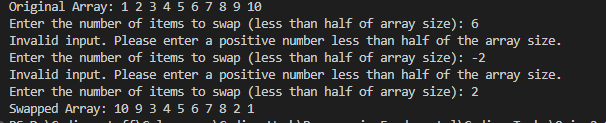
**Q1. Write a program that initializes an array with the values 1 to 10 . The user is prompted to enter the number of items to swap. This number must be less than half of the array size. The program should also ensure that the user inputs a positive number. If the user enters a negative number, prompt them to re-enter a valid number.**

**The program will then perform the swapping operation as follows:**

* **Swap the first element with the last element.**
* **Swap the second element with the second-to-last element.**
* **Continue this pattern until the specified number of swaps is completed.**

****

**Hint:**

* **The last element of the array is less than 1 of its total size.**

**#include <stdio.h>**

**int main() {**

**int arr[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};**

**int size = sizeof(arr) / sizeof(arr[0]); // size of the array is 5**

**int swaps;**

**// Display the original array**

**printf("Original Array: ");**

**for (int i = 0; i < size; i++) {**

**printf("%d ", arr[i]);**

**}**

**printf("\n");**

**// Input validation for number of swaps**

**do {**

**printf("Enter the number of items to swap (less than half of array size): ");**

**scanf("%d", &swaps);**

**if (swaps <= 0 || swaps >= size / 2) {**

**printf("Invalid input. Please enter a positive number less than half of the array size.\n");**

**}**

**} while (swaps <= 0 || swaps >= size / 2);**

**// Perform swapping**

**for (int i = 0; i < swaps; i++) {**

**int temp = arr[i];**

**arr[i] = arr[size - 1 - i];**

**arr[size - 1 - i] = temp;**

**}**

**// Display the swapped array**

**printf("Swapped Array: ");**

**for (int i = 0; i < size; i++) {**

**printf("%d ", arr[i]);**

**}**

**printf("\n");**

**return 0;**

**}**

**Q2. Below is Code Snippet of Treasure hunter game (project 2) your task is:**

**When the player's score becomes a multiple of 5, generate a new obstacle (X) on the grid.**

**The total number of obstacles should not exceed 10.**

**When a player collides with any of the obstacles (X) the game should end.**

**Hints:**

* **Use two arrays to store the X and Y positions of the obstacles.**
* **Use a flag for knowing if it's an obstacle or not when printing it on the Grid.**
* **Dots will be printed after obstacle logic.**

**int main() {**

**//Rest of code**

**int coinX = rand() % WIDTH;**

**int coinY = rand() % HEIGHT;**

**int flag = 1;**

**int maxObstacles = 10; // Maximum number of obstacles**

**int obstacleCount = 0; // Current number of obstacles**

**int obstacleX[maxObstacles];**

**int obstacleY[maxObstacles];**

**while (flag == 1) {**

**system("cls");**

**//Coin collection Code**

**// Add an obstacle if the score is a multiple of 5**

**if (score % 5 == 0 && obstacleCount < maxObstacles) {**

**obstacleX[obstacleCount] = rand() % WIDTH;**

**obstacleY[obstacleCount] = rand() % HEIGHT;**

**obstacleCount++;**

**}**

**// Check if player hits an obstacle**

**for (int i = 0; i < obstacleCount; i++) {**

**if (playerX == obstacleX[i] && playerY == obstacleY[i]) {**

**printf("Game Over! You hit an obstacle.\n");**

**flag = 0;**

**}**

**}**

**for (int i = 0; i < HEIGHT; i++) {**

**for (int j = 0; j < WIDTH; j++) {**

**if (i == playerY && j == playerX) {**

**printf("@");**

**} else if (i == coinY && j == coinX) {**

**printf("O");**

**} else {**

**int isObstacle = 0;**

**for (int k = 0; k < obstacleCount; k++) {**

**if (i == obstacleY[k] && j == obstacleX[k]) {**

**printf("X");**

**isObstacle = 1;**

**}**

**}**

**if (!isObstacle) {**

**printf(".");**

**}**

**}**

**}**

**}**

**printf("\n");**

**}**

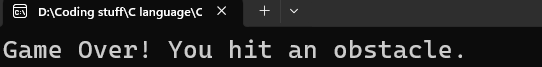
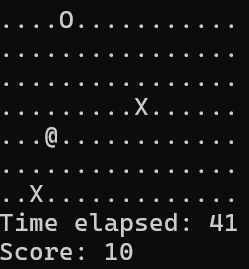
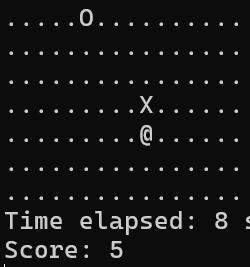
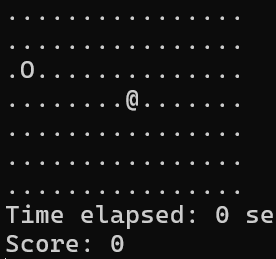
**// Display elapsed time**

**printf("Time elapsed: %ld seconds\n", time(0) - startTime);**

**printf("Score: %d\n", score);**

**//Movement Code**

**} } return 0; }**

****