```
// TreasureHuntGame.cpp : This file contains the 'main' function. Program execution begins and ends there.
// Author: Gabriela Gutierrez
// Purpose: Player has to guess X and Y coordinates to find the hidden treasure (T).
//

#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

// GLOBAL CONSTANTS
const int ROWS = 10;
const int COLUMNS = 10;

// STRUCTURES
struct Treasure {
   string name;
   int xCoordinate;
   int yCoordinate;
   double value;
   bool found;
};

// FUNCTION PROTOTYPES
void fillArray(Treasure[]);
void fillGrid(char[][COLUMNS], const char);
void printGrid(char[][COLUMNS]);
void introHuntGame();
int validateInput(int);
```

```
double checkForTreasure(char[][COLUMNS], Treasure[], int, int, int);
void endGameMsg(char[][COLUMNS], Treasure[], double, int);

int main() {
   // CONSTANTS, VARIABLES AND ARRAYS
   const int TREASURE = 5;  // Size of the structure array
   const int ATTEMPTS = 5;
   const char GRID_SYM = '-';
   int xUser = 0;
   int yUser = 0;
   int counter = 0;  // Counter for # of guesses
   double accumu = 0;  // Accumulator for money looted
   double total = 0.0;
   char reply = ' ';
   Treasure hunt[TREASURE];
   char grid[ROWS][COLUMNS] = { ' ' };

   cout << fixed << setprecision(2);

   fillArray(hunt);
   introHuntGame();
   fillGrid(grid, GRID_SYM);


   do {
      printGrid(grid);
      cout << "Enter the x coordinate:";
      cin >> xUser;
      xUser = validateInput(xUser);
      cout << endl;
      cout << "Enter the y coordinate:";
```

```
        cin >> yUser;
        yUser = validateInput(yUser);
        cout << endl;

        accumu = checkForTreasure(grid, hunt, xUser, yUser, TREASURE);
        total += accumu;
        counter++;
        cout << "Would you like to play again (Y or N)? \n";
        cin >> reply;
        cout << endl;

        if (counter >= ATTEMPTS) {
           reply = 'N';
        }

    } while (toupper(reply) == 'Y');

    endGameMsg(grid, hunt, total, TREASURE);

    return 0;
}

// FUNCTION DEFINITIONS(7)
void fillArray(Treasure ht[]) {
    ht[0] = { "Pirate's Chest", 2, 4, 145.90, true };
    ht[1] = { "Dragon's Chest", 5, 8, 258.24, true };
    ht[2] = { "Troll's Chest", 3, 1, 203.71, true };
    ht[3] = { "Queen's Chest", 8, 0, 190.15, true };
    ht[4] = { "Thief's Chest", 1, 9, 98.95, true };
}
```

```
void fillGrid(char grid[][COLUMNS], const char SYM) {
    for (int r = 0; r < ROWS; r++) {
        for (int c = 0; c < COLUMNS; c++) {
            grid[r][c] = SYM;
        }
    }
}

void printGrid(char arr[][COLUMNS]) {
    cout << "  0 1 2 3 4 5 6 7 8 9\n";
    for (int r = 0; r < ROWS; r++) {
        cout << r << ' ';
        for (int c = 0; c < COLUMNS; c++) {
            cout << arr[r][c] << ' ';
        }
        cout << endl;
    }
    cout << endl;
    cout << "Legend: - (Unknown); E (Empty); T (Treasure)\n\n";
}

void introHuntGame() {
    cout << "This is a treasure hunt game.\n"
        << "You have 3 attempts to find the 5 treasure chests hidden in the grid below.\n";
    cout << endl;
}

int validateInput(int input) {
    while ((input < 0 || input > 9) || cin.fail()) {
        if (cin.fail()) {
            cin.clear();
```

```
            cin.ignore(100, '\n');
        }
        cout << "That entry is invalid. Please enter a valid integer based on the grid size:\n";
        cin >> input;
        cout << endl;
    }
    return input;
}

double checkForTreasure(char grid[][COLUMNS], Treasure st[], int x, int y, int SZ) {
    const char TREASURE = 'T';
    const char EMPTY = 'E';
    double total = 0.0;

    for (int i = 0; i < SZ; i++) {
        if (st[i].xCoordinate == x && st[i].yCoordinate == y) {
            if (st[i].found) {
                cout << "You have found the " << st[i].name << "!\n";
                cout << "Inside you find $" << st[i].value << " worth of valuables.\n\n";
                total = st[i].value;
                grid[st[i].xCoordinate][st[i].yCoordinate] = TREASURE;
                st[i].found = false;
            }
            else {
                cout << "This treasure has already been claimed.\n";
            }
        }
    }
    if (grid[x][y] != TREASURE) {
        cout << "Sorry, there is no treasure at those coordinates.\n";
        grid[x][y] = EMPTY;
```

```
    }
    return total;
}

void endGameMsg(char arr[][COLUMNS], Treasure st[], double total, int TREASURES) {
    const char TREASURE = 'T';

    cout << "The game is now over. You looted $" << total << " worth of valuables from chests.\n";
    cout << "The grid below shows the position of all treasure chests:\n";
    cout << endl;
    for (int i = 0; i < TREASURES; i++) {
        arr[st[i].xCoordinate][st[i].yCoordinate] = TREASURE;
    }
    printGrid(arr);
}
```