code.sprint MT

Preparatory Workbook

Your starting point towards Codesprint Malta

DIRECTORATE FOR LEARNING &
ASSESSMENT PROGRAMMES

Table of Contents

Welcome to CODESPRINTmt 2
Workbook Guidelines. 3
Grading Distribution 4
Section A 5

Task 1: Nuclear Power Plant Access Area 5
Task 2: Lotto Ticket 7
Task 3: Point-of-Sale System 8
Task 4: Vanishing Creatures (180 minutes) 10

Assessment Criteria 14
Section B 16

Self-Reflection Exercise 16

Final Presentation Guidelines 17

Welcome to CODESPRINTmt

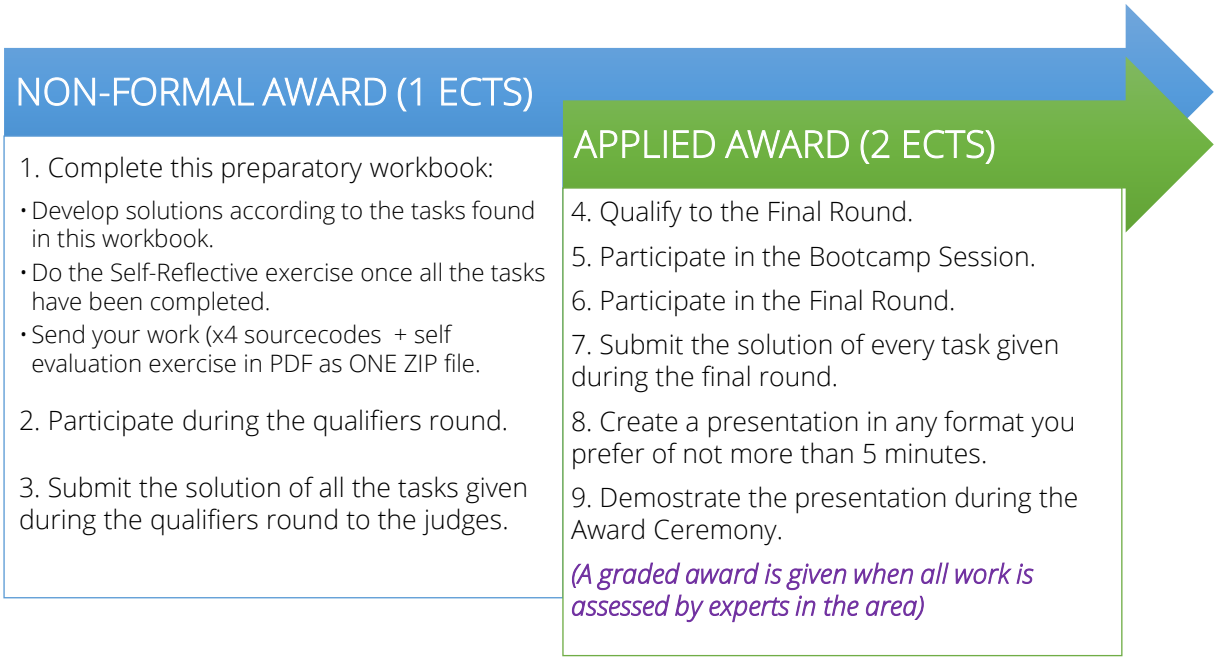
Join code.sprint[MT], a timed coding challenge that puts your problem-solving / coding skills to the test. Besides the coding fun, you will have the opportunity to meet coding enthusiasts and sharpen your coding competitive skills while having the chance to win amazing prizes.

CODESPRINTmt is divided into two phases: 1) Qualifiers Round, and 2) Finals Round. Half the registered participants (maximum of 20 participants) will pass on to the final round.

During both rounds:

- you will be given a series of tasks ordered by level of difficulty.
- every task is timed according to the level of difficulty.
- you are to carry out the tasks on an individual basis.
- you must submit your solutions to the judges at the end of every task.
- a judging panel will give points to your work according to an established criteria rubric.
- a detailed assessment rubric will be given during the contest. Therefore, you will be aware of what the judges are expecting from your work.

