GD1P02 **Algorithms and Data structures**	**Summative Project** **Algorithms and Data structures** **Weightage (40%)**	**Date:** **13th May,2019** **Submission Dates:** **9:00am, 10th June,2019**

Submission filename:
YYYY-MM-DD – GD1P02 – Solitaire Game – Student Name.zip

Classic Solitaire:
In this project you will implement the classic Solitaire game as included with Microsoft Windows since 1990.

The goal of the game is to sort a deck of 52 cards in to four stacks of cards, one for each suit, in ascending order, from Ace to King.

The game begins with 28 cards arranged into seven stacks. The first stack contains one card, the second has two cards, and so on. The top card in each column is face up, the rest are face down.

The remaining 24 cards are placed in a deck. You flip 1 to 3 cards from the deck and place them in alternating colour sequential order. The 7 stacks serve as your holders for alternating colour sequential rows. The 4 suits are started with an ace and are added to as the game progresses.

Team-based:
This assignment can be done in groups of 2 or 3. If you are working with a partner, you must declare the tasks assigned to each member via a table of authorship.

Build Quality:
The source code is required to display the following features:

- Compiling code:
  - **Code must build as submitted** in both Debug and Release.
  - No warnings or errors present at Warning Level three for all build targets.
- A folder containing an electronic source code must be included with the submission.
  - Visual Studio 2015 (and above) solution file, project file, and source files are required.
  - Required external game data, such as .ini files (if used).
  - All other files must be removed.

Coding Standards:
The source code is required to adhere to the Media Design School's Game Development Faculty's Coding Standard.

GD1P02 **Algorithms and Data structures**	**Summative Project** **Algorithms and Data structures** **Weightage (40%)**	**Date:** **13th May,2019** **Submission Dates:** **9:00am, 10th June,2019**

Runtime Quality:
The application must not have the following issues:

- Memory leaks.
- Bugs.
- Crashes.

Technical Features:
The game is required to demonstrate the following features:

- Appropriate, effective and correct usage of:
  - C++.
  - Object Oriented Programming.
  - Data structures
  - Algorithms

Interface Features:
The game's user interface should be **graphical (Windows application)**. The executable is required to provide an intuitive interface with the following features:

- Clear instructions are provided both in the game and via an external readme file.
- Controls are clearly identifiable and intuitive while playing.
- The UI design layout must make effective use of screen space.

Release Build Zip:
A release build executable must be zipped and included with the submission. This is equivalent to the final build of the game which is about to be mastered for release. Ensure that project settings are set to Release when creating this build.

GD1P02 **Algorithms and Data structures**	**Summative Project** **Algorithms and Data structures** **Weightage (40%)**	**Date:** **13th May,2019** **Submission Dates:** **9:00am, 10th June,2019**

Submission Checklist:
Source code folder:

- Solution file (.sln).
- Project file (.vcproj).
- Source files (.cpp, .h).
- Library files, if any (.lib).
- **Intermediate files have been removed.**

Release build zip:

- Stand alone executable (.exe) file.
- Readme file (.txt).

The work is to be placed in a zip file and submitted via the "Web Drop Box." The file structure and file names of the submission must follow the file hierarchy listed below. Replace the underlined portions with the appropriate values; italic text identifies the required folders.

- **YYYY-MM-DD – GD1P02 – Solitaire Game – Student Name.zip**
  - **Solitaire – Student Name.zip**
    - ***Source – Student Name***
      - **Solitaire.sln**
      - …Project and source code, etc.