code.sprint MT

TASK BOOKLET

- Qualifiers Round -

SEC 2020

DIRECTORATE FOR LEARNING &
ASSESSMENT PROGRAMMES

Qualifiers Round Schedule

Task 1 Analysis (10 min)
Task 1 Development (40 min)
Task 1 Submission (5 min)
Interval (30 min)
Task 2 Analysis (10 min)
Task 2 Development (50 min)
Task 2 Submission (5 min)

FINALISTS ARE ANNOUNCED!

RIU (40 minutes)

The Rapid Intervention Unit (RIU) is a specialised section of the Malta Police Force. The main objective of the RIU is to have a stronger police presence on the roads.

Whenever a police constable needs to run checks on a certain car, a call is made to the police headquarters. The responder finds the car's number plate on the police system and informs the constable about the name of the car owner and whether the car owner has a valid license and is wanted by the police or not.

This process takes up precious time and often causes delays when quick decisions need to be taken. Therefore, an app needs to be developed and installed on the system in the police car, which lets the user input a number plate and outputs:

- the driver's name, and
- "License not up to date", if the driver's license is not paid, and/or
- "Driver wanted - ARREST", if the driver is wanted, or
- "Driver is clear", if the driver's license is paid and he/she is not wanted, or
- "Number plate not registered", if the number plate is not found, or
- "INVALID Number Plate", if the number plate is not three letters followed by three numbers (XXX000)

The app's database should be populated with the following five (5) sets of data:

NUMBER PLATE	CAR OWNER	LICENSE	WANTED
BOB111	Robert Zarb	Paid	YES
WIZ238	Carl Attard	Paid	NO
XIK764	Jane Schembri	Not Paid	NO
BIL888	Kenneth Refalo	Paid	NO
BON007	Benji Borg	Paid	YES

Name the class containing the main method **RunApp1.**

Submit your program in a folder named **RIU**

Assessment Rubric

Program Functionality	User-Friendly Interface	Proper use of Comments	Proper Conventions (Camel case, meaningful var names etc.)	Name of Folder & Class/es	User Input	Suitable Prompts / Messages displayed
Proper use of Data Structures	Input Validation (Number Plate)	Other Features (Not listed in the task)	**Maximum Score: 20** **+ 2 for every extra feature.**			
0 – Not Satisfactorily | 1- Partly Satisfactorily | 2- Entirely Satisfactorily						

21 Sticks Game (50 minutes)

21-Sticks is a variant of the popular strategy game called 'Nim'. Nim dates to at least the 1500s and possibly earlier. It's very similar to an ancient Chinese game using only two piles of stones called Tsyan-shizi (Picking Stones).

The game play is very simple: two players take turns removing matchsticks from a pile of 21 sticks. The loser is the person who is forced to take the last matchstick.

Develop a program that simulates this game according to the functionalities below:

Functionality #1 – Establishing players

1. Player 1 enters player name.
2. Player 2 enters player name.

Functionality #2 – Gameplay

1. The game starts by showing the number of sticks. Sticks are represented in a series of '*'.
2. Either Player 1 or Player 2 starts the game; choice of player must be randomised.
3. Players take turn by eliminating only one or two sticks at a time.
4. With each player's turn, the number of remaining stick must be shown in a series of '*'.
5. The player who eliminates the last matchstick loses.
6. The game displays the name of the winner.

** A sample screen shot is shown in Figure 1.*

Validation Required

- Players cannot have the same name.
- Players can input only 1 or 2 during game play.
- Avoid runtime error if the player enters a character instead of a number during gamplay.

* Proper messages are displayed if players' input does not abide by the validation rules.

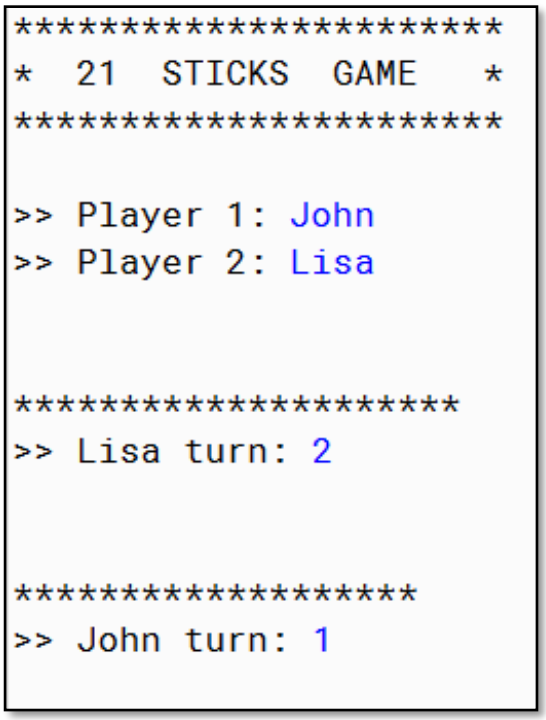

Figure 1: Sample screenshot during gameplay

Name the class containing the main method **RunApp2.**

Submit your program in a folder named **sticksGame**

Assessment Rubric

Program Functionality	User-Friendly Interface	Proper use of Comments	Proper Conventions (Camel case, meaningful var names etc.)	Name of Folder & Class/es	User Input	Suitable Prompts / Messages displayed
Randomisation (Initial Player)	Validation (Player Names)	Validation (Elimination of sticks)	Code Efficiency (avoid duplicate lines of code)	Other Features (Not listed in the task)	**Maximum Score: 24** **+ 2 for every extra feature.**	
0 – Not Satisfactorily | 1- Partly Satisfactorily | 2- Entirely Satisfactorily						

ICE Malta

eSkills Malta Foundation

L-Università ta' Malta
Faculty of Information & Communication Technology

Middlesex University Malta

mita