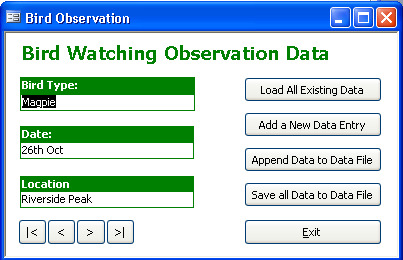
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Student Name | Kyle Kent | | Student Number | | 465510139 |
| Unit Code/s & Name/s | ICTPRG418 Apply intermediate programming skills in another language | | | | |
| Assessment Name | Class Test | | Assessment Task No. | | AT3 |
| Date of test/exam | 10/8/2018 | | | | |
| **Student Declaration:**  I declare that this assessment is my own work. I am aware of and understand the rules related to assessment as outlined in TAFE Queensland Student Rules and acknowledge that failure to comply with these rules will be regarded as misconduct and will be subject to disciplinary action as outlined. | | | | | |
| Student Signature | K Kent | | | Date | 10/09/2018 |
| Assessor Feedback:  Student provided with feedback *(check box when completed)* | | | | | |
| Attempt 1 | Satisfactory | Unsatisfactory | | Date | / / |
| Attempt 2 | Satisfactory | Unsatisfactory | | Date | / / |
| Assessor Name | David Hunt | Assessor Signature | |  | |
| Note to assessor: Please record any reasonable adjustment below that has occurred during this assessment. E.g. written assessment given orally; scribe provided. | | | | | |
|  | | | | | |
| **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. | | | | | |

|  |  |
| --- | --- |
| Assessment rules | Only students enrolled in the unit of competency, the Assessor / Supervisor, and other authorised personnel may enter or remain in the room during a written test / exam. The Assessor/ Supervisor may ask you to produce photographic identification (e.g. student ID card, driver’s licence).  Unless approved by the Assessor / Supervisor prior to the written test / exam (e.g. for open-book exams) you may not bring any devices capable of conveying information relevant to the content (e.g. text books, course notes, mobile phones, pagers, notebook computers, and other devices). You must ensure mobile phones and other electronic devices are turned off prior to the commencement of the written test / exam.  You are required to comply with all directions:   1. Detailed in assessment material supplied; 2. Set out on any notice displayed in the room; and 3. Given by the supervisor.   During a written test / exam session you may not:   1. Communicate with any person other than the supervisor; 2. Assist another person to communicate with another person; and 3. Willingly receive communication from any person except with the approval of the supervisor.   Unless permitted by the supervisor, you may not take from the room any papers or other materials provided for use during the written test / exam.  You are expected to be considerate of other students when entering or leaving the room or when in the vicinity of the room.  If you consider that your performance in the written test / exam has been adversely affected by illness, disability, bereavement or other exceptional circumstances you may apply for special consideration. **For more information, refer to the Student Rules.** |
| Instructions to Student | **Number of Questions:** 15  **Time Allowed:** 4 hours  **Examination Conditions:**  This is a closed book examination; All questions must be attempted. |

|  |  |
| --- | --- |
|  | **Materials to be supplied:**  Examination paper  **Materials to be supplied by the Student:**  Paper for recording answers  **General Instructions:**  You are required to answer each of the questions provided. You must use a black or blue pen to provide answers, not pencil. Sketches, however, may be in pencil.  **Calculators:**  Calculators may be used during this examination. Before the examination commences, all memories must be fully cleared and programs erased.  **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 questions with you and will arrange a date your 2nd attempt. If your 2nd attempt is unsatisfactory (U), or you fail to attend the scheduled date for 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.** |
| Instructions for the Assessor | This is a closed book examination.Remind students of TAFE Queensland Student Rules assessment requirements. |
| Submission details  (if relevant) | Students are to submit written answers to the teacher at the end of the exam. |
| Note to Student | An overview of all Assessment Tasks relevant to this unit is located in the Unit Study Guide. |

You have been asked to create a simple program that allows the user to enter birds that they sight, the date and location. On pressing the button: Append Data to Data File, the program copies the three entries to a new line in the data file which is later read into a spreadsheet program for analysis.



Java screenshot used with permission from [Oracle](https://www.oracle.com/legal/copyright.html).

In relation to this program, you are required to answer the following questions:

**Introductory**

* 1. Write down a portion of code that would be used to save a line of Bird data.

string bird = txtBirdType.Text.ToString();

string date = txtDate.Text.ToString();

string location = txtLocation.Text.ToString();

ArrayList birdArrList = new ArrayList();

GetData Bird = new GetData(typeOfBird, dateOfBird, locationOfBird);

birdArrList.Add(Bird);

* 1. Explain in detail where the principles of Modularisation (Cohesion and Coupling) and/or Object Oriented Programming could be applied in this program.