Moreover, CODESPRINTmt is accredited at MQF Level 3, thus offering you the opportunity to top up your list of academic achievements. You can be awarded either a **1 ECTS Non-Formal MQF3 Award** (not graded) or a **2 ECTS Applied MQF3 Award** which is graded as explained in section Grading Distribution.

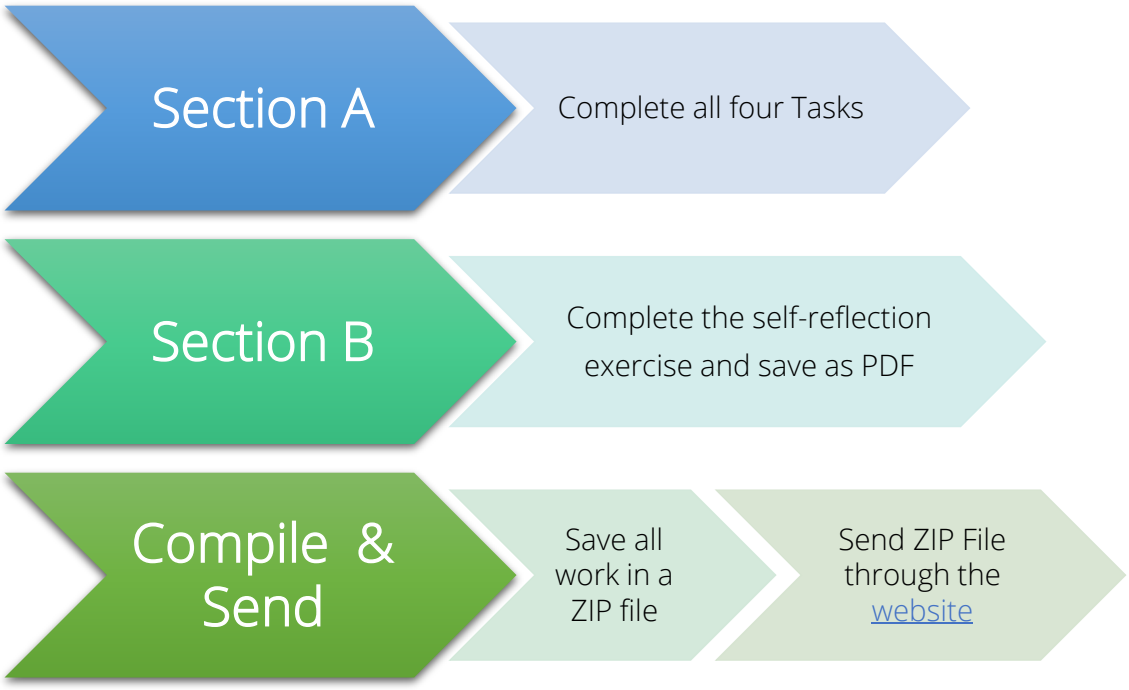


Workbook Guidelines.

This workbook includes:

- Four tasks in Section A
- Assessment Criteria for every task
- Self-Reflection Exercise in Section B
- Guidelines for the Final Presentation *(requirement for the 2 ECTS Applied MQF3 Award)*

Participants are expected to complete all the tasks presented in section A. Besides the development of the source code, participants are also expected to complete the self-reflection exercise in Section B.



ALL WORK MUST BE SAVED IN **ONE ZIP FILE** AND SENT BY THE CLOSING DATE AS INDICATED IN ACCREDITATION SECTION IN THE WEBSITE.

