

National College of Ireland

**Higher Diploma in Science in Computing, Year 1, HDSDEV\_SEP23**

**Higher Diploma in Science in Computing, Year 1, HDCSDEV\_INT**

**Higher Diploma in Science in Computing, Year 1, HDAIML\_SEP23OL**

**Higher Diploma in Science in Computing, Year 1, HDBC\_SEP23OL**

**Higher Diploma in Science in Computing, Year 1, HDSDEV\_SEP23**

**Higher Diploma in Science in Computing, Year 1, HDSDEV\_SPOL23\_YR1**

**Higher Diploma in Science in Computing, Year 1, HDWD\_SEP23OL**

**Certificate in Computing, Year 1, CIC\_SEPOL23**

**Semester One Terminal Assignment-based Assessment (TABA) – 2023/24**

**Release Date on Moodle: 5th of January 2023 @09:00am**

**Online Moodle Submission Deadline: 7th of January 2023 @23:55**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Software Development**

**IMPORTANT: It is your responsibility to avoid plagiarism. Please read the comprehensive guidelines on academic honesty and academic integrity, and how to avoid plagiarism made available by the NCI Library (**[**https://libguides.ncirl.ie/referencingandavoidingplagiarism**](https://libguides.ncirl.ie/referencingandavoidingplagiarism)**).**

**NOTE: YOU ARE NOT ALLOWED TO PUBLISH THIS ASSIGNMENT BRIEF OR A PART THEREOF ON ANY WEBSITES. YOU ARE NOT ALLOWED TO PUBLISH/SHARE YOUR SOLUTION WITH OTHERS.** All work submitted **should be your own** and completed **using only the programming concepts and data types covered in this module.** Conferring with others **is not permitted.**

**Note that all submissions will be electronically screened (via Turnitin) for evidence of academic misconduct (i.e. plagiarism and collusion).**

**Note:**

* This is an open book assessment.
* The requirements for your solutions are assigned based on **your student ID number**. Please read the ***Assignment Description*** section carefully for guidelines concerning the functionalities you must implement. **This is a submission requirement**. If the incorrect functionalities are implemented, no marks will be provided for those functionalities.

# Assignment Description

For this assessment you are required to implement an application that should work according to the two main questions and all their subquestions described in this section.

For each question you must plan, develop, and manually test the applications. Ensure your implementation provides a solution matching the specification in this section and the unique requirements assigned to you.

**Note**: Where user input is required this should be provided via the keyboard. For reading user input from the keyboard, you are required to either use the ***Scanner*** class in conjunction with ***System.in or*** *the* ***JOptionPane****’s* ***showInputDialog() and showMessageDialog().***

1. A company has hired you to develop an application to generate URLs (web addresses) for their website, see Figure 1 below. For the application we are developing we are concerned with three main components of the URL, the “Protocol”, the “Hostname”, and the “Path”.

A close-up of a sign

Description automatically generated

Figure An example URL for students to consider for their application.

The application prompts the user to provide one single line of text with the name of the company, for example: Meta Platforms Inc, Google LLC, Amazon Inc., Deloitte Touche Tohmatsu Ltd. (**Please note that you are not required to validate the input, we assume that the input is well formed.**) Next, the application uses the given company name to generate/create the corresponding URL containing a Protocol, Hostname, and Path. The URL is created using several rules. The rules that you should implement to generate/create the URL have been assigned based on the **penultimate (i.e. second to last) digit of your student ID number**. Please check *Table 1 Question 1. a. Rules to Generate a URL* to identify the rules assigned to you.

**Note:** The application should work irrespective of how the user provides the company name i.e. using upper case letters, lower case letters or a combination of both upper case and lower case letters. You should use the company name as provided i.e. you **should not** modify the given full company name to a different letter case. Question 1. a. and Question 1. b. are to be developed in conjunction, Question 1. a. is the instantiable class (URLGenerator) and Question 1. b. is the App class (URLGeneratorApp).

1. Develop an **instantiable class**, named **URLGenerator,** for this application which contains:
   * A class definition
   * Suitable data members (instance variables)
   * A constructor
   * A setter method to set the given company name.
   * A compute method to generate/create the URL according to the rules assigned to you in *Table 1*. **Note:** This method should demonstrate the use of programming concepts covered in our module. Marks will not be awarded for solutions that use concepts and data types/classes which have not been addressed/covered in our module. **This is a submission requirement.**
   * A getter method to return the generated URL.

