

2331962 Olcay Sivas

by Olcay Sivas

Submission date: 09-Jan-2026 01:18PM (UTC+0000)

Submission ID: 272674764

File name: 54350_Olcay_Sivas_2331962_Olcay_Sivas_703116_1114450451.docx (2.68M)

Word count: 883

Character count: 4498

Name: Olcay Sivas

Student Number: 2331962

WORKSHEET 5 – TESTING (Battleships)

For this worksheet, the program can be implemented in a number of different ways and demonstrate different features.

Create your own test set to demonstrate strengths and/or limitations of your solution. Use as many of the template boxes below as you need.

Test i. Randomization test. Run the program twice and check if the battleship's placement randomized or not.

Test ii. Round progression test. Run the program and sink all ships in level 1 (6 ships) and see if the column gets bigger or not.

Test iii. Power up logic. Buy horizontal shot power up and shoot the very right of the column and see if program crashes or not.

Test iv. Shop and gold carry over test. Get gold on first level and beat the level then see if gold still there in level 2 or not.

Test: Test i. Randomization test. Run the program twice and check if the battleship's placement randomized or not.

Screenshot(s) of output:

```
C:\Users\slivas\OneDrive\Desktop >
 0 1 2 3 4
0 M M X X X
1 M M M M X
2 X X X X X
3 X X X X X
4 X X X X X

--- STATUS: Level 1 | Ships: 6 | Gold: 0 ---
1. Standard Shot (Free)
2. HORIZONTAL BLAST (100 Gold)
3. VERTICAL BLAST (100 Gold)
Choose action: 1
Enter Row (0-4): 1
Enter Col (0-4): 4

--- FIRING ---
(1,4) BOOM! Hit! (+50 Gold)

 0 1 2 3 4
0 M M X X X
1 M M M M H
2 X X X X X
3 X X X X X
4 X X X X X

--- STATUS: Level 1 | Ships: 5 | Gold: 50 ---
1. Standard Shot (Free)
2. HORIZONTAL BLAST (100 Gold)
3. VERTICAL BLAST (100 Gold)
Choose action: |
```

```
C:\Users\slivas\OneDrive\Desktop >
 0 1 2 3 4
0 X X X X X
1 X X X X X
2 X X X X X
3 X X X X X
4 X X X X X

--- STATUS: Level 1 | Ships: 6 | Gold: 0 ---
1. Standard Shot (Free)
2. HORIZONTAL BLAST (100 Gold)
3. VERTICAL BLAST (100 Gold)
Choose action: 1
Enter Row (0-4): 1
Enter Col (0-4): 4

--- FIRING ---
(1,4) Miss.

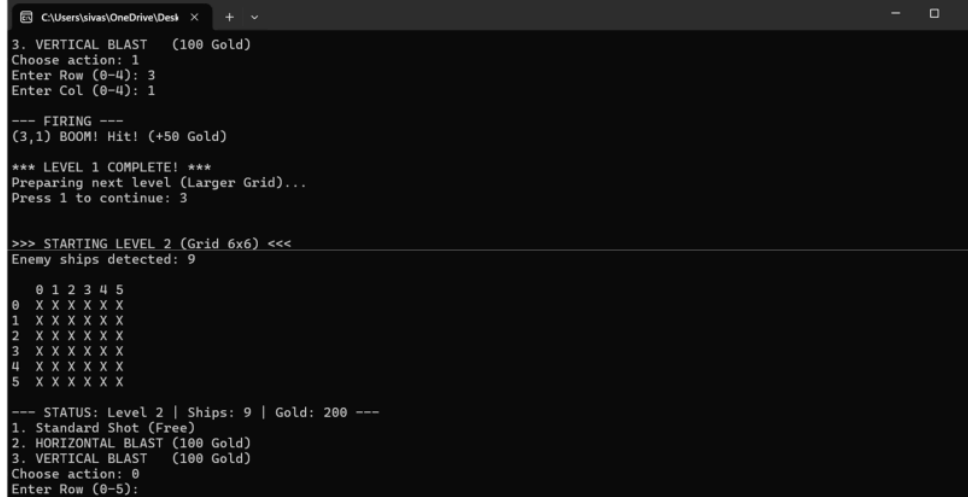
 0 1 2 3 4
0 X X X X X
1 X X X X M
2 X X X X X
3 X X X X X
4 X X X X X

--- STATUS: Level 1 | Ships: 6 | Gold: 0 ---
1. Standard Shot (Free)
2. HORIZONTAL BLAST (100 Gold)
3. VERTICAL BLAST (100 Gold)
Choose action: |
```

Result: Did the programme behave as expected ie syntactically and logically correct?
Yes, the ships were generated in different coordinates every time the program ran and randomization was confirmed.