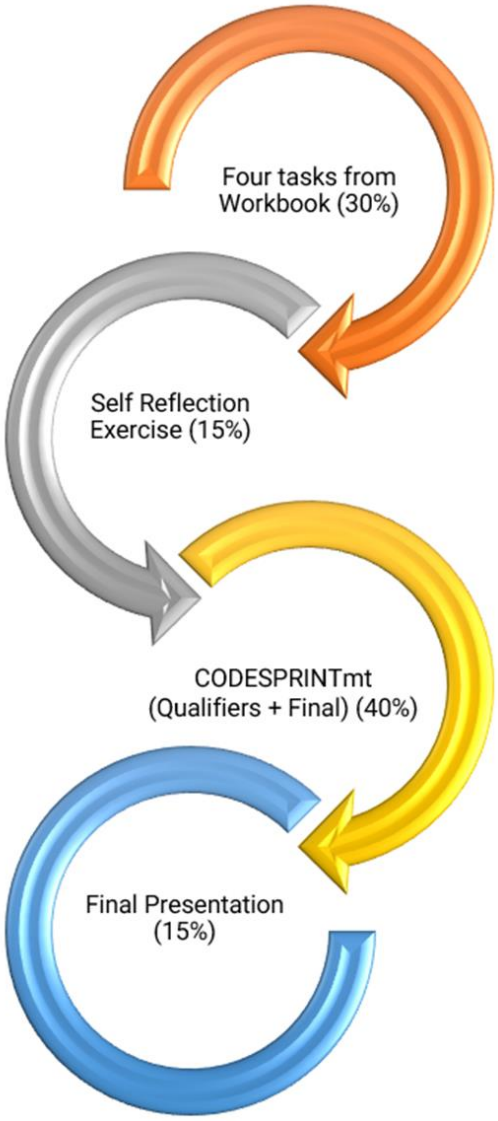
The Zip filename should be your name_surname_school.pdf
For example: alexia_brincat_stignatiuscollege.zip

Grading Distribution

The 2 ECTS Applied MQF 3 Certification is graded as follows:

0 - 39	40 - 59	60 - 89	90 - 100
FAIL	PASS	MERIT	DISTINCTION

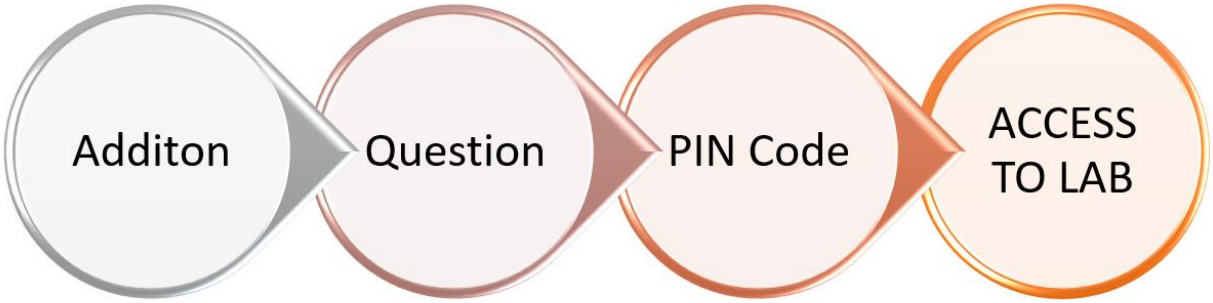
The marks from each phase of the event contribute to a global mark. The distribution of marks is as follows:



Section A

Task 1: Nuclear Power Plant Access Area

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 for the user's answer (char), the user's 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);
```

1. Name the class containing the main method RunApp2
2. Submit your program in a folder called Task2_Name; *e.g.* Task2_ John

Task 2: Lotto Ticket

The lottery is a game in which players pay for a ticket, select a group of numbers and win prizes based on how they match the drawn results.

Write a program that simulates a lottery system according to rules below:

Program Rules:

- The lottery system has a set of numbers available which is from 1 to 45.
- The lottery prize is that of €500,000.
- Five lottery numbers are automatically drawn and are not visible to the user. *Proper validation is required to avoid duplicate numbers.*
- The user is asked to purchase a lottery ticket.
- A lottery ticket is made up of five non-duplicate numbers. *Proper validation is required to avoid non-valid and duplicate numbers.*
- According to the numbers guessed, a prize is won:
  - With three numbers guessed, the user wins 10% of the lottery prize.
  - With four numbers guessed, the user wins 25% of the lottery prize.
  - With five numbers guessed, the user wins lottery prize in full.
- The result-screen should display the numbers drawn, the amount of numbers guessed, and the prize won (if applicable).

1. Name the class containing the main method **RunApp4**
2. Submit your program in a folder called **Task4_Name**; *e.g.* Task4_ John

Task 3: Point-of-Sale System

POS systems have replaced most traditional cash registers due to their ability to connect to the retailer's main database system. Having all data stored and accessible within one system makes daily operations more efficient and more profitable.

Write a program that simulates a POS system. This system has a Log-In Screen and a Main Menu as shown below:

Log In Screen	
Option	**Description**
Cashier Log In	A cashier must be logged-in to proceed to the Main Menu. *(Details of the registered cashiers is shown in **Table 2**)*
Exit	Terminates the program.

Main Menu	
Option	**Description**
Enter new transaction	Allows the cashier to enter the items that the client wants to buy. *(A list of the stock items is shown in **Table 1**)*
Issue Receipt	This option displays the receipt of the last transaction carried out. *(A sample is shown in **Figure 1**)*
Display Stock List	This option displays the list of items that a client can buy. *(A list of the stock items is shown in **Table 1**)*
Cashier Sign Out	Allows a logged-in cashier to sign out and return to the Log-In Screen. *(Details of the current registered cashiers is shown in **Table 2**)*

Stock List	
Items	Price
Printer	€67.99
Monitor	€138.00
Keyboard	€12.50
Graphics Card	€114.99
Soundbar	€249.00
Hard Disk	€66.95
Headset	€17.55
Smartwatch	€135.00
Camcorder	€329.00
Drone	€449.99

Table 1

```
-----------------------------------
 << EASY-SAVE SUPERMARKET >>
 102, Flower Street, Filfla
***********************************
 Sun Jan 20 11:48:45 CET 2019
 Receipt: 66432020
 Cashier: mallia.amy
***********************************
 MONITOR                 Eur 138.00
 HEADSET                 Eur 17.55
 SOUNBAR                 Eur 249.00


 SUBTOTAL:               EUR 404.55
 VAT:                    Eur 72.82
 TOTAL:                  Eur 477.37

***********************************
          FISCAL RECEIPT
            THANK  YOU
------------ CUT-HERE --------------
```

Figure 1: Sample Generated Receipt

Registered Cashiers	
Username	Password
borg.steve	Borg87max
zammit.rita	RitPopSing!
agius.john	ToyotaBeSt
vella.carlos.02	Rock!n!Roll
mallia.amy	$Gaga$Lady$

Table 2

1. Name the class containing the main method **RunApp5**
2. Submit your program in a folder called **Task5_Name**; *e.g.* Task5_ John

Hints:

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 'borg.joe', it can be changed to 'BORG.JOE'. This can be done using the code: `userInput = userInput.toUppercase();`

2. When comparing a String variable in a conditional statement the **`.equals()`** method should be used.
   - ⇨ **`if(userInput == "BORG.JOE"){}`** will not work because a String variable cannot be compared using **`==`**.
   - ⇨ The correct version is: **`if(userInput.equals("BORG.JOE")){}`**

3. To display the date:
   a. the library **`java.util.Date`** needs to be imported
   b. an instance of class Date should be created: **`Date myDate = new Date();`**
   c. display the date: **`System.out.println(myDate.toString());`**

Task 4: Vanishing Creatures (180 minutes)

The crew of the Starship Enterprise was destined to go on a time-travel adventure. But unlike any other time-travel movie dating to the 1980s, this journey is a creative piece of commentary on an environmentalism movement that put endangered species at its heart.

In the fall of 1986, Star Trek IV: The Voyage Home not only became a box-office hit, but it also propelled climate change and concern for endangered species into the mainstream. In short, Star Trek literally tried to save the whales in 1986, and it basically worked.

https://www.inverse.com/entertainment/star-trek-4-the-voyage-saved-the-whales

Develop a two-player game in which the players strive to become the hero of the game by saving as many vanishing animals as possible.

Functionality #1: Player Registration

Present a welcome screen with a menu prompting the two players to register their names to start playing or exit the program.

Functionality #2: Assign deck of cards to players

1. The game is populated with the records shown in Table 1 below.
2. The game randomly splits the 10 records into two decks of 5 cards, one for every player.
3. The decks should remain hidden from the players until revealed during gameplay.

Animal	Weight (kg)	Height (cm)	Life Expectancy	Population	Conservation Status*
African Elephant	4535	304	70	415000	1
Black Rhino	1397	158	40	5500	1
Orangutan	90	180	35	112200	1
African Wild Dog	31	76	10	1409	2
Chimpanzee	60	170	33	299700	2
Tiger	299	110	11	3900	2
Bonobo	55	88	25	50000	2
Hippopotamus	3628	160	36	130000	3
Giant Panda	150	121	20	1864	3
Polar Bear	589	240	30	3100	3
*Conservation Status: 1 = Critically Endangered, 2 = Endangered, 3 = Vulnerable					

Table 1: List of Endangered Animals

Functionality #3: Gameplay

1. The game picks a record from the list of endangered animals (as per Table 1) and presents it in the form of a card without showing the Conservation Status field, as in the sample screenshot below.