The source code should be commented throughout highlighting and explaining where the key functionality is being addressed.

**(30 marks)**

* **Assigned Rules to Generate the URL**: The rules that you should use to generate the URL are assigned in *Table 1 Question 1. a. Rules to Generate a URL*. The rules are assigned based on the **penultimate digit of your student ID**. **IMPORTANT: This is a submission requirement.** If the incorrect rules are implemented, no marks will be provided for that functionality.

Table 1 Question 1. a. Rules to Generate a URL.

| **Penultimate (i.e. second to last) digit of your student ID** | **URL Rules ID** | **Rules to Generate/Create a URL for a Given company name** | **Examples**  **(Given Company Name and Corresponding URL)** |
| --- | --- | --- | --- |
| **1**  OR  **3**  OR  **5**  OR  **7**  OR  **9** | UR1 | Given a company name in full i.e., Amazon Inc.  Note: Any letter can be in either upper case or lower case in the provided company name.  To select the appropriate URL protocol:   1. If the company name contains the word “Google” regardless of case, use “https”. Otherwise use “http”   The URL hostname is generated from the given company name as follows:   1. Remove any instances of Inc., Ltd., or LLC, from the company name.   Then the rest of the URL hostname is generated from the given company name as follows:   1. Eliminate all spaces between the words in the company name and replace them with hyphens e.g., ‘-’. 2. Apart from the above-mentioned letters and characters, all the other letters should be copied unchanged to the URL hostname from the given company name. 3. If the company name contains an odd number of vowels add “.ie” to the end of the hostname. If it contains an even number of vowels add “.com” to the end of the hostname.   To set the URL’s path:   1. Count the number of pairs of consonants next to each other in the company name. If there are no pairs the path name should be “index”, 1-3 consonant pairs contain “contactDetails”, 3+ pairs “basket” | For example, if the instantiable class receives:   * The company name “Deloitte Touche Tohmatsu Ltd." then the compute method should generate the URL “http://Deloitte-Touche-Tohmatsu.com/basket”. * The company name “Google LLC" then the compute method should generate the URL “https://Google.ie/contactDetails”. * The company name “Monsters Inc.” then the compute method should generate the URL “http://Monsters.com/basket”. * The company name “Meta Ltd.” then the compute method should generate the URL “http://Meta.com/index”. |
| **0**  OR  **2**  OR  **4**  OR  **6**  OR  **8** | UR2 | Given a company name in full i.e., Amazon Inc.  Note: Any letter can be in either upper case or lower case in the provided company name.  To select the appropriate URL protocol:   1. If the company name contains the word “Meta” regardless of case, use “coap”. Otherwise use “ftp”   The URL hostname is generated from the given company name as follows:   1. Replace any instances of incorporated, limited, or limited liability company with Inc, Ltd, LLC respectively for the company name.   Then the rest of the URL hostname is generated from the given company name as follows:   1. The spaces are replaced by the underscore symbol '\_' 2. Apart from the above-mentioned letters and characters, all the other letters should be copied unchanged to the URL hostname from the given company name. 3. If the company name contains an odd number of consonants add “.org” to the end of the hostname. If it contains an even number of consonants add “.edu” to the end of the hostname.   To set the URL’s path:   1. Count the number of pairs of vowels next to each other in the company name. If there are no pairs the path name should be “bio”, 1-3 vowel pairs contain “FAQ”, 3+ pairs “Glossary” | For example, if the instantiable class receives:   * The company name “Deloitte Touche Tohmatsu limited" then the compute method should generate the URL “ftp://Deloite\_Touche\_Tohmatsu\_Ltd.edu/FAQ” * The company name “Google limited liability company" then the compute method should create the URL “ftp://Google\_LLC.org/FAQ * ” The company name “Monsters Inc.” then the compute method should generate the URL “ftp://Monsters\_Inc.edu/bio”. * The company name “Meta Ltd.” then the compute method should generate the URL “coap://Meta\_Ltd.edu/bio”. |

Example: A student with the student ID = 229876**5**4 would generate URL using the rules identified by *UR1* (because the penultimate digit of that student ID is **5**).

1. **Develop an application** that uses the instantiable class **URLGenerator** (the instantiable class previously developed at Question 1. a.) to generate URLs. The application should allow a user to enter multiple company names. In order for the application to allow entering multiple company names to generate the corresponding URL, please refer to *Table 2 Approaches to entering multiple company names to generate corresponding URLs*. The approach you have to implement has been assigned to you based on the **last digit of your student ID number**. The application will display on the screen the generated URL, based on the given company names. In the application class, please add a short comment for each method of the *URLGenerator* class that you use/call in your application to explain why that method is needed.

