|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Student Name | | Ivon Harris | Student Number | 472832219 |
| Unit Code/s & Name/s | | ICTPRG418 Apply intermediate programming skills in another language | | |
| Assessment Type | | Written | | |
| Assessment Name | | **Written Assignment**  Programming Assignment 1 | Assessment Task No. | AT1 |
| Assessment Due Date | | Thu 24-Feb-2022 (Week 5) | Date submitted | 02/03/2022 |
| Assessor Name | |  | | |
| **Student Declaration:** I declare that this assessment is my own work. Any ideas and comments made by other people have been acknowledged as references. I understand that if this statement is found to be false, it will be regarded as misconduct and will be subject to disciplinary action as outlined in the TAFE Queensland Student Rules. I understand that by emailing or submitting this assessment electronically, I agree to this Declaration in lieu of a written signature. | | | | |
| Student Signature | Ivon Harris | | Date | 02/03/2022 |
| **PRIVACY DISCLAIMER:** TAFE Queensland is collecting your personal information for assessment purposes. The information will only be accessed by authorised employees of TAFE Queensland. Some of this information may be given to the Australian Skills Quality Authority (ASQA) or its successor and/or TAFE Queensland for audit and/or reporting purposes. Your information will not be given to any other person or agency unless you have given us written permission or we are required by law. | | | | |

|  |  |
| --- | --- |
| Instructions to Student | **Learning Support**  Additional support is available to help you achieve your learning goals. Speak to your teacher or a Learning Skills Centre team member if you feel that you may benefit from some extra support. The Institute provides extra support through the Disability Support Unit and the Learning Skills Centre.  RPL (Recognition of Prior Learning) is available for this unit. Speak to your teacher/assessor to check if you qualify for RPL.  **Conditions of Assessment**  You will need to complete the learning and undertake all assessments satisfactorily to be deemed competent. You are responsible for complying with all assessment item instructions; submission and collection requirements; undertaking assessment tasks honestly and retaining a copy of all assessment items.  You must submit assessment items by the **due date**, unless an extension has been granted by your teacher. Failure to submit assessment items by the due date will result in a “did not submit” being recorded and depending on your circumstances, you may be granted one final resubmission.  To be judged competent in this assessment item the student is required to demonstrate competence in all indicators shown in the marking guide.  **The Classroom as a Simulated Work Environment**  Students must be aware and take responsibility for the problems of working in a shared IT environment. Problems such as noise levels, production flow, interruptions and time variances are common to workplaces. In the simulated environment provided in the classroom these problems can take the form of:   * Other students who continually ask questions or talk aloud while thinking * Fire drills, projector not working, printers running out of paper or toner cartridge * Miscalculating how much work you can do in one day, missing classes and so on.   Some things are unavoidable and you must devise strategies to overcome them, for example, we cannot stop students from asking questions or entering at exiting the class. Other things are unpredictable (e.g. fire drills). You need to be aware and plan and organise your work allowing some extra time for unavoidable and unpredicted events. |
|  | **Assessment Criteria:**  To achieve a satisfactory result, your assessor will be looking for your ability to demonstrate key skills/tasks/knowledge to an acceptable industry standard.  Refer to the marking criteria document for a detailed list of items.  **Number of Attempts:**  You will receive up to two (2) attempts at this assessment task. Should your 1st attempt be unsatisfactory (U), your teacher will provide feedback and discuss the relevant sections / questions with you and will arrange a due date for the submission of your 2nd attempt. If your 2nd submission is unsatisfactory (U), or you fail to submit a 2nd attempt, you will receive an overall unsatisfactory result for this assessment task. Only one re-assessment attempt may be granted for each assessment task, with the exception of Apprentices or Trainees who are permitted an additional supplementary assessment. **For more information, refer to the Student Rules.** |
| Submission details | Submit your assessment to the allocated dropbox in **Connect** or to the allocated network folder.  Your teacher will provide all the details for the submission system or network.  What to submit:You are to submit a zip archive file containing the following to CONNECT (ICTPRG418)   1. Visual Studio project files (including source code, .csv data file) Source code files must be written and commented according to the provided software 2. README.md file for software project based on the readme template provided 3. XML-generated document displaying (minimum) all methods and classes – name this file: BirthdayTracker.xml 4. Standard MIT license for your software project containing copyright statement (current year and your full name) – name this file: LICENSE.txt 5. Completed documentation.docx file (containing completed test plan and description/evidence of your debugging).   Your zip archive assignment file must be saved with your surname\_student number\_unit/cluster\_AssessmentNumber. For example:  **surname\_1234567890\_ICTPRG418\_1**  For re-submissions, an “R” must be added to the file name. For example:  **surname\_1234567890\_ ICTPRG418\_1\_R**  The Marking Criteria Sheet must be signed and submitted with your work. |
| Instructions for the Assessor | To be judged competent in this assessment item the student is required to demonstrate competence in all indicators shown in the marking guide.  The assessment must be conducted in safe conditions replicating the workplace. Noise levels, production flow, interruptions and time variances must be typical of those experienced in the programming and software development field of work and include access to:   * a software development environment * the technical requirements as outlined in the task instructions |
| Note to Student | An overview of all Assessment Tasks relevant to this unit is located in the Unit Study Guide. |

# Instructions to Students

**‘My-Friend Tracker’ Software Development Assignment**

Case Study Script and Requirements

Sasha has asked you to prepare a small Windows application that might be used for **tracking friends, clients and family names, likes, dislikes and birth-dates**.