```
++++++++++++++++++++++++++
Animal: African Elephant
Weight: 4535 Kg
Height: 304 cm
Life Expectancy: 70 years
Population: 415000
++++++++++++++++++++++++++
```

2. The game randomly prompts either Player 1 or Player 2 to guess the Conservation Status of the chosen record by entering: 1 for Critically Endangered, 2 for Endangered, or 3 for Vulnerable.
3. If the chosen player guesses the Conservation Status, he/she wins the 'Conservation Status Challenge'. If not, the other player automatically wins this challenge.
4. The player who wins the Conservation Status Challenge is prompted to reveal the first card from his/her deck. The revealed card should not contain the Conservation Status field.

Assuming Kevin is the player who won the Conservation Status challenge, the prompting screen may look like the following sample screenshot.

```
----- Round 1 -----
Kevin, press Enter key to reveal your card

++++++++++++++++++++++++++
Animal: Bonobo
Weight: 55 Kg
Height: 88 cm
Life Expectancy: 25 years
Population: 50000
++++++++++++++++++++++++++
```

5. When the player's card is shown, the player then chooses which attribute to challenge the opponent with. For example, if in round 1 the player chooses "Population", the game will compare the population value of the first card of the player (50000) with that of the first card of the other player. The player whose card has the bigger attribute value wins the round and gains a Hero Point.
6. The game pauses until the player presses Enter to continue. Check sample screenshot below.

```
To challenge your opponent, choose an attribute:
Weight (w), Height (h), Life Expectancy (e), Population (p)
p

You have chosen Population.
You win this round.

Press the Enter key to continue
```

7. The winner of the round will be prompted to reveal the next card from his/her deck. The gameplay continues for five rounds.
8. After the 5th round, the Hero Points of each player are shown and the player with the higher Hero Points is declared as the winner and the hero of the game.
9. The game prompts the players to play again or exit the program.

Functionality #4: Validation

To enhance the user experience, warning messages should be used, and runtime errors avoided when invalid inputs or non-existing options are entered. These include:

- players' name cannot contain less than 3 characters.
- conversation status can only be 1, 2 or 3.
- attributes chosen can only be 'w', 'h', 'e', 'p' *(not case sensitive).*

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

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

Hint:

- To ignore case sensitivity, the player's input can be changed into its lowercase 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.toLowerCase(userInput);
  ```

- To check number of characters in a string, the **`.length()`** is used. For example, the code below displays the number of characters in variable name:
  ```
  String name = "Ramona";
  System.out.println(name.length());
  ```

Assessment Criteria

Nuclear Power Plant 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)*	**Maximum Score:** 26 + 2 for every extra feature			
0 – Not Satisfactorily | 1- Partly Satisfactorily | 2- Entirely Satisfactorily					

Lotto System				
Program Functionality	User Friendly Interface	Code Efficiency	Proper use of In-line Text *(Comments)*	Use of Proper Conventions *(Camel Case, meaningful variable names etc.)*
Name of Folder & Class/es	User Input	Suitable Prompts / Messages displayed	Numbers Drawn Automatically *(5 numbers)*	Validation *(non-duplicates & valid numbers)*
Arithmetic Calculations *(total numbers guessed & prize won)*	Proper Use of Data Structure *(such as Arrays)*	Other Features *(not listed in the task)*	**Maximum Score:** 24 + 2 for every extra feature	
0 – Not Satisfactorily | 1- Partly Satisfactorily | 2- Entirely Satisfactorily				

