10
Use of logger throughout the code and classes	10
Modules (Naming and Usage of packages and imports)	10
Good naming of variables and/or attributes etc. throughout	10
	40
Total	**400**