Introduction:

You will code and execute a C language program **applying user-defined data types (structures).** You will be programming a small game that has hidden **'bombs'** and **'treasure'** along a path of variable '**distance**'. The game requires the player to enter move location commands to reveal what is hidden at a given position along the path. The object of the game is for the player to find as many treasures as possible before running out of moves or lives. Discovering a bomb will reduce the player's life count. Discovering a treasure will earn the player treasure points. Discovering both, a treasure with a bomb in the same location will reduce the player's life count and earn the player treasure points (consider it a life insurance payout). Prior to playing the game, the program will prompt the user to perform some upfront configurations to the player and the game components – these settings will define how the game is played.

Part-1

Instructions

Part-1 will focus on the **player** and **game** configuration settings in preparation for gameplay which will be done in Part-2.

1. Carefully review the "Part-1 Output Example" (next section) to see how this program is expected to work (Note: This game is highly user-configurable and should be coded to implement the settings as defined by the user and not be limited to just the example provided – you will have to test your work thoroughly in both part's 1 and 2!)
2. Code your program in a file named "**w5p1.c**"
3. You will need to create a user-defined data type called **PlayerInfo** which is used for configuring a player in the game with members that can store the following related information:
   - The **number of "lives"** a player can have for the game
   - A **character symbol** that will be used to represent the player
   - A counter to store the **number of "treasure's"** found during the game
   - A **history of all past entered positions** entered by the player during the game (hint: you should size this array based on a macro that represents the **maximum path length** that a game can be configured for – see example output to see what the maximum is)
4. You will need to create another user-defined data type called **GameInfo** which is used for configuring the game settings with members that can store the following related information:
   - The **maximum number of "moves"** a player can make for a game
   - The **path length** (number of positions) the game path will have for a game
   - A series of **0**'s and **1**'s in an array that represents where **bombs** are buried along the path (hint: you should size this array based on a **macro that represents the maximum path length** that a game can be configured for – see example output to see what the maximum is)
   - A series of 0's and 1's in an array that represents where **treasure** is buried along the path (hint: you should size this array based on a **macro that represents the maximum path length** that a game can be configured for – see example output to see what the maximum is)

5. Configure the **player** (store these values to a variable of type **PlayerInfo**):
   - Prompt to set the player's **character symbol** (any printable character that will represent the player)
     - Note: Place a **single space** before the % specifier in the scanf to properly read this value

       ```
       scanf(" %c"...
       ```
   - Prompt to set the **number of lives** a player is limited to for the game
     - The value must be between **1** and **10** inclusive
     - Note: you should design your code so that the maximum value rule can be easily modified in one place, so you **do not need to make changes to the logic** of the program
     - Validation should repeat as many times as necessary until a valid value is entered
   - Make sure the history of moves (all user entered positions during gameplay) is set to a safe empty state – **you should assume there is potentially previous game data still stored that needs each element to be reset)**
6. Configure the **game** (store these values to a variable of type **GameInfo**):
   - Prompt to set the **length of the game path** (this is the number of positions in the path)
     - The value must be between **10** and **70**
     - The value must be a **multiple of 5**
     - Note: you should design your code so that these rules (values: 5, 10, 70) can be easily modified in one place, so you **do not need to make changes to the logic** of the program
     - Validation should repeat as many times as necessary until a valid value is entered
   - Prompt to set the **maximum number of moves** a player can make during gameplay
     - The value must be at least the value of the **player's "lives"** setting
     - The value cannot be greater than 75% of the game's **path length** setting (round down to nearest whole number)
     - Validation should repeat as many times as necessary until a valid value is entered
   - Prompt to set the **BOMB's** placements along the path (within the game's path length limits)
     - Values **must be entered 5 at a time** (sets of 5) until all positions along the set path length are set (space delimited)
       - Reminder: The multiple of 5 rule can be modified with another version of this application and should be coded with this mind (see note at the beginning of #6)
     - A '**1**' value represents a **hidden bomb**, while a '**0**' value represents **no bomb**
     - Note: You do not need to validate for **1**'s and **0**'s; you may assume this is entered properly
   - Prompt to set the **TREASURE** placements along the path (within the game's path length limits)
     - The same rules apply as described for the bomb settings
7. As the last major step, **display a summary** of the values entered that will define the gameplay

Part-1: Output Example:

```
================================
         Treasure Hunt!
================================

PLAYER Configuration
--------------------
Enter a single character to represent the player: @
Set the number of lives: 0
     Must be between 1 and 10!
Set the number of lives: 11
     Must be between 1 and 10!
Set the number of lives: 3
Player configuration set-up is complete

GAME Configuration
------------------
Set the path length (a multiple of 5 between 10-70): 9
     Must be a multiple of 5 and between 10-70!!!
Set the path length (a multiple of 5 between 10-70): 71
     Must be a multiple of 5 and between 10-70!!!
Set the path length (a multiple of 5 between 10-70): 19
     Must be a multiple of 5 and between 10-70!!!
Set the path length (a multiple of 5 between 10-70): 35
Set the limit for number of moves allowed: 2
    Value must be between 3 and 26
Set the limit for number of moves allowed: 27
    Value must be between 3 and 26
Set the limit for number of moves allowed: 10

BOMB Placement
--------------
Enter the bomb positions in sets of 5 where a value
of 1=BOMB,and 0=NO BOMB. Space-delimit your input.
(Example: 1 0 0 1 1) NOTE: there are 35 to set!
   Positions [ 1- 5]: 0 0 0 0 1
   Positions [ 6-10]: 1 0 0 1 1
   Positions [11-15]: 1 0 1 1 1
   Positions [16-20]: 0 1 0 0 0
   Positions [21-25]: 1 0 1 0 0
   Positions [26-30]: 0 0 0 1 0
   Positions [31-35]: 1 0 1 0 1
BOMB placement set
```

```
TREASURE Placement
------------------
Enter the treasure placements in sets of 5 where a value
of 1=TREASURE, and 0=NO TREASURE. Space-delimit your input.
```

```
(Example: 1 0 0 1 1) NOTE: there are 35 to set!
   Positions [ 1- 5]: 0 0 1 0 0
   Positions [ 6-10]: 1 1 1 0 1
   Positions [11-15]: 1 1 0 1 0
   Positions [16-20]: 0 1 0 0 0
   Positions [21-25]: 1 1 0 1 0
   Positions [26-30]: 1 0 1 0 0
   Positions [31-35]: 0 1 1 1 1
TREASURE placement set

GAME configuration set-up is complete...

------------------------------------
TREASURE HUNT Configuration Settings
------------------------------------
Player:
   Symbol     : @
   Lives      : 3
   Treasure   : [ready for gameplay]
   History    : [ready for gameplay]

Game:
   Path Length: 35
   Bombs      : 00001100111011101000101000001010101
   Treasure   : 00100111011101001000110101010001111

====================================
~ Get ready to play TREASURE HUNT! ~
====================================
```