Point of Sale System

Program Functionality	User Friendly Interface	Code Efficiency	Proper use of In-line Text (Comments)	Use of Proper Conventions (Camel Case, meaningful variable names etc.)
Name of Folder & Class/es	User Input	Suitable Prompts / Messages displayed	Options Validation (Login Screen, Main Menu & New Transaction)	Functionality Validation (Issuing of Receipt)
Ignoring Case Sensitivity (when searching for username & item)	Proper Use of Data Structure (such as Arrays)	Searching of Records (Stock & Cashiers)	Arithmetic Calculations (Subtotal, VAT & Total)	Generating Receipt Number (8-digit Number)
Display Receipt (simulating a real receipt as much as possible)	Display list of items in stock (including formatting)	Other Features (not listed in the task)	Maximum Score: 34 + 2 for every extra feature	

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

Vanishing Creatures

Program Functionality	User-Friendly Interface	Proper use of Comments	Proper Conventions (Camel case, meaningful var names etc.)	Name of Folder & Class/es	User`s Input
Suitable Prompts / Messages displayed	Randomly assigns two decks of 5 cards	Randomly choose Player to start	Gameplay starts with the winner of the Conservation Status Challenge	Player winning the previous round, leads the next round.	Proper use of data structures.
Validation (Players' name)	Validation (Conservation Status)	Validation (Choosing attribute)	Validation (Avoid Runtime Error)	Other Features (Not listed in the task)	

Maximum Score: 32 + 2 for every extra feature.

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

Section B

Self-Reflection Exercise

The following ten questions will guide your reflection on the work that you did and the effort that you invested in preparation for CODESPRINTmt.

Answer all questions:

1. To what extent did you manage to complete the tasks? Rate 1-5.
   [1 being all tasks were not successfully finished and 5 being all tasks were successfully finished]
2. What did you learn from completing these tasks?
3. What did you like most about these tasks? Why?
4. What difficulties did you encounter in tackling these tasks? How did you solve them?
5. Which tasks / part of tasks did you find most challenging?
6. What level of support from your mentor did you seek in completing the tasks to the level you did?
7. What other resources have you used in completing this workbook?
8. Mention some testing procedures that you carried out for each task.
9. Suggest further improvements to your work.
10. What did you learn from completing this workbook?

Each question is awarded 10 marks, of which:

- 7 marks are given for the content, and
- 3 marks for using a reflective rather than a descriptive approach.

Final Presentation Guidelines

Prepare a presentation to briefly outline your CODESPRINT journey including:

- An introduction of:
  - yourself,
  - your coding backgrounds,
  - your past CODESPRINT experiences (if any).

- The problem-solving strategies that you employed in the different tasks.
  - If any, identify situations where you have found more than one solution to tackle part/full task and on what grounds did you choose one strategy over another.

- The difficulties you encountered, and which were:
  - solved and how did you overcome your difficulties,
  - not solved.

- The lessons you have learned from this experience.

- Your suggestions to future candidates.

- Your suggestions to CODESPRINT organisers.

Your presentation will be assessed as follows:

40%	30%	15%	10%	5%
Showing a reflective approach, not just demonstrating a descriptive journal of your experience.	Using at least three different sources of support, such as mentor's support and online sources.	Answering all above questions.	Completing your presentation in maximum of five minutes.	Showing a sense of originality in your presentation.

Your presentation:

- can take any format that you prefer such as a slide-show, word-processed document, a video, a website or any other preferred presentation method,
- should not exceed five minutes,
- should be presented during the CODESPRINT Award Ceremony.

NOTES

Contact Us:

www.codesprintmalta.edu.mt

Code Sprint Malta

A project created by the Computing Department, within the Directorate for Learning and Assessment Programmes (DLAP)

Directorate for Learning and Assessment Programmes, MEDE