**CARDIFF SCHOOL OF MANAGEMENT: ASSIGNMENT FEEDBACK PROFORMA**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| STUDENT NAME: | | | | | PROGRAMME: Computing, Software Engineering, BIS | | | |
| STUDENT NUMBER: | | | | | YEAR: 2018-19 | | GROUP: Comp & IS | |
| Module Number: CIS6003 | Term: 2 | | | | Module Title: Advanced programming | | | |
| Tutor Responsible For Marking This Assignment: Dr Thanuja Mallikarachchi, Dr Ambikesh Jayal | | | | | | | | |
| Module Leader: Ambikesh Jayal | | | | | | | | |
| Assignment Due Date: W8 Tuesday & W11 Tuesday via Moodle | | | Hand In Date: W8 Tuesday & W11 Tuesday via Moodle | | | | | |
| ASSIGNMENT TITLE: Design and develop an application to recognise handwritten digits | | | | | | | | |
| **SECTION A: SELF ASSESSMENT (TO BE COMPLETED BY THE STUDENT)** | | | | | | | | |
| **In relation to each of the set assessment criteria, please identify the areas in which you feel you have strengths and those in which you need to improve. Provide evidence to support your self-assessment with reference to the content of your assignment.** | | | | | | | | |
| STRENGTHS | | | | AREAS FOR IMPROVEMENT | | | | |
| I certify that this assignment is a result of my own work and that all sources have been acknowledged:  Signed:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | | | | | | | |
| SECTION B: TUTOR FEEDBACK **(based on assignment criteria, key skills and where appropriate, reference to professional standards)** | | | | | | | | |
| STRENGTHS | | | | AREAS FOR IMPROVEMENT AND TARGETS FOR FUTURE ASSIGNMENTS | | | | |
| MARK/GRADE AWARDED | | DATE: | | | | SIGNED | | |
| ASSIGNMENT MODERATED BY: | | | | | | | | DATE |
| MODERATOR’S COMMENTS: | | | | | | | | |

1. **Learning outcomes and key skills**

This assignment addresses the following learning outcomes of the module.

* Demonstrate fluency in contemporary programming languages, development tools and environments.
* Evaluate and demonstrate the theory and concepts of contemporary/industry standard programming and design in the software development life cycle.
* Demonstrate awareness of industry standards of professional and ethical software development, software carpentry and codemanship.

This assignment also addresses the following key skills

* Written communication
* Use of IT
* Requirement analysis, technical interpretations of problem statements

1. **Description of task and guidance notes**

**Part 1: Design and Develop application to recognise handwritten digits (70%)**

The assignment requires you to develop a java-based application to recognise handwritten digits. The inputs to the program will be a set of training images and appropriate labels (see below for details on MNIST database), along with an image of a handwritten digit, which needs to be recognised. The image of the handwritten digit to be recognised can be provided to the program either as an external image upload (a set of test images are given), or allowing the user to draw a digit using mouse, within a canvas area available in the application’s user interface.

Use case 1: User uploads an image of a handwritten digit for recognition

In this case, the program should allow the user to upload an image that contain a handwritten digit. The application should display the uploaded image, its predicted result.

Use case 2: User draws a digit using mouse inside the canvas area

In this case, the program should allow the user to draw a desired digit (0 – 9) within a canvas area using mouse motions. As a programmer, you need to think about the line thickness, background colour of the canvas based on the sample images found in the MNIST database.

Once the drawing is complete, the digit should be extracted as an image which is resized accordingly (to match the size of the training set image samples), and centred with sufficient amount of padding before being used in the recognition.

Recognition of digits using k-Nearest Neighbour (k-NN) Algorithm

Once the image with handwritten digits, and the images and labels from the training set are available, the recognition process should be initiated. There are numerous algorithms available in the literature which can be utilised for the purpose of recognition (I.e., SVMs, DNNs etc.). However, in this assignment, it is recommended to use k-NN algorithm with Euclidean distance over raw pixels as the decision criteria. The pseudo code for a potential implementation of the k-NN algorithm is given below.

train\_dataset := [train\_images, train\_lables] *# 2D array that holds training images and labels*

d\_img *# Variable that holds handwritten digit image*

distance\_lbl\_array *# 2D array to hold distances and labels*

*# iterate through each element in the train\_dataset 2D array*

for each element e in train\_dataset

t\_img := train\_dataset[e][0] *# eth training image*

t\_lbl := train\_dataset[e][1] *# eth label for the corresponding image*

*# compute the Euclidian distance between training image and current image*

# *store the* *Euclidian distance and the label of the training image*

distance\_lbl\_array[e] := [distance, t\_lbl]

end

*# Sort distance\_lbl\_array according to the distance*

sorted\_distance\_lbl\_array := sort (distance\_lbl\_array)

*# extract elements for k closest labels. e.g., if k=10, get the first 10 elements in the sorted\_distance\_lbl\_array.*

closest\_k\_label\_array := sorted\_distance\_lbl\_array[:k]

*# Find the label class with maximum number of label counts*

Estimated\_label := max\_label\_count(closest\_k\_label\_array)

*# Find the confidence as a ratio of selected label count to total number of closest elements*

Confidence := max\_label\_count(closest\_k\_label\_array)/size(closest\_k\_label\_array)

Presentation of the results

The digit recognised should be displayed with the confidence of the result.

