THE INDEPENDENT INSTITUTE OF EDUCATION

MODULE NAME:	MODULE CODE:
PROGRAMMING 1B	PROG6112

ASSESSMENT TYPE:	**TEST TO BE COMPLETED ELECTRONICALLY**
TOTAL MARK ALLOCATION:	**60 MARKS**
TOTAL TIME:	**The time given to students to complete this assessment will be indicated on your module in *Learn***

By submitting this assessment, you acknowledge that you have read and understood all the rules as per the terms in the registration contract, in particular the assignment and assessment rules in The IIE Assessment Strategy and Policy (IIE009), the intellectual integrity and plagiarism rules in the Intellectual Integrity Policy (IIE023), as well as any rules and regulations published in the student portal.

INSTRUCTIONS:

1. *Please **adhere to all instructions**. These instructions are different from what is normally present, so take time to go through these carefully.*
2. ***Independent work is required**. Students are not allowed to work together on this assessment. Any contraventions of this will be handled as per disciplinary procedures in The IIE policy.*
3. ***No material may be copied from original sources, even if referenced correctly, unless it is a direct quote indicated with quotation marks.***
4. *All work must be adequately and correctly referenced.*
5. *You should paraphrase (use your own words) the concepts that you are referencing, rather than quoting directly.*
6. *This is an open-book assessment.*
7. *Assessments must be typed unless otherwise specified.*
8. ***Ensure that you save a copy of your responses.***
   a. *Complete your responses in a Word document.*
   b. *The document name must be your **Name.Student number. Module Code**.*
   c. *Once completed the assessment, upload your document under the **submission link** in the correct module in Learn.*

Additional instructions:

1. *Calculators are allowed.*
2. *Answer All Questions.*
3. *Show all calculations, where applicable (marks may be awarded for this).*
4. *Instructions for assessments including practical computer work:*
   - *Use of good programming practice and comments in code is compulsory.*
   - *Save your application in the location indicated by the administrator (e.g. the Z:\ drive or your local drive).*
   - *Create a folder as follows: use the module code and your own student number and create a folder with a folder name as per the format shown here:*

© The Independent Institute of Education (Pty) Ltd 2020

- ***StudentNumber_ModuleCode_Test****. Save all files (including any source code files, template files, design files, image files, text files, database files, etc.) within this folder.*
- *E.g. if your student number is 12345, and you are writing an examination for the module PROG121, create a folder named **12345_Prog121_Test** and use this throughout the session to save all of your files.*
- ***Important:** Upon completion of your assessment, you must save and close all your open files and double click the ExamLog application on your desktop. You must follow the instructions carefully to ensure that the information about the files that you have submitted for this assessment has been logged on the network. Specify the location of your source code on your question paper.*
- *If you do not have access to the software that is required and you are unable to come to campus due to Covid-19 restrictions, then write the Java statements independent of the Java Integrated Development Environment (IDE). Your lecturer will not penalise you on the syntax but logic of the Java statement. It is important for you to know how to write Java statements (skill) and not be dependent on any Java IDE (s).*

© The Independent Institute of Education (Pty) Ltd 2020

Referencing Rubric

Providing evidence based on valid and referenced academic sources is a fundamental educational principle and the cornerstone of high-quality academic work. Hence, The IIE considers it essential to develop the referencing skills of our students in our commitment to achieve high academic standards. Part of achieving these high standards is referencing in a way that is consistent, technically correct and congruent. This is not plagiarism, which is handled differently.

Poor quality formatting in your referencing will result in a penalty **of a maximum of ten percent being deducted from the percentage awarded**, according to the following guidelines. Please note, however, that **evidence of plagiarism in the form of copied or uncited work (not referenced), absent reference lists, or exceptionally poor referencing, may result in action being taken in accordance with The IIE's Intellectual Integrity Policy (0023)**.

Markers are required to provide feedback to students by indicating **(circling/underlining) the information that best describes the student's work.**

Minor technical referencing errors: 5% deduction from the overall percentage – the student's work contains **five or more errors** listed in the minor errors column in the table below.

