RIPHAH INTERNATIONAL UNIVERSITY, LAHORE CAMPUS.

SCHOOL OF COMPUTING & INNOVATION

PROGRAMMING FUNDAMENTALS

ASSIGNMENT 03

Objectives:

- Implementation of arrays
- Working on problems with arrays and loops

Instructions:

- Assignment type is individual, so no sharing is allowed.
- You can use internet and books as helping resources but sharing content with peers is strictly prohibited.
- Make a word document add your solution code and output screens

---

Submission Method:

- Submit this file at **Moellim**.

Hearts Mining Game

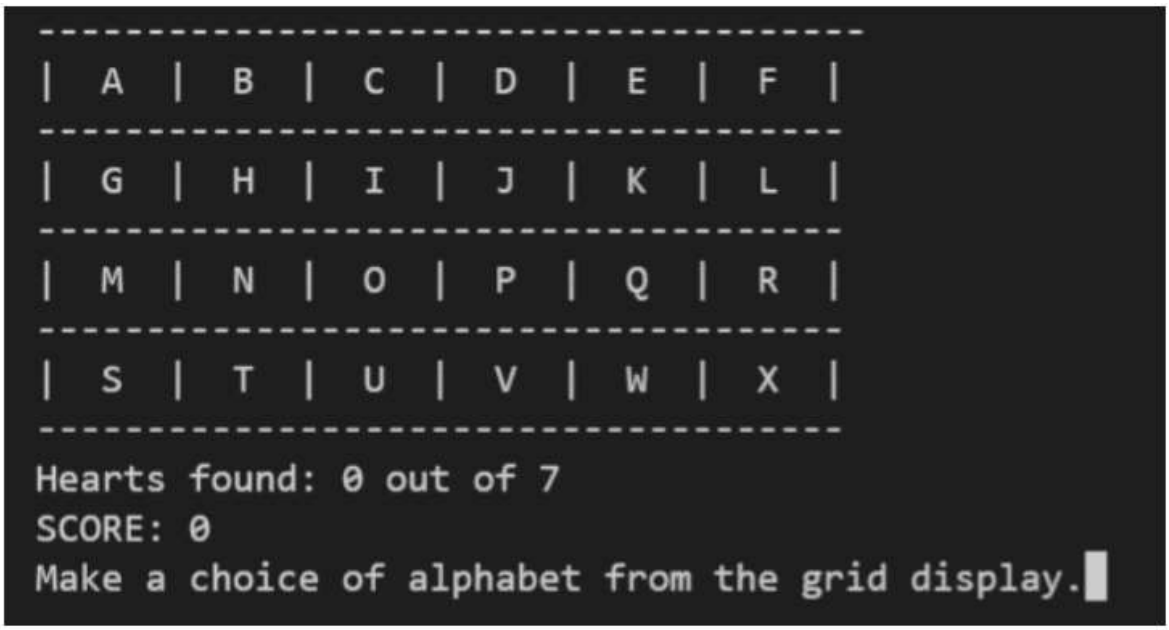
We have put together a hearts mining game for the programming portfolio module, which

has a total of 24 spades ♠ and hearts ♥ buried in grid cells. There are four rows and six columns in the game grid. When a user selects an alphabet assigned to a grid cell, the item buried (spade or hearts) in that grid is revealed to them.

The game aims to discover all the buried hearts with fewer grid cell selections. Your score increases when you get "hearts", while it decreases upon discovering "spades".

Now, write a program that will allow a user to play the game. You could make use of Arrays. The program will have the following steps.

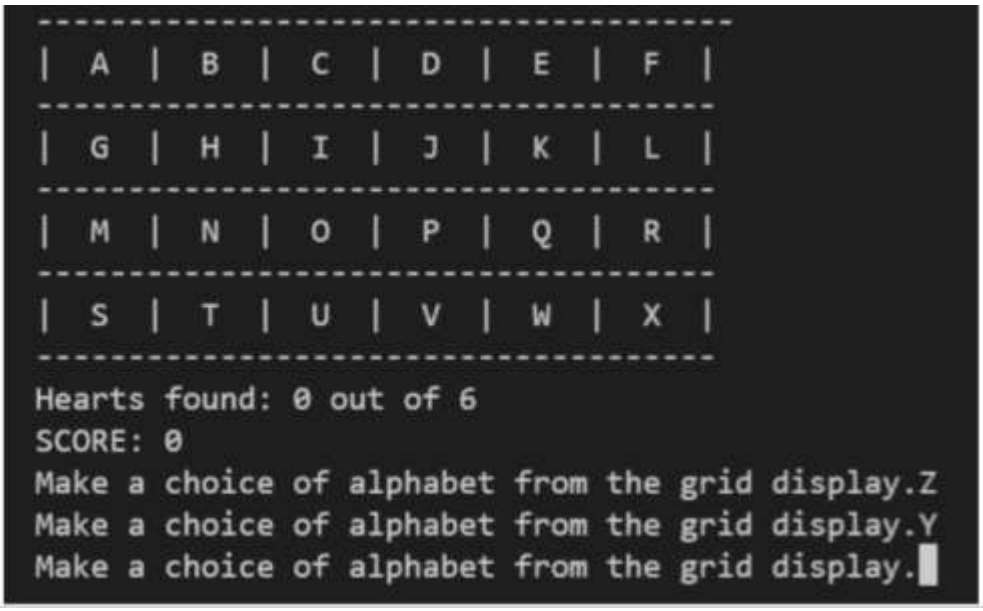
1. Your program should display the initial game grid, consisting of letters (alphabets) when the correct level is selected. The "total hearts" buried and the initial score of zero should also be displayed (see below). The program should then prompts the player to choose from the letters to reveal the treasure buried behind that letter (alphabet).