Test: Test ii. Round progression test. Run the program and sink all ships in level 1 (6 ships) and see if the column gets bigger or not.

Screenshot(s) of output:



```
C:\Users\slvas\OneDrive\Desktop > .\game.exe
3. VERTICAL BLAST (100 Gold)
Choose action: 1
Enter Row (0-4): 3
Enter Col (0-4): 1

--- FIRING ---
(3,1) BOOM! Hit! (+50 Gold)

*** LEVEL 1 COMPLETE! ***
Preparing next level (Larger Grid)...
Press 1 to continue: 3

>>> STARTING LEVEL 2 (Grid 6x6) <<<
Enemy ships detected: 9

  0 1 2 3 4 5
0  X X X X X X
1  X X X X X X
2  X X X X X X
3  X X X X X X
4  X X X X X X
5  X X X X X X

--- STATUS: Level 2 | Ships: 9 | Gold: 200 ---
1. Standard Shot (Free)
2. HORIZONTAL BLAST (100 Gold)
3. VERTICAL BLAST (100 Gold)
Choose action: 0
Enter Row (0-5):
```

Result: Did the programme behave as expected ie syntactically and logically correct?
Yes, after level 1 had been cleared, the game correctly advanced to Level 2 and the grid was expanded to 6x6.

Test: Test iii. Power up logic. Buy horizontal shot power up and shoot the very right of the column and see if the program crashes or not.

Screenshot(s) of output:

```
C:\Users\shvas\OneDrive\Desktop x + v
0 M M M M X
1 H M H X X
2 X X X X X
3 X X X X X
4 X X X X X

--- STATUS: Level 1 | Ships: 4 | Gold: 100 ---
1. Standard Shot (Free)
2. HORIZONTAL BLAST (100 Gold)
3. VERTICAL BLAST (100 Gold)
Choose action: 2
>> EQUIPPED: Horizontal Blast!
Enter Row (0-4): 0
Enter Col (0-4): 4

--- FIRING ---
(0,4) Miss.

  0 1 2 3 4
0 M M M M M
1 H M H X X
2 X X X X X
3 X X X X X
4 X X X X X

--- STATUS: Level 1 | Ships: 4 | Gold: 0 ---
1. Standard Shot (Free)
2. HORIZONTAL BLAST (100 Gold)
3. VERTICAL BLAST (100 Gold)
Choose action: |
```

Result: Did the programme behave as expected ie syntactically and logically correct?
Yes, the grid edge was correctly handled by the horizontal shoot power-up and it didn't crash.

Test: Test iv. Shop and gold carry over test. Get gold on first level and beat the level then see if gold still there in level 2 or not.

Screenshot(s) of output:

```
C:\Users\shvas\OneDrive\Desktop x + v
3. VERTICAL BLAST (100 Gold)
Choose action: 1
Enter Row (0-4): 3
Enter Col (0-4): 1

--- FIRING ---
(3,1) BOOM! Hit! (+50 Gold)

*** LEVEL 1 COMPLETE! ***
Preparing next level (Larger Grid)...
Press 1 to continue: 3

>>> STARTING LEVEL 2 (Grid 6x6) <<<
Enemy ships detected: 9

  0 1 2 3 4 5
0 X X X X X X
1 X X X X X X
2 X X X X X X
3 X X X X X X
4 X X X X X X
5 X X X X X X

--- STATUS: Level 2 | Ships: 9 | Gold: 200 ---
1. Standard Shot (Free)
2. HORIZONTAL BLAST (100 Gold)
3. VERTICAL BLAST (100 Gold)
Choose action: 0
Enter Row (0-5):
```

Result: Did the programme behave as expected ie syntactically and logically correct?
Yes, gold earned in Level 1 was successfully saved and available for use in the Level 2 shop.

Final code:

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define MAX_SIZE 10

void main()
{
    char grid[MAX_SIZE][MAX_SIZE];
    int row, col;
    int gold = 0;
    int level, choice, shotType, turns, gridSize, totalShips, shipsLeft;

    srand(time(NULL));

    printf("=====\n");
    printf("  ULTIMATE BATTLESHIPS: CAMPAIGN   \n");
    printf("=====\n");

    for (level = 1; level <= 3; level++)
    {
        if (level == 1) { gridSize = 5; totalShips = 6; }
        if (level == 2) { gridSize = 6; totalShips = 9; }
        if (level == 3) { gridSize = 7; totalShips = 12; }

        shipsLeft = totalShips;
        turns = 0;

        for (int i = 0; i < gridSize; i++)
        {
            for (int j = 0; j < gridSize; j++)
            {
                grid[i][j] = 'X';
            }
        }

        int shipsPlaced = 0;
        while (shipsPlaced < totalShips)
        {
            int rShip = rand() % gridSize;
            int cShip = rand() % gridSize;

            if (grid[rShip][cShip] != 'S')
            {
                grid[rShip][cShip] = 'S';
                shipsPlaced++;
            }
        }
    }
}
```