Major technical referencing errors: 10% deduction from the overall percentage – the student's work contains **five or more errors** listed in the major errors column in the table below**.**

If both minor and major errors are indicated, then 10% only (and not 5% or 15%) is deducted from the overall percentage. The examples provided below are not exhaustive but are provided to illustrate the error.

Required: **Technically correct referencing style**	**Minor errors in technical correctness of referencing style** **Deduct 5% from percentage awarded**	**Major errors in technical correctness of referencing style** **Deduct 10% from percentage awarded**
Consistency • The same referencing format has been used for all in-text references and in the bibliography/reference list.	Minor inconsistencies. • The referencing style is generally consistent, but there are one or two changes in the format of in-text referencing and/or in the bibliography. • For example, page numbers for direct quotes (in-text) have been provided for one source, but not in another instance. Two book chapters (bibliography) have been referenced in the bibliography in two different formats.	Major inconsistencies. • Poor and inconsistent referencing style used in-text and/or in the bibliography/ reference list. • Multiple formats for the same type of referencing have been used. • For example, the format for direct quotes (in-text) and/or book chapters (bibliography/ reference list) is different across multiple instances.
Technical correctness • Referencing format is technically correct throughout the submission. • Position of the reference: a reference is directly associated with every concept or idea. • For example, quotation marks, page numbers, years, etc. are applied correctly, sources in the bibliography/reference list are correctly presented.	**Generally, technically correct with some minor errors.** • The correct referencing format has been consistently used, but there are one or two errors. • Concepts and ideas are typically referenced, but a reference is missing from one small section of the work. • Position of the references: references are only given at the beginning or end of every paragraph. • For example, the student has incorrectly presented direct quotes (in-text) and/or book chapters (bibliography/reference list).	**Technically incorrect.** • The referencing format is incorrect. • Concepts and ideas are typically referenced, but a reference is missing from small sections of the work. • Position of the references: references are only given at the beginning or end of large sections of work. • For example, incorrect author information is provided, no year of publication is provided, quotation marks and/or page numbers for direct quotes missing, page numbers are provided for paraphrased material, the incorrect punctuation is used (in-text); the bibliography/reference list is not in alphabetical order, the incorrect format for a book chapter/journal article is used, information is missing e.g. no place of publication had been provided (bibliography); repeated sources on the reference list.
Congruence between in-text referencing and bibliography/ reference list • All sources are accurately reflected and are all accurately included in the bibliography/ reference list.	**Generally, congruence between the in-text referencing and the bibliography/ reference list with one or two errors.** • There is largely a match between the sources presented in-text and the bibliography. • For example, a source appears in the text, but not in the bibliography/ reference list or vice versa.	**A lack of congruence between the in-text referencing and the bibliography.** • No relationship/several incongruencies between the in-text referencing and the bibliography/reference list. • For example, sources are included in-text, but not in the bibliography and vice versa, a link, rather than the actual reference is provided in the bibliography.
In summary: the recording of references is accurate and complete.	In summary, at least **80%** of the sources are correctly reflected and included in a reference list.	In summary, at least **60%** of the sources are incorrectly reflected and/or not included in reference list.

Overall Feedback about the consistency, technical correctness and congruence between in-text referencing and bibliography:

.....

© The Independent Institute of Education (Pty) Ltd 2020

Question 1 **(Marks: 30)**

Write a Java program to display the three highest monthly speeding fines recorded for four different cities. The following rows and columns represent the monthly speeding fines of each city.

	JAN	FEB	MAR
JHB	128km	135km	139km
DBN	155km	129km	175km
CTN	129km	130km	185km
PE	195km	155km	221km

Using a Two Dimensional array, produce the speeding fines report including the speeding fines statistics. The speeding fines statistics must display the highest and lowest speeding fines in the two dimensional array.