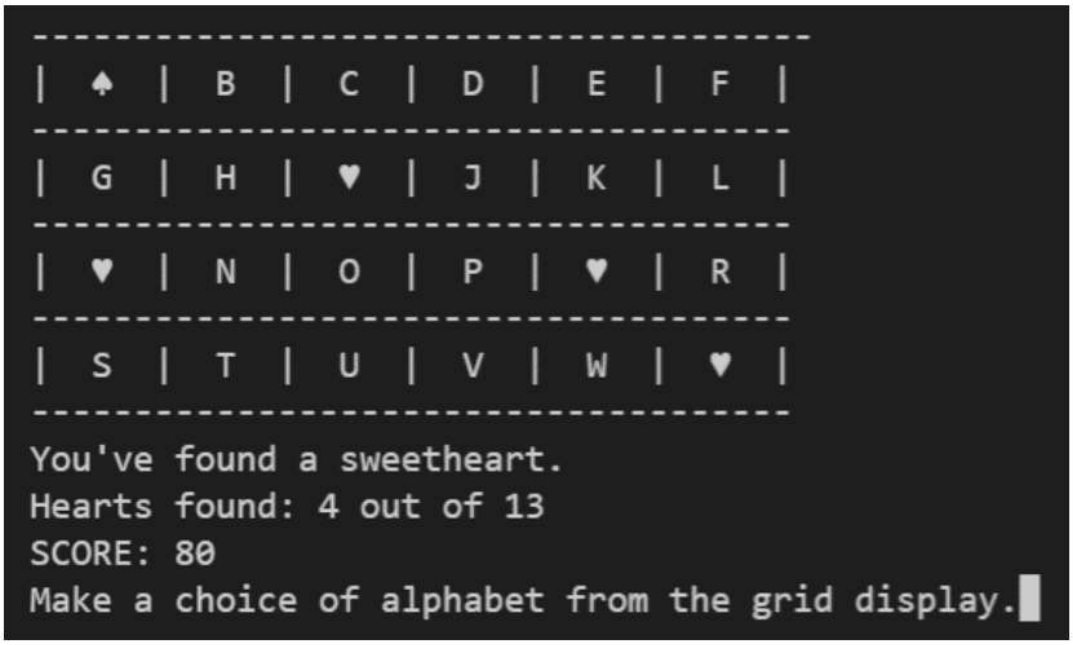
2. At the start, you should randomly bury the "hearts" and "spades" behind the alphabets. To implement this, you could use two arrays. The first array stores the alphabets displayed to the user, and the second stores Unicode characters of spadesand hearts, which is not visible to the player. The Unicode character for spades and hearts are '\u2660' and '\u2665', respectively.

3. Start the game loop by doing the following:

   - Prompt the player to select a letter from the displayed alphabets.
   - If a player selects an alphabet not currently displayed, the program should prompt them to enter the alphabet again until they have entered a correct letter. To do this, search the alphabet grid to find the indices of the letter entered. If found, terminate the prompt, otherwise, ask the player to enter the letter again.



- When the player's choice of the alphabet is found, replace the letter store at the found position of the alphabet array with the Unicode character of spade or hearts stored at the same location in the "isHeart" array.
- Then display the updated alphabet array grid together with the player's score.



6. Once the player has found all the buried hearts, the score is calculated as the number of hearts found divided by the number of alphabets selected by the user, then multiplied by 100. To do this, you must keep track of the number of hearts found, the total number of alphabets selected. In the above, there are four hearts and a spade:

   - Number of hearts = 4
   - Total number of alphabets = 4 + 1 = 5
   - SCORE = (4/5) * 100 = 80

7. The program writes the player game statistics to a file. It includes the score, difficulty level, total number of hearts generated for the game, and total number of selections made before finding all "hearts".

8. The program prompts the player if they want to play again. The game should continue if the player answers yes (Y) at this prompt and terminate otherwise.