Name the application class **URLGeneratorApp.**

**(20 marks)**

* **Approach for entering multiple company names to generate the corresponding URLs**: Use *Table 2 Approaches to entering multiple company names to generate corresponding hostnames*. Based on the **last digit of your student ID** find the approach you have to implement in your application.

**IMPORTANT: This is a submission requirement.** If the incorrect approach is implemented, no marks will be provided for that functionality.

Table 2 Approaches to entering multiple company names to generate corresponding hostnames

|  |  |  |
| --- | --- | --- |
| **Last digit of your student ID** | **Approach ID** | **Approaches to entering multiple company names to generate/create the corresponding URL (MCNA)** |
| **0** OR  **1** OR  **2** OR  **3** OR  **4** | MCNA1 | Ask the user at the beginning of the application how many URLs they would like to generate and ensure that the application enables the user to provide that amount of company names and for each company name generates the corresponding URL. |
| **5** OR  **6** OR  **7** OR  **8** OR  **9** | MCNA2 | Ask the user to provide a company name and after the corresponding URL is generated and displayed on the screen, ask the user if they would like to generate another URL. As long as the user enters “yes” the application should work as described in the previous sentence. When the user enters anything other than “yes”, no other items are created. |

Example: A student with the student ID = 2298765**4** would implement the approach corresponding to *MCNA1* (because the last digit of that student ID is **4**).

1. Develop further the application as follows:
2. First, implement in the instantiable class *URLGenerator* (the instantiable class previously developed at Question 1. a.) **another method** which takes in as a parameter a one-dimensional array of String URLs. The method should compute whether the given URLs are valid (i.e. a given URLs conform to the rules assigned to you) or invalid (i.e. a given URLs does NOT conform to those rules). The method should return an array of Boolean values, where a true value means that a particular URL is valid, and a false value means that a particular URL is invalid. The URL assigned to you and the rules that you should implement to determine if the URLs are valid or not have been assigned based on the **penultimate digit of your student ID number**. Please check *Table 3 Question 2. a. URLs and Rules to Determine the Validity of a URL* to identify the functionality assigned to you.

**Note 1**: the method should work irrespective of the letter case used in the URL i.e. upper case letters, lower case letters or a combination of both upper case and lower case letters. The method **can modify** the letter case of the given URL. Each URL should be converted to lower case letters, and the converted URL will be further check if it is valid or not.

**Note 2**: this method should demonstrate the use of programming concepts covered in our module. Marks will not be awarded for solutions that use concepts and data types/classes which have not been addressed/covered in our module. **This is a submission requirement.**

The source code should be commented throughout highlighting and explaining where the key functionality is being addressed.

**(22 marks)**

* **Method**: You are required to implement the functionality assigned to you in *Table 3 Question 2. a. URL and Rules to Determine the Validity of a URL.* The functionality is assigned based on the **penultimate digit of your student ID**. **IMPORTANT: This is a submission requirement.** If the incorrect functionality is implemented, no marks will be provided for that functionality.

Table 3 Question 2. a. URL and Rules to Determine the Validity of a URL

|  |  |  |  |
| --- | --- | --- | --- |
| **Penultimate (i.e. second to last) digit of your student ID** | **Assigned URL** | **Rules[[1]](#footnote-1) to determine the validity of a URL** | **Example** |
| **0**  OR  **1**  OR  **2**  OR  **3**  OR  **4** | Google URLs | Functionality: Check the validity of an google URL.  A valid Google URLs satisfies all the following rules:   * Must contain Google in the hostname. * The hostname can only contain the extension “.ie” or “.com” * It cannot be shorter than 6 characters. * It can only contain letters (i.e. a – z inclusive, A – Z inclusive), digits (i.e. 0 – 9 inclusive), hyphens (i.e. ‘-’), forward slashes (i.e. ‘/’), and periods (i.e. ‘**.**’) | For example,   * if the method is passed in an array with the following URLs: * https://Google.com/bin * www.amazon.ie * google * google.com/12b * goo * then the method should determine for each URL if it’s valid or not, and return an array of Boolean values, which in this example is the array with the following values/elements: true, false, false, true, false. |
| **5**  OR  **6**  OR  **7**  OR  **8**  OR  **9** | Amazon URLs | Functionality: Check the validity of an Amazon URL  A valid Amazon URL satisfies all the following rules:   * it must contain “amazon” or “aws” in the URL hostname. * It cannot be shorter than 5 characters. * It cannot be longer than 16 characters. * It can only contain letters (i.e. a – z inclusive, A – Z inclusive), digits (i.e. 0 – 9 inclusive), and underscores (i.e. ‘\_’), forward slashes (i.e. ‘/’), and periods (i.e. ‘**.**’) | For example   * if the method is passed in an array with the following URLs: * amazon.ie/uni\_cash * aws.ie/servers * amazon * aws.com/%s * aws.com/hosts12   Then the method should determine for each URL if it’s valid, and return an array of Boolean values, which in this example is the array with the following values/elements: false, true, false, false, true. |

