Computer Science 2A

Practical Assignment 01

Assignment date: 2020-02-05

Deadline 2020-02-11 12h00

Marks: 70

This practical assignment must be uploaded to eve.uj.ac.za **before** 2020-02-11 12h00. Late or incorrect submissions **will not be accepted**, and will therefore not be marked. You are **not allowed to collaborate** with any other student.

Good coding practices include a proper coding convention and a good use of documentation. Marks will be deducted if these are not present. Every submission **must** include a batch file.

The reminder page includes details for submission. Please ensure that **ALL** submissions follow the guidelines. The reminder page can be found on the last page of this practical.

This practical aims to familiarise you with the similarities and differences between `C++` and `Java`.

The nation of Dystopia has a lot of problems...most notably, viruses, fires, and heatwaves. You have started your new job at the **Dystopian Emergency Services Council (DESC)**. As a Computer Scientist in the **DESC**, your superiors believe you can assist in modernising the **DESC**'s tools used to manage their response during crisis situations. Your first **DESC** job is to improve an old program used to transmit crisis messages. (Note: The information is sent in an encoded form to make it more difficult for eavedroppers to edit or spoof crisis messages.)

The **DESC** currently uses a `C++` program that encodes all crisis information into an unreadable form, and then converts it back for other **DESC** employees using the program. This program is not viable as the **DESC** needs a program that can be deployed onto any of its available devices. Your job is to *convert* the old `C++` program into a `Java` program for the **DESC** (optionally from the comfort of your desk...).

The **DESC** has provided you with the following to facilitate your development:

- Existing `C++` Code - The complete and working `C++` program in use by the **DESC** (available under the additional files section on eve).
- Examples of how crisis messages can be encoded and decoded.

Your `Java` program must be able to:

- ***Encode*** (Crisis Message to Unreadable Text)
- ***Decode*** (Unreadable Text to Crisis Message)

For example, Crisis Messages can be encoded/decoded as follows:

- `Fire Warburton`: `C5LQABDLUML0W8`
- `Fire Longreach`: `C5LQA6W8ILQDPO`
- `Virus Kintoref`: `J5LMEAZ580WLQ`
- `OQD0BDJQA1D6GABD0QLE`: `HEATWAVE DALY WATERS`

Hints

One of your colleagues in the **DESC** has suggested the use of the following to assist in converting the Crisis Message Converter from C++ to Java:

- **`String`** - read the documentation to understand what can be done with **`Strings`** in `Java`.
- **`String`** ***`charAt(int index)`*** - used to pull a **`char`** out of a **`String`** at a given **`index`**.

Marksheet

1. **`NameConverter`**
   (a) Constructors **[03]**
   (b) Instance Variables **[04]**
   (c) Helper Method **[02]**
   (d) Encode Method **[07]**
   (e) Decode Method **[05]**
2. **`Main`**
   (a) User input **[03]**
   (b) Instance of **`Converter`** **[02]**
   (c) Display **[03]**
3. Coding convention (structure, layout, OO design) **[05]**
4. Commenting **[10]**
5. Correct execution **[26]**

NB

Submissions which **do not compile** will be capped at 40%!

Practical marks are awarded subject to the student's ability to explain the concepts and decisions made in preparing the practical assignment solution. (Inability to explain code = inability to be given marks.)

Execution marks are awarded for a correctly functioning application and not for having related code.

Reminder

Your submission must follow the naming convention below.

```
SURNAME_INITIALS_STUDENTNUMBER_SUBJECTCODE_YEAR_PRACTICALNUMBER
```

Example

Surname	Berners-Lee	**Module Code**	CSC02A2
Initials	TJ	**Current Year**	2020
Student number	209912345	**Practical number**	P01

```
Berners-Lee_TJ_209912345_ CSC02A2_2020_P01
```

Your submission must include the following folders:

Folder	***State***	**Purpose**
`bin`	*Required*	Should be empty at submission but will contain runnable binaries when your submission is compiled.
`docs`	*Required*	Contains the batch file to compile your solution, UML diagrams, and any additional documentation files. Do not include generated JavaDoc. All files must be in **PDF** format. Your details must be included at the top of any **PDF** files submitted.
`src`	*Required*	Contains all relevant source code. Source code must be places in relevant sub-packages! Your details must be included at the top of the source code.
`data`	*Optional*	Contains all data files needed to run your solution.
`lib`	*Optional*	Contains all libraries needed to compile and run your solution.

NB

Every submission **must** include a batch file that contains commands which will:

- compile your Java application source code.
- compile the associated application JavaDoc.
- run the application.

Do not include generated JavaDoc in your submission. All of the classes/methods which were created/updated need to have JavaDoc comments.