CODE SPRINT MT

TASKS INFO BOOKLET

Powered by

ICE Malta

Contest Schedule

Start	End	Analysis	Task	Submit	Break	
09:00	09:05	Task 1				PHASE 1
09:05	09:25		Task 1			
09:25	09:30			Task 1		
09:30	09:35	Task 2				
09:35	10:20		Task 2			
10:20	10:25			Task 2		
10:25	**10:40**				**Break 1**	
10:40	10:45	Task 3				PHASE 2
10:45	11:45		Task 3			
11:45	11:50			Task 3		
11:50	**12:10**				**Break 2**	
12:10	12:15	Task 4				PHASE 3
12:15	13:30		Task 4			
13:30	13:45			Task 4		

Task 1 – Automated Teller Machines (20 minutes)

Automated teller machines (ATM) require the user to enter a PIN Number to access the bank card. Write a program that simulates this part of the ATM program.

The program should ask the user to enter a four-numbered PIN code. If it matches the PIN code assigned to the Bank Card, it displays *"CORRECT PIN";* otherwise it displays *"INCORRECT PIN"*. The program should warn the user that the ATM will hold the bank card by displaying *"BANK CARD HELD"* upon entering the wrong PIN code for three (3) consecutive times.

Program Rules:

- PIN assigned to Bank Card is a constant value - 9681.
- If the PIN code entered does not match the proper format (not a 4-numbered pin), the program displays *"INVALID PIN FORMAT"* and does not count the entry as an incorrect PIN.
- It is assumed that only numbers are entered by the user since only numpads are usually available on ATMs

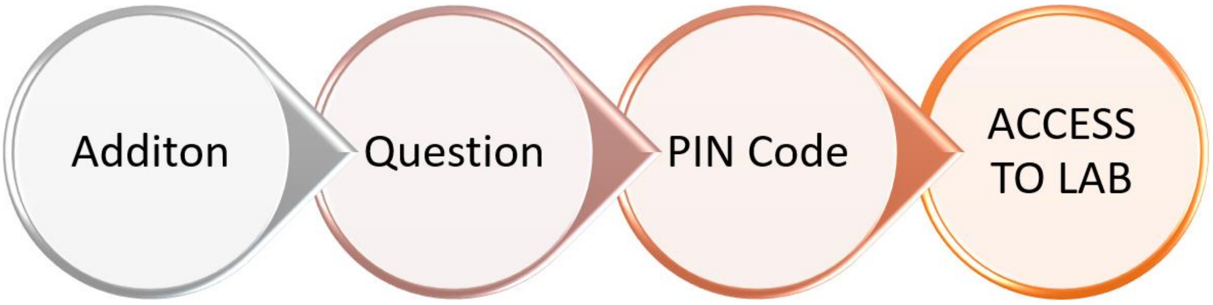
Name the class containing the main method **RunApp1**

Submit your program in a folder called TASK1_INDEXNUM

e.g. **TASK1_0025 or TASK1_0004**

Task 2 – Nuclear Power Plant Access Area (45 minutes)

A scientist can access the Nuclear Powerplant lab by going through a three-tiered validation protocol as in the diagram below:



Step 1: Addition

The scientist must answer correctly a randomly generated addition problem with two numbers. The values to be randomised are between 0 and 9; e.g. 3+7 = ? where 3 and 7 are randomly generated.

Step 2: Question

The scientist must answer correctly a multiple-choice-answer question which is randomised from a list of three pre-set questions. The pre-set questions & answers are listed below (pg. 4); the correct answers are enclosed in a blue box.

Pre-set Questions

Question 1:

Which element is used as fuel in a nuclear power stations?

A: Water	B: Gas	C: Uranium

Question 2:

Which country uses the most nuclear power?

A: The United States	B: Russia	C: France

Question 3:

Which country opened the first nuclear power plant in 1954 known as 'Atom Mirny'?

A: North Korea	B: The Soviet Union	C: Japan

Step 3: PIN Code

The scientist must enter a four-numbered PIN Code which is the constant value 6502.

Log-In Access Flow

From one validation step to another the program will not provide any feedback to the scientist. At the end of the validation process, the scientist will be granted or denied access to the lab by displaying a message accordingly.

Hint:

To ignore case sensitivity of the user's answer (char) of the question displayed, the user input can be changed into its uppercase equivalent. For example, if the user enters character 'b', it can be changed to character 'B'. This can be done using the code: `userInput = Character.toUpperCase(userInput);`

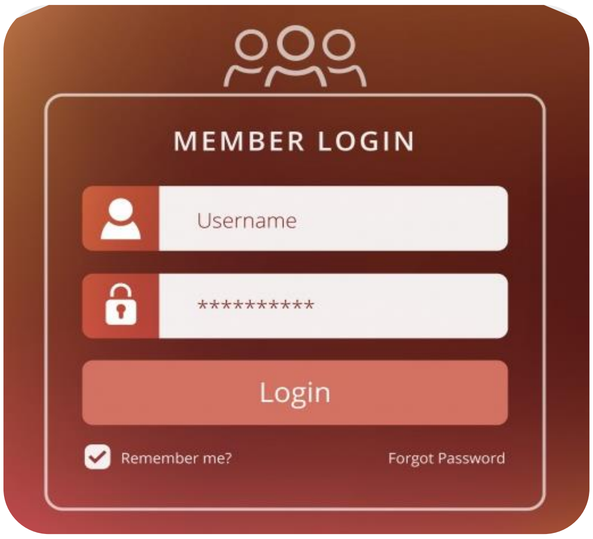
Name the class containing the main method **RunApp2**