Example: A student with the student ID = 229876**5**4 would validate URLs (because the penultimate digit of that student ID is **5**).

1. Next, develop further the application class *URLGeneratorApp* (the class previously developed at Question 1. b) to use the method defined in *Question 2. a*. to demonstrate its functionality. First, prompt the user to provide the number of URLs they would like to validate, and then prompt the user to provide those URLs by reading one URLs at a time from the keyboard, and store those URLs in an array of URLs. Next, call/invoke/use the method defined at *Question 2. a*. to validate the URLs according to the functionality assigned to you. Finally, the application should display on the screen the values computed by the method implemented at *Question 2. a*.

**(8 marks)**

# Application Development

The applications should be developed, compiled, and run using a text editor such as TextPad (or similar Text Editor alternative for macOS or Linux users) which enables you to compile and run Java applications.

Note that Java Development Kit (JDK) has to be installed on your machine in order to compile and run your application.

Alternatively, you should be able to access these tools through the Student Virtual Desktop (details available [here](https://ncisupporthub.ncirl.ie/hc/en-ie/articles/4407343669394-How-to-I-access-the-student-virtual-desktop-)).

# Deliverables

The Terminal-Assignment Bases Assessment deliverables **must be submitted via Moodle**, they cannot be submitted by email. You are required to submit the following deliverables:

1. Complete Java source code (.java files) for the questions assigned to you.

Submit the complete Java source code as a .zip file via the submission page ***TABA Source-Code*** available on the Software Development Moodle page.

1. A report (in .pdf or .doc format) which includes:
   * A description of the input, main processing, and output (IPO) for each of the two main questions
   * The specific digits, mapped to the options/requirements that you have used to develop your applications.
   * You may optionally include a class diagram for your application to aid in its description. For creating the class diagram, you can either use an online tool, such as <http://app.diagrams.net>, or draw it by hand and take a photo.
   * Be careful to adhere to appropriate syntax and structure and identify suitable data types for each data member/ instance variable.
   * ***Note that the entire java source code of your application must also be included as an appendix in your report*! *Note that the java source code must be included as text (i.e. copy and paste your code in the report, do not take a screenshot of your code!)*.** This is a submission requirement.

Submit the report document via the ***TABA Report*** Turnitin submission page available on the Software Development Moodle page.

# Marking Scheme

The marks for this assignment will be allocated as follows:

* **Application Implementation (80 marks)**
  + Question 1. a. (30 marks)
  + Question 1. b. (20 marks)
  + Question 2. a. (22 marks)
  + Question 2. b. (8 marks)
  + Note that the implementation evaluation includes the following
    - Fully compiling and executing application with no syntax or logical errors which addresses the full requirements of the application
    - All requirements have been implemented, and the application works according to the specification assigned to you
    - The application produces accurate output
* **Good coding practices and code understanding (13 marks)**
  + The application should be fully commented throughout highlighting and explaining where the key functionalities of the application are being addressed (10 marks)
  + Well formatted and properly indented code. Appropriately named variables, methods, classes using the Java naming conventions (3 marks)
* **Report (7 marks)**
  + Evidence of designing and planning of the application prior to coding (the report should include the IPO diagram) (7 marks)

**NOTE**: the examiners reserve the right to conduct mini presentations with a sample of the students, where students will provide answers to questions related to their assignment.

1. **DISCLAIMER**: note that some of the rules specified for this functionality are provided for this assessment purposes only and may not represent the actual rules used in their real-word counterparts. You are required to implement the rules as described in this assessment. [↑](#footnote-ref-1)