```

    }

    printf("\n\n>>> STARTING LEVEL %d (Grid %dx%d) <<<\n", level, gridSize,
gridSize);

    printf("Enemy ships detected: %d\n", totalShips);

    while (shipsLeft > 0)
    {
        printf("\n ");
        for (int j = 0; j < gridSize; j++) printf("%d ", j);
        printf("\n");

        for (int i = 0; i < gridSize; i++)
        {
            printf("%d ", i);
            for (int j = 0; j < gridSize; j++)
            {
                if (grid[i][j] == 'S')
                {
                    printf("X ");
                }
                else
                {
                    printf("%c ", grid[i][j]);
                }
            }
            printf("\n");
        }

        printf("\n--- STATUS: Level %d | Ships: %d | Gold: %d ---\n", level,
shipsLeft, gold);

        printf("1. Standard Shot (Free)\n");
        printf("2. HORIZONTAL BLAST (100 Gold)\n");
        printf("3. VERTICAL BLAST (100 Gold)\n");
        printf("Choose action: ");
        scanf("%d", &choice);

        shotType = 1;

        if (choice == 2)
        {
            if (gold >= 100) {
                gold -= 100; shotType = 2;
                printf(">> EQUIPPED: Horizontal Blast!\n");
            }
            else printf(">> NOT ENOUGH GOLD!\n");
        }
        else if (choice == 3)
        {
            if (gold >= 100) {
                gold -= 100; shotType = 3;

```

```

        printf(">> EQUIPPED: Vertical Blast!\n");
    }
    else printf(">> NOT ENOUGH GOLD!\n");
}

printf("Enter Row (0-%d): ", gridSize - 1);
scanf("%d", &row);
printf("Enter Col (0-%d): ", gridSize - 1);
scanf("%d", &col);

if (row < 0 || row >= gridSize || col < 0 || col >= gridSize)
{
    printf("Invalid coordinates!\n");
    continue;
}

int shots = 1;
if (shotType > 1) shots = 2;

printf("\n--- FIRING ---\n");
for (int k = 0; k < shots; k++)
{
    int r = row;
    int c = col;

    if (k == 1)
    {
        if (shotType == 2) c++;
        if (shotType == 3) r++;
    }

    if (r >= 0 && r < gridSize && c >= 0 && c < gridSize)
    {
        if (grid[r][c] == 'S')
        {
            printf("(%d,%d) BOOM! Hit! (+50 Gold)\n", r, c);
            grid[r][c] = 'H';
            shipsLeft--;
            gold += 50;
        }
        else if (grid[r][c] == 'X')
        {
            printf("(%d,%d) Miss.\n", r, c);
            grid[r][c] = 'M';
        }
        else
        {
            printf("(%d,%d) Already hit.\n", r, c);
        }
    }
}

```

```
        turns++;
    }
    printf("\n*** LEVEL %d COMPLETE! ***\n", level);
    if (level < 3)
    {
        printf("Preparing next level (Larger Grid)...\n");
        printf("Press 1 to continue: ");
        scanf("%d", &choice);
    }
}

printf("\n===== \n");
printf("  CAMPAIGN VICTORY! YOU WON!   \n");
printf("===== \n");
}
```

FINAL GRADE

25/30

GENERAL COMMENTS

WORKSHEET 5

Coding- Really good effort with advanced features such as randomising the board and different levels. Excellent work. The only thing I'd say is, you could streamline your program through the use of function. You could move the scoring to a function, for example.
13/15

Testing - Could have shown that coordinates cannot overlap (or discussed it at least!) Could have been a little more extensive in the tests shown given the complexity of the code, to show it works in all situations.

12/15

Overall mark Worksheet 5: 25/30

Organisation of report, writing style, presentation of work.

Very good overall. Occasionally just a little hard to follow your English expression, but I appreciate that it is your second language :) Very good effort on using the passive writing style.

9/10

Overall mark for assignment (5 worksheets) -
82.5%

Test mark - 93%

I've been really impressed with your progress during this module - well done.

Overall mark for module: 86%

PAGE 1
PAGE 2
PAGE 3
PAGE 4
PAGE 5
PAGE 6
PAGE 7
PAGE 8