Submit your program in a folder called TASK2_INDEXNUM

e.g. **TASK2_0025 or TASK2_0004**

Task 3 – App Log-In (1 hour)

Nowadays, most apps require an authentication process for various reasons, of which security is one important aspect and personalising the user's experience is another.



Write a program which asks the user to enter the username and the password to gain access to the app. The app's database is populated in arrays with the following five (5) usernames:

Username	Password
matik_10	Matik.hello
gta_guru	gta!98
borgsteve	app_in?
alan_1979	mt"ALAN"
dieselmt	rock!N?Roll

Name the class containing the main method RunApp3

Submit your program in a folder called TASK3_INDEXNUM
***e.g.* TASK3_0025 or TASK3_0004**

Program Rules:

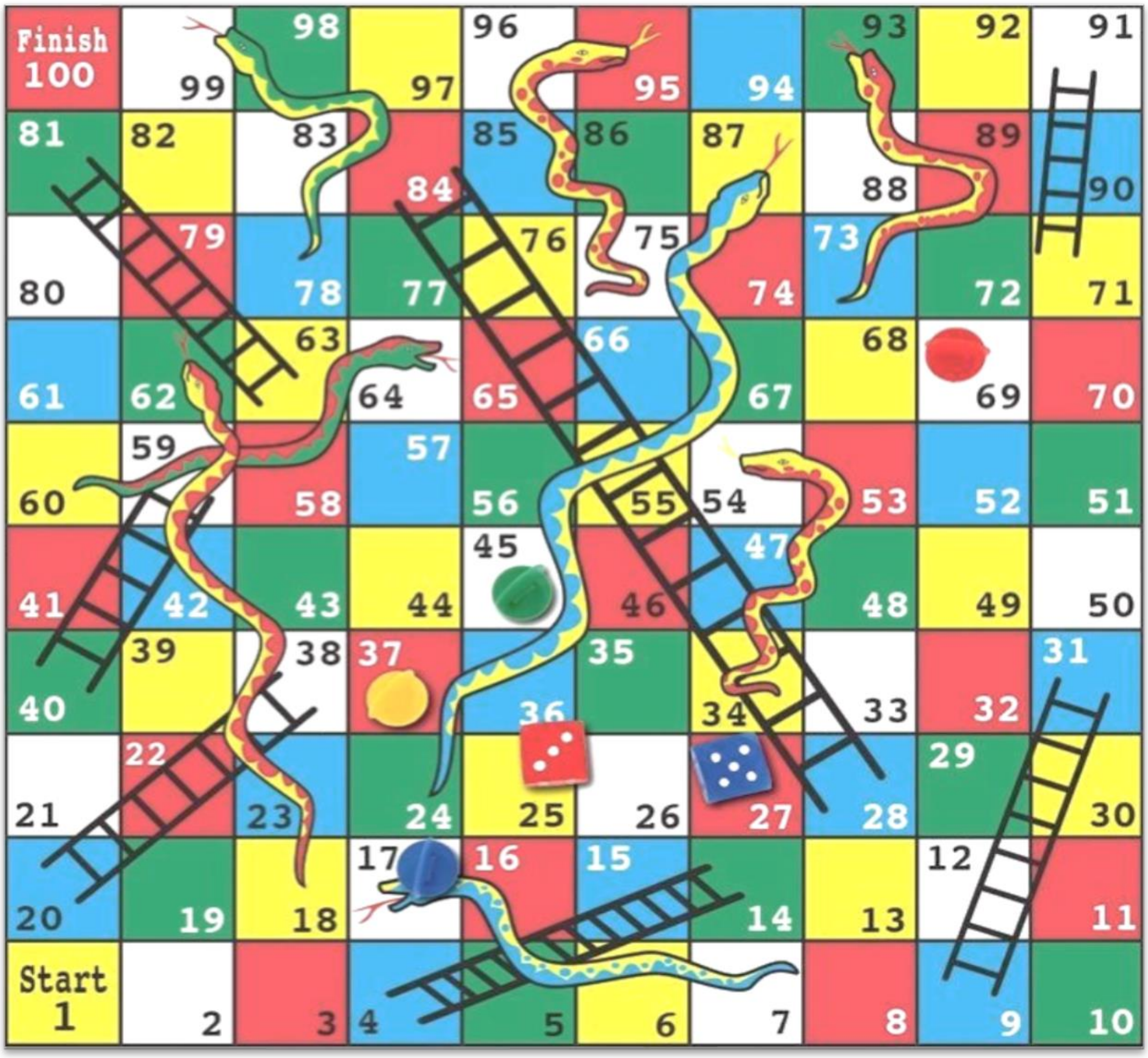
- If the username entered is not found, it displays *"Username does not exist".*
- If the username is found but the password entered is not correct, it displays *"Incorrect Password".*
- If the username is found & password is correct, it displays *"Logged-In Successfully".*
- The username *IS NOT* case sensitive, and the password *IS* case sensitive.