The aim is to provide Sasha with a listing per month of upcoming birthdays including details as to individual’s likes and dislikes - assisting with the selection of presents.

The application will allow Sasha to:

Enter new person records - names, likes, dislikes, day-of-birth and month-of-birth.

Use various navigation buttons:

* First, last, next and previous record buttons
* A Find button to search for an existing record by person name

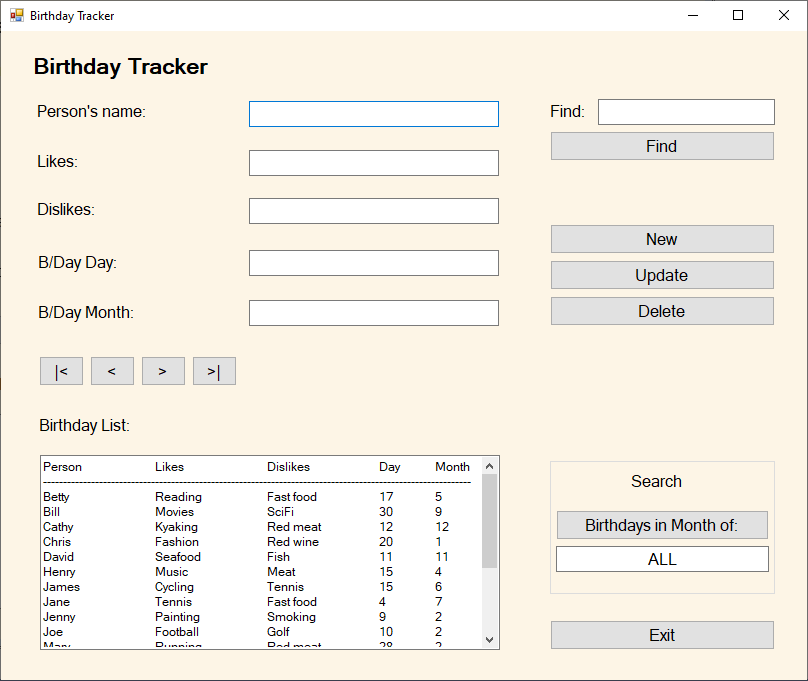
Use New (to add a new record), Update (to update an existing record) and Delete (to delete an existing record) buttons.

View birthdays for a selected month.

Use a search method for finding an existing record by person name, within the sorted list, using a binary search.

The application will save and load the people data to and from a comma-delimited csv file: **MyFriendData.csv**.

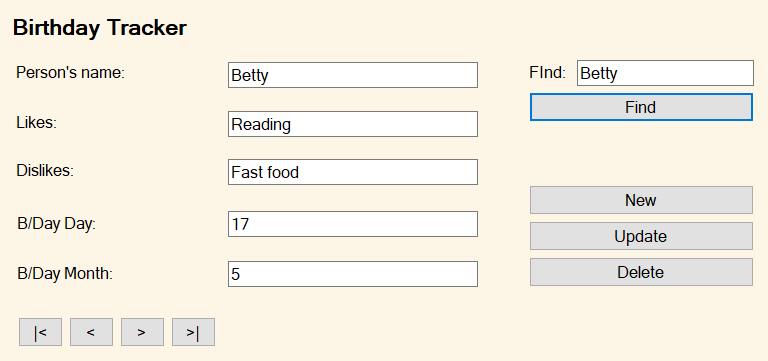
You are to use Visual Studio (C# .NET framework) to build your GUI form application similar to the screen-shot example below:



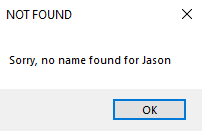
When the application is launched, the text field below the BIRTHDAYS IN MONTH OF button should display the text “ALL”. When the button is clicked, the text field cycles the month number and abbreviated month – for example: “1 - Jan”, “2 - Feb”, “3 - Mar” etc.

Also, when launched, the application reads the data - stored in a comma-separated format in an external file (“MyFriendData.csv”) and displays the data in the BIRTHDAY LIST text box on the bottom left – refer above screen-shot example.