***Hint:*** The MNIST database contain training and testing datasets. You are encouraged to develop your code to test on the images provided in test data set, before attempting to perform the recognition for the user drawn digit.

**MNIST dataset**

MNIST database of handwritten digits is a popular dataset for studying machine learning and pattern recognition methods. The details of the images and their formats are given in the MNIST official web site[[1]](#footnote-1). As specified in the web page, the images are not in any standard image format. Therefore, you may have to write your own program to read and convert them. Some helpful links are provided in the Moodle page.

**Part 2: Individual’s Work Sheets Portfolio (30%)**

For the assignment you need to work on worksheets from week 7-11 about various topics on object oriented programming. All students need to submit 3 worksheets and related code in a zip folder. The purposes of these weekly worksheets is to explain the key issues in modeling and analysing real world problems using appropriate techniques. Further details can be found from Moodle.

1. **Assessment criteria**

|  |  |
| --- | --- |
| **Design documentation (30%)** |  |
| Use case diagram | (**5%**) |
| Sequence diagram | (**5%**) |
| Class diagram – this should encapsulate ALL functionality of the application | (**10%**) |
| Documentation for implementation details and experimental results | (**10%**) |
|  | **(30%)** |
| **Implementation (70%)** |  |
| Implementation of classes identified in the class diagram. Marks are awarded for completeness (properties and methods) and correctness (methods working correctly, correct use of data types to model associations for example). | (**20%**) |
| Appropriate use of OOP concepts within the design (i.e., use of inheritance, polymorphism, abstraction and encapsulation), and appropriate use of Java features such as interfaces and generics etc. | (**15%**) |
| Appropriate implementations for exception handling, visibility of system status (e.g., progress bar to display the time left in prediction) | (**5%**) |
| Implementation of a suitable user interface for drawing a digit using mouse, and uploading an external image | (**10%**) |
| Implementation of k-NN algorithm for computing the Euclidean distance between training images and input handwritten digit image | (**5%**) |
| Appropriate use of Threads for performance enhancement when computing the Euclidean distance | (**5%**) |
| Use of best coding practices. i.e., use of appropriate variable, function names, consistence naming conventions, suitable documentation, commenting of the code and version control. | (**10%**) |
|  | **(70%)** |

1. **Submission details**

**Part I (70%):**

Upload your submissions to the link provided in Moodle, no later than the hand-in date, shown on the from sheet of the assignment. Your submission can be a zip file. Please name your zip file with your student number (e.g., st12345678.zip). Your zip file should contain your implementation and design documentation. The report should also contain a link your version controlling repository (e.g., GitHub, bitbucket), with proper access being granted for module tutors for evaluation.

**Part II: Individual’s Worksheets Portfolio (30%)**

* **Worksheets portfolio:** For the assignment you need to work on worksheets from week 7-11 about various topics on industry standard programming and design in the software development life cycle. All students need to submit **worksheets and related code in a zip folder**. The purposes of these weekly worksheets is to explain the key issues in modeling and analysing real world problems using appropriate techniques. Further details can be found from Moodle.

1. **Referencing requirements**

All referencing should use the Harvard referencing style. **Any images used in the implementation should also be cited and included in an appendix in the design document.**

1. **Marking scheme**

|  |  |
| --- | --- |
| **Mark range** | **criteria** |
| **70 – 100** | An excellent design is given that covers all the requirements of the assignment (use case 1 and 2) and correctly uses UML notation throughout. An excellent implementation is given that is both efficient (multi-threaded implementation) and implements all aspects of the design. No errors are evident in the code and additional elements have been included to good effect. A very detailed analysis / discussion of the results obtained is given that discusses why the results were obtained, demonstrating a very good understanding of the code. |
| **60 – 69** | A very good design is given that covers all the requirements of the assignment, though some notational errors are evident. A good implementation is given that closely reflects the given design. Few to no errors are evident in the code but some aspects could be implemented more efficiently. A good analysis / discussion of the results obtained is given that discusses to some extent why the given results were obtained, demonstrating a good understanding of the code. |
| 1. **– 59** | A good design is given that covers the key requirements of the assignment but this could be expanded. A good implementation is given that closely reflects the design, but a number of omissions and errors are evident in the code. The implemented code could also be more efficient. Some analysis of the results is given, but this is largely descriptive and needs to be expanded. |
| 1. **– 49** | A basic design is given but does not address all of the requirements of the assignment. A basic implementation is given but this does not implement all aspects of the design and contains a number of errors and inefficiencies throughout. This needs to be significantly expanded. Some analysis of the results obtained is given but this is descriptive in nature. More discussion of *why* the given results were obtained is needed. |
| 1. **– 39** | A very basic design is given that does not meet all of the requirements of the assignment and does not convey enough information to implement the solution. A basic implementation is given, but this does not implement all of the required features and contains numerous errors and inefficiencies throughout. Only a very basic analysis of the results is given. |
| **Under 35** | No meaningful design has been given and little to no code has been implemented. There is no evidence of any meaningful testing or analysis of the results obtained. Little to no understanding of the problem and the techniques required to create a solution are evident. |

1. http://yann.lecun.com/exdb/mnist/ [↑](#footnote-ref-1)