Hint:

1. To ignore case sensitivity of the username (String), the user's input can be changed into its uppercase equivalent. For example, if the user enters username 'borgjoe', it can be changed to 'BORGJOE'. This can be done using the code: `userInput = userInput.toUppercase();`
2. When comparing a String variable in a conditional statement the .equals() method of the String class should be used. For example: `if(userInput == "BORGJOE"){}` will not work because a String variable cannot be compared using ==. The correct version is: `if(userInput.equals("BORGJOE")){}`

Task 4 – Snakes & Ladders (1 hour 15 minutes)

Snakes and Ladders is an ancient Indian board game regarded today as a worldwide classic. It is played between two or more players on a gameboard having numbered, gridded squares starting from 1 to 100. Many "ladders" and "snakes" are pictured on the board, each connecting two specific board squares as shown in the diagram below. The objective of the game is to navigate, according to die rolls, from the start (Step 1) to the finish (Step 100), helped or hindered by ladders and snakes respectively.



The game is a simple race contest based on sheer luck. The historic version had roots in morality lessons, where a player's progression up the board represented a life journey complicated by virtues (ladders) and vices (snakes).

Program Rules:

- The game must be played in TWO (2) players mode.
- Players' name must be entered before the game starts.
- Player 1 starts playing and player 2 follows until one of the players reaches step 100.
- When one of the players reaches step 100, the program displays the name of the winner.
- Player 1 can roll the dice by pressing ' X '.
- Player 2 can roll the dice by pressing ' Z '.
- The dice provides a random number from 1 to 6.
- If the dice gets the maximum value (6), the player HAS AN EXTRA CHANCE to play before the other player continues.
- If the player lands on a step that contains a ladder or a snake, THE PLAYER'S POSITION WILL CHANGE according to the diagram on page 6, or as simplified in the table below.
- During the game PROPER INDICATIONS of the players' status should be displayed.

Player's Position	Element	New Position
4	Ladder	14
9	Ladder	31
17	Snake	7
20	Ladder	38
28	Ladder	84
40	Ladder	59
54	Snake	34
62	Snake	18

Player's Position	Element	New Position
63	Ladder	81
64	Snake	60
71	Ladder	91
87	Snake	24
93	Snake	73
95	Snake	75
99	Snake	78

Name the class containing the main method **RunApp4**

Submit your program in a folder called TASK4_INDEXNUM

e.g. **TASK4_0025 or TASK4_0004**

Assessment Rubric

TASK 1: ATM

Program Functionality	User Friendly Interface	Code Efficiency	Proper use of In-line Text (Comments)	Use of proper Conventions
Name of Folder & Class/es	Constant Declaration & Initialisation	User Input	Suitable Prompts/Messages displayed	Validation of PIN Code (4 Digits only)
Extra Features (not listed in the task)				

0 – Not Satisfactorily | 1- Partly Satisfactorily | 2- Entirely Satisfactorily

Maximum Score: 20 + 2 for every extra feature

TASK 2: Nuclear Powerplant Access Area

Program Functionality	User Friendly Interface	Code Efficiency	Proper use of In-line Text (Comments)	Use of proper Conventions
Name of Folder & Class/es	Constant Declaration & Initialisation	User Input	Suitable Prompts/Messages displayed	Proper Randomisation of Addition Problem
Proper Randomisation of Questions	Proper Validation of User's Answer (A, B or C only)	Proper Validation of PIN Code (4 Digits Only)	Extra Features (not listed in the task)	

0 – Not Satisfactorily | 1- Partly Satisfactorily | 2- Entirely Satisfactorily

Maximum Score: 26 + 2 for every extra feature

TASK 3: App Login Feature

Program Functionality	User Friendly Interface	Code Efficiency	Proper use of In-line Text (Comments)	Use of proper Conventions
Name of Folder & Class/es	Arrays Declaration & Initialisation	User Input	Suitable Prompts/Messages displayed	Ignoring Case Sensitivity for Username
Other Features (not listed in the task)				

0 – Not Satisfactorily | 1- Partly Satisfactorily | 2- Entirely Satisfactorily

Maximum Score: 20 + 2 for every extra feature

TASK 4: Snakes & Ladders

Program Functionality	User Friendly Interface	Code Efficiency	Proper use of In-line Text (Comments)	Use of Proper Conventions
Name of Folder & Class/es	User Input	Suitable Prompts/Messages displayed	Proper Randomisation of the dice	Proper Validation of the User's input to roll the dice (X or Z only)
Ignoring Case Sensitivity for User's Input to Roll the Dice	Other Features (not listed in the task)			

0 – Not Satisfactorily | 1- Partly Satisfactorily | 2- Entirely Satisfactorily

Maximum Score: 22 + 2 for every extra feature