Example of output:

```
**********************************************************************
SPEEDING FINES REPORT
**********************************************************************
                        JAN             FEB             MAR
JHB                     128km           135km           139km
DBN                     155km           129km           175km
CTN                     129km           130km           185km
PE                      195km           155km           221km
**********************************************************************
SPEEDING FINES STATISTICS
**********************************************************************
Maximum speed captured: 221km
Minimum speed captured: 128km
**********************************************************************
```

Marking Guideline	Mark	Examiner	Moderator
Declaration and Population of Two Dimensional Array Excellent: Correct declaration – 8 marks Good: Correct but just a minor error – 5 to 7 marks Developing: Attempted but not correct – 2 to 4 marks Poor: Not attempted – 0 marks	**8**		

© The Independent Institute of Education (Pty) Ltd 2020

Printing of the rows and columns Excellent: Correct printing of rows and columns – 7 marks Good: Correct but just a minor error – 5 to 6 marks Developing: Attempted but not correct – 2 to 4 marks Poor: Not attempted – 0 marks	**7**		
Determining of highest and lowest speeding fines Excellent: Correct highest and lowest speeding fines – 7 marks Good: Correct but just a minor error – 5 to 6 marks Developing: Attempted but not correct – 2 to 4 marks Poor: Not attempted – 0 marks	**7**		
Printing the highest and lowest speeding fines Excellent: Correct printing of highest and lowest speeding fines – 3 marks Good: Correct but just a minor error – 2 marks Developing: Attempted but not correct – 1 mark Poor: Not attempted – 0 marks	**3**		
Report produced as per sample Excellent: Report produced correctly – 3 marks Good: Attempted but not correct – 2 marks Developing: Attempted but not correct – 1 mark Poor: Not attempted – 0 marks	**3**		
Comments	**2**		
TOTAL	**30**		

Question 2 **(Marks: 30)**

Design a console application that will print the fine due for a citizen who was caught speeding in a 120km zone. Make use of an abstract class named Fine that contains variables to store the citizen name, speed and fine payable. Create a constructor that accepts the citizen name and speed as parameters and create get methods for the variables. The Fine class must also contain a method to calculate the fine. If the speed is greater than or equal to 120km, multiply the speed by R10.20, or else no fine if payable.

© The Independent Institute of Education (Pty) Ltd 2020

The Fine class must implement an iFine interface that contains the following:

```
public interface iFine
{
   public void PrintFine();
}
```

Create a subclass called SpeedingFines that extends the Fine class. The SpeedingFines class must contain a constructor to accept the citizen name and speed as parameters. Write code for the PrintFine() method which displays the citizen name, speed and the total fine due.

Finally write a SpeedingFineApplication class to capture the input and instantiate the SpeedingFines class.

Sample output is shown below and you may use the same values to test your application.

Example of input and output:

```
Enter the person name: Joe Bloggs
Enter the speed: 185
********************************
PERSON: Joe Bloggs
SPEED: 185km
FINE PAYABLE: R 1887
********************************
```

Marking Guideline	Mark	Examiner	Moderator
iFine interface class created Excellent: Correct creation of the interface – 6 marks Good: Correct but just a minor error – 4 to 5 marks Developing: Attempted but not correct – 2 to 3 marks Poor: Not attempted – 0 marks	**6**		
Abstract class created with Constructor, Variables, Methods and Input and Output. Implements the Interface. Excellent: Correct abstract class created with constructor, variables, methods and Input and Output – implements the interface– 10 marks	**10**		

© The Independent Institute of Education (Pty) Ltd 2020

Good: Correct but just a minor error – 5 to 9 marks Developing: Attempted but not correct – 2 to 4 marks Poor: Not attempted – 0 marks			
Speeding Fines class created that extends the Fine class Excellent: Correct – the speeding fines class was created and extends the fine class - 7 marks Good: Correct but just a minor error – 4 to 6 marks Developing: Attempted but not correct – 2 to 3 marks Poor: Not attempted – 0 marks	**7**		
Use the Speeding Fines Application class to capture input and instantiate the Speeding Fines class Excellent: Correct – 3 marks Good: Correct but just a minor error – 2 marks Developing: Attempted but not correct – 1 mark Poor: Not attempted – 0 marks	**3**		
Output produced as per sample Excellent: Correct – 3 marks Good: Attempted but not correct – 2 marks Developing: Attempted but not correct – 1 mark Poor: Not attempted – 0 marks	**3**		
Comments	**1**		
TOTAL	**30**		

END OF PAPER

© The Independent Institute of Education (Pty) Ltd 2020