Modularisation can be applied throughout this program. As the program is a form, most of the code within the form would simply be calls to other classes and methods. All buttons, including navigation buttons could be separated into classes with their own methods. From there the classes can simply be called when the user clicks the appropriate button. This applies the OOP concept of low coupling to ensure cohesion in the projects code.

1.3 In searching for an error in the saving routine, how might you use your IDE’s debugging tool?

First, you would start program in debugging mode. From there you can set breakpoints and watches. Set breakpoints just after the array or list that is meant to house the data being saved and place a watch on the array or list to see if it is being passed the information. If the data is being saved the watch should show the values being passed, if it is not then there’s a problem in the logic passing the data that needs to be fixed.

1.4 What Java utility could be used to create detailed technical program documentation?

Utilities such as, Antominor pro or JavaDoc could be used for program documentation. What Antominor and JavaDoc do is document the entire class for you. Adding comments to the class, detailing parameters and describing what variables and methods do.

1.5 Prepare a test procedure and/or plan for testing this program.

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Actual Result | Expected Result | Reasons |
| First/Prev/Next/Last buttons | Last falls off index | First/Prev/Next/Last of Bird Observations | Program error-checking |
| Add new data entry | Exception error if text boxes unfilled | Add new bird entry based off text inside text boxes | Program error-checking |
| SaveData() | Logical error with write | Save all Bird observations to csv file | Program error-checking |
| GetData() | Correct | All txt.Text grabbed and stored | Program error-checking |
| DeleteData() | Correct | Delete selected bird observation | Program error checking |
| AppendData() | Writes, not appends | Append selected bird observation to csv | Program error checking |
| NewData() | Fails to create new observation | Create new bird observation | Program error checking |

**Intermediate**

2.1 In relation to objects and arrays:

* Draw a class diagram for a BirdWatchingObservation class that would hold one record (entry) of the data for the Bird Watching Observation program. Add at least 3 properties and 6 methods to this diagram.

Methods:

GetData(string birdName, string birdDate, string birdLocation)

DeleteData(string birdName, string birdDate, string birdLocation)

Position(int indext)

AppendData(string birdName, string birdDate, string birdLocation)

LoadData(Array birds)

SaveAllData(Array birds)

NewData(string birdName, string birdDate, string birdLocation)

Properties:

string birdType

string birdDate

string birdLocation

BirdWatchingObservation

* Show how you would declare an array of these Bird Watching Observation Records.

ArrayList birdArrList = new ArrayList();

2.2 In relation to the array of Bird Watching Observation records:

* Write full code or pseudo code for a Delete method that deletes a Bird Watching Observation record. This method should demonstrate appropriate Low Coupling and High Cohesion.

string bird = txtBirdType.Text.ToString();

string date = txtDate.Text.ToString();

string location = txtLocation.Text.ToString();

typeOfBird = bird;

dateOfBird = date;

locationOfBird = location;

DeleteData(typeOfBird, dateOfBird, locationOfBird);

{

foreach(GetData bird in form1.birdArrList)

{

Var birdName = bird.Bird

Var birdDate = bird.Date;

Var birdLocation = bird.Location;

If(bridName == typeOfBird

&& birdDate == dateOfBird && birdLocation == locationOfBird)

{

form1.birdArrList.Remove(bird);

}

}

}

* Write a line of code for sorting a single array of Bird Types.

Array.Sort(birdTypes);

* Explain or give example code for how a binary search could be applied to this sorted array of Bird Types.

string typeOfBird = txtBirdType.Text.ToString();

birdTypes.BinarySearchOf(typeOfBird);

* Explain or give example code for how this sorted array might be saved to a Random Access File.

string filePath = “FileLocation.txt”

StringBuilder br = new StringBuilder();

Foreach(GetData bird in form1.birdArrList)

{

GetData.Bird = string typeOfBird;

GetData.Date = string typeOfDate;

GetData.Location = string typeOfLocation;

var format = br.format(“{0}, {1}, {2}”, typeOfBird, typeOfDate, typeOfLocation).ToString();

}

File.WriteToFile(filePath, format);

2.3. Write a line of code that declares and instantiates a 2D array to store 100 entries of Bird Watching data.

string [][] birdWatching = new string [][]

2.4 Outline how 2 functions of your IDE’s debugging tool assist you with your programming?

Watches allow the programmer to view the value of variables, which can be incredibly useful when trying to find logical errors.