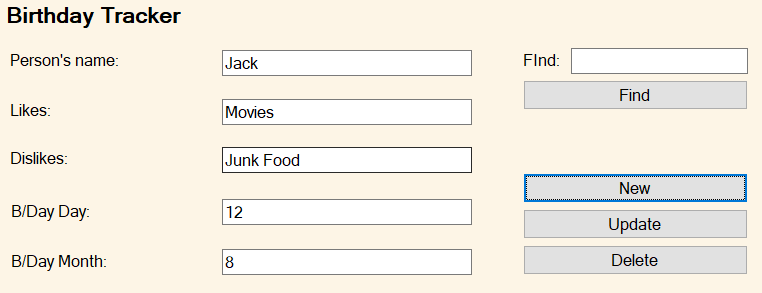
The application should allow the user to enter a name to search. If the search name exists in the records, then details are displayed in the text box fields. Navigation buttons below should display previous and next records from that of the currently viewed record. The far left navigation button should display the first record and the far right button should display the last record.



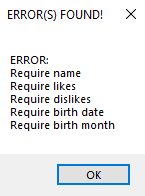
If the search name does not exist in the records, then an appropriate message box should be displayed.



The application should allow the user to enter a new record (with all fields correctly entered)

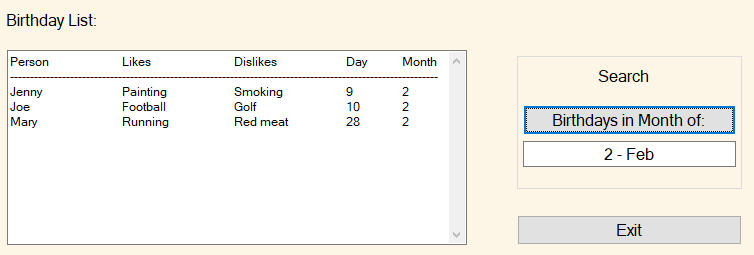


If the fields are empty and/or contain incorrect data (e.g. day and month values are non-numeric or the numbers are incorrect – for example showing B/Day Day value as 31 for the month of 4 [April]), then an appropriate error message box should be displayed.

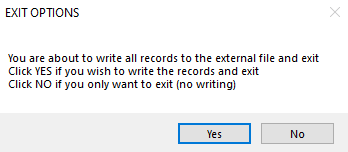


In a similar manner, any attempted updates with problematic text fields will also display an error message. A record to be deleted should be displayed in its entirety for proper validation prior to being deleted.

Listings can be displayed by month – for example, when clicking the BIRTHDAYS IN THE MONTH OF button, the month number and month text appear in the text box below the button. In addition, the display of all records with the selected month will appear in the BIRTHDAY LIST text box on the bottom left. Refer to February example in the screen shot below.



The EXIT button should allow for the writing of all current records held in the program (whether existing, new, updated or deleted) before closing the application.



**Your tasks:**

Sasha requires this application to:

|  |  |  |
| --- | --- | --- |
| 1.1 | Have appropriate **design and technical documentation**, a **test plan**, **test cases** plus the **results** of testing. The source code is to be supplied and should contain appropriate in-line comments. Your programming should follow your development team’s programming standards. | DONE |
| 1.2 | **Load** all the current person records from the **MyFriendData.csv** data file on the hard drive (or equivalent) when the program is first launched. | DONE |
| 1.3 | **Find** an existing person**; Add** new person records; **Update** existing person records - names, likes, dislikes, day-of-birth and month-of-birth; and **Delete** an existing person record. | DONE |
| 1.4 | Provide **navigation buttons** and a **search** facility to navigate through the various person records. | DONE |
| 1.5 | Allow the program user to **save** all the person records back to the hard drive (or equivalent) when the program is closed. | DONE |
| 1.6 | Display a **list** of birthdays (person’s name, likes, dislikes, birth-date and birth-month) for a selected month. The displayed list should be sorted by name in alphabetical order (a-z). The display should also allow for the display of all listed people in the data record. | DONE |
| 1.7 | Be provided with a basic supporting **user manual** appropriately considered and structured with correct spelling and grammar. | DONE |

Additionally, you are required to:

|  |  |  |
| --- | --- | --- |
| 1.8 | **Review your internal and external documentation** for your application. Ensure that it aligns with organisational documentation standards provided, contains correct grammar and spelling, and is appropriate for developers who may provide support to this application in the future. | DONE |
| 1.9 | Prepare **program documentation** using a facility provided within your IDE. For Visual Studio C# .NET framework, this typically means utilising the XML documentation that can be created from within the source code files. A README.md file is also required based on the provided template. | DONE |
| 1.10 | Provide evidence of your use of the **debugging facilities** with the Integrated Development Environment (IDE) you are utilising. This may be demonstrated with a series of screen images of your debugging of the application within your selected IDE – showing at least one breakpoint, a set of associated watches, and your tracing through several lines of code. |  |