Breakpoints allow the programmer to halt the program when it hits a certain line of code, in conjunction with watches this allows the programmer to review values before jumping to the next line of code, to see what happens to the values.

2.5 In preparing a detailed set of documentation standards for a team of developers name four (4) sections or topics that would need to be covered

* Methodology
* Coding Conventions and Standards
* Diagrams (FlowChart, UML classes, ER Diagrams)
* Test Plans
* Source Control

2.6 Outline 4 lines of well-considered test data this program. (An example might be: Magpie – 26th Oct – Riverside Peak). Explain why you would use each line.

txtBird.Text.ToString() (eg. Pigeon)

txtLocation.Text.ToString() (eg. Urban environments)

txtDate.Text.ToString() (eg. 12th Oct)

All these lines are required for the program to run as intended. If these lines are wrong or failing to work as intended, then it essential to the program that they run correctly. Without them none of the programs features would work.

SaveAllData(Array Birds) (eg. Array of all bird entries. Eg. A single entry contains Bird, location and date)

SaveAllData is a method that uses the above lines to save all added bird entries. The point of this application is void is you can’t save the information. Making this a requirement.

2.7 Present a pseudo code listing for a method that saves all the Bird Watching Observation data to the data file.

string filePath = “FileLocation.csv”

StringBuilder br = new StringBuilder();

Foreach(GetData bird in Main.birdObservations)

{

GetData.birdType = string typeOfBird;

GetData.birdDate = string typeOfDate;

GetData.birdLocation = string typeOfLocation;

var format = br.format(“{0}, {1}, {2}”, typeOfBird, typeOfDate, typeOfLocation).ToString();

}

File.WriteToFile(filePath, format);

**Knowledge Evidence:**

2.8 Answer the following in detail:

1. Using an example, describe static and a dynamic variable and explain the difference between them.

A static variable is called at compilation and is given the value it is assigned at compilation. Dynamic is the opposite. A dynamic variable is called at runtime and its value can shift. Since a dynamic variable is called at runtime and not compilation the IDE cannot see the variable and as such, you cannot assign it properties or values.

A static can be given properties: string static =””;

static = “abc”;

If you tried to do this with a dynamic variable. The IDE would not know what the string “static” is, resulting in the strings value not changing to “abc”;

1. Describe with an example, a medium-size application. What strategies would you utilise to develop a medium-size application?

The first step would be considering a development methodology. Once a methodology was agreed upon, the next step would be writing documentation based upon that methodologies’ standards. Another required step is source control. Before building the project, source control must be put in place, to ensure programmers can easily backtrack to working versions. Backup strategies must also be in place. While source control can work as backups, copies should be saved to servers to guarantee if one copy is lost, no progress is lost.

1. Describe each, with an example, a standard array structure, and standard algorithms for data file handling.

A standard array structure looks like a list of items. It holds a list of items that the programmer inputs. When viewing the contents of an array it would look similar to this:

This is an array of bird observations:

Pigeon, Urban Environments, 12th Oct

Magpie, Urban Environments, 19th Oct

Crow, Urban Environments, 22nd Oct

Standard algorithms for data file handling allow the programmer to manipulate data and read from and write to outside files, such as text files or csv files. Standard algorithms include:

File.WriteToFile()

File.ReadFromFile()

File.AppendFile()

BinaryReader()

StreamReader()

1. Describe with two (2) examples, user-defined data structures.

One example of a user defined data structure is classes. Everything within a class is defined by the user. From variables to methods, all of it is designed by the programmer.

Another would be a method, as just like a class everything within a method is defined by the programmer

2.9 List two (2) development methodologies and their application

One development is the Waterfall methodology. The water fall methodology is rather rigid in its application. A benefit of this is having most of the project planned out in great detail. While this can be considered a good thing, it can also be considered a bad thing. If plans change and project needs to shift in some way, the waterfall methodology can completely halt any progress on the project, due to the entire methodology needing readjustment. A good alternative to the Waterfall methodology is the Agile methodology. The Agile methodology is the exact opposite of the Waterfall methodology, where the Waterfall methodology works well with static projects, the Agile methodology works well with dynamic projects. An Agile methodology provides a broad overview of the project, which allows for easy adjustment in case something does need to be changed.

2.10 What are the main features of programming languages?

Abstraction: Extracts values and uses them in another method, variable etc.

Encapsulation: Encapsulating objects variables and methods so they can only be used within a certain context or hidden from view.

Polymorphism: A copy of an object, method or variable with different functions.

Inheritance: Inherit values or properties from other variables or objects

**End of assessment**