



Computer Organization & Assembly Language

Final Project

Number Crush Game

Submitted To: Mr Muhammad Abid Hussain

Submitted By:

- 1) Malik Moiz
- 2) Sheryar Baloch
- 3) Heera Kareem

Contents

Serial No	Topics	Page no
1	Introduction	3
2	Functions	
3	Irvine and Masm installation	
4	Output Demo	

Computer Organization & Assembly Language

Final Project

Number Crushing Game

We have used MASM 32 bit code in Visual Studio and sole purpose of this was to get access to the Irvine Library which generates random numbers that will be used in this project.

Game Description

Number Crush is a "match-three" game, where the core game play is based on swapping two adjacent random values among several on the game board to make a row or column of at least 3 matching-random values. On this match, the matched random values are removed from the board, and random values above them fall into the empty spaces, with new random values appearing from the top of the board. This may create a new matched set of random values, which is automatically cleared in the same manner. The game is split among many levels, which must be completed in sequence.

LEVEL 1

Level 1 has a 10x10 board. When a number is swapped with another number, if a combo exists, the combo is crushed, dropped, and score updated accordingly. Otherwise, the numbers are swapped back! The board is filled with random numbers from 1 to 5. It has bomb 'B' too. When a number is swapped with bomb, all of its occurrences are destroyed.



How to Update SCORE AND MOVES

- 1) During crushing, the score added depends on the size of combo. A combo of 3 adds 3 to the score. A combo of 4 will add 4 and so on.
- 2) During explosion, it is different though. The added score depends on how many occurrences are destroyed and from which location they are destroyed. If a number is at bottom, more numbers will have to be dropped from top and hence more score.
- 3) The user is given a total of 15 moves in each level.

File Handling

- All Individual levels score will be saved in the file.
- Stores the highest score and player name in a same file.
- Record in a file should look like in the format given below

Malik Moiz

Level 1: 20

Highest Score: 20

Code:

```
28
29 ;-----FILE HANDLING-----;
30
31 filename byte "data.txt",0
32 filehandle dword ?
33 buffer byte 30 dup(?)
34 file_cr byte "File is created ",0
35 file_cr1 byte "File is not created",0
36 file_l1 byte " : LEVEL 1 : ",0
37 file_l2 byte " : LEVEL 2 : ",0
38 file_l3 byte " : LEVEL 3 : ",0
39 noofbytes dword ?
40 file_score dword 10 dup("0")
41 ;-----END OF FILE HANDLING-----;
```

BONUSES

- When bomb is used, all occurrences of the exploding row/col are first highlighted for a second, then explosion proceeds.
- If after swapping, no combo exists, the numbers are swapped back.
- The string 'crushing' is displayed when combos are being crushed and score is being updated.
- 'Explosion' is displayed when a bomb destroys a row or col in board.

Divide and Conquer

We have written different procedures to help ourself out in this project. We have used some of these procedures for performing the game. We have also written some extra procedures for complete functionality. The main method consists of only calling other procedures.

populateBoard (.)

It populates the board using random numbers.

drawBoard (.)

This includes drawing border.

updateBoard (.)

This function checks for combinations, crushes them, auto-fills and drops until all the combos are removed from the array. It makes use of another function **checkCombo** which is called in a loop unless and until the value of 'crush' variable is set to 0. 0 means there are no more combinations in our board array. Vertical and horizontal combos are checked.

takeInput (.)

It takes input for two cells. Then if cells are adjacent, they are swapped. If no combo is formed after swapping, the no's are swapped back.

initiateBomb (.)

If any no. is swapped with bomb, it destroys all values of that row or column through which number is swapped by bomb.

drawString (.)

This function draws the player's name, score and moves left on the top. Along with this, it also shows the current level, and strings of 'explosion' and 'crushing'.

Array to initiate 10*10 board with zeros:

```
42 ;-----Initialization-----;
43 game_arr db 10 dup(0)
44          db 10 dup(0)
45          db 10 dup(0)
46          db 10 dup(0)
47          db 10 dup(0)
48          db 10 dup(0)
49          db 10 dup(0)
50          db 10 dup(0)
51          db 10 dup(0)
52          db 10 dup(0)
53
54 row db 10
55 col db 10
56
57 ;-----End Initialization-----;
58
```

Data section:

```
14
15 .data
16 ;-----Guiding strings-----;
17 cr_str db "-----CRUSH ROW-----",0
18 cc_str db "-----CRUSH COL-----",0
19
20 ;level_3 db "-----Level 3 started-----",0
21 ;level_2 db "-----Level 2 Started-----",0
22 level_1 db "-----Level 1 Started-----",0
23
24 rand_str db "-----PERC UP-----",0
25 fill_rand db "-----RANDOM NUMBER-----",0
26 bombx db "-----BOMB-----",0
27
28
29 ;-----FILE HANDLING-----;
30
```

Name, Score, Move:

```
58
59 ;-----name-score-moves-----;
60 namex db 30 dup("0")
61 score dword 0
62 moves dword 0
63 name_string db "NAME : ",0
64 score_string db "SCORE : ",0
65 moves_string db "MOVES : ",0
66 info_space db "      ",0
67 ;-----;
68
69
```

Code of input function:

```
82
83
84
85 ;-----Input function Starts-----;
86 row1 db 0
87 col1 db 0
88 row2 db 0
89 col2 db 0
90 block1_str db "SELECT YOUR BOX 1",0
91 row1_str db "Enter Row(1) : ",0
92 col1_str db "Enter Col(1) : ",0
93
94 block2_str db "SELECT BOX TO SWAP",0
95 row2_str db "Enter Row(2) : ",0
96 col2_str db "Enter Col(2) : ",0
97 ;--conditonsblock2--;
98 row_up db 0
99 row_down db 0
100 col_up db 0
101 col_down db 0
102
103 invalid_input db "You've Chosen an Invalid Box",0
104
105 enter_name db "Enter Your Name : ",0
106 swap_str db "SWAPPED SUCCESS",0
107 ;-----Input function Ends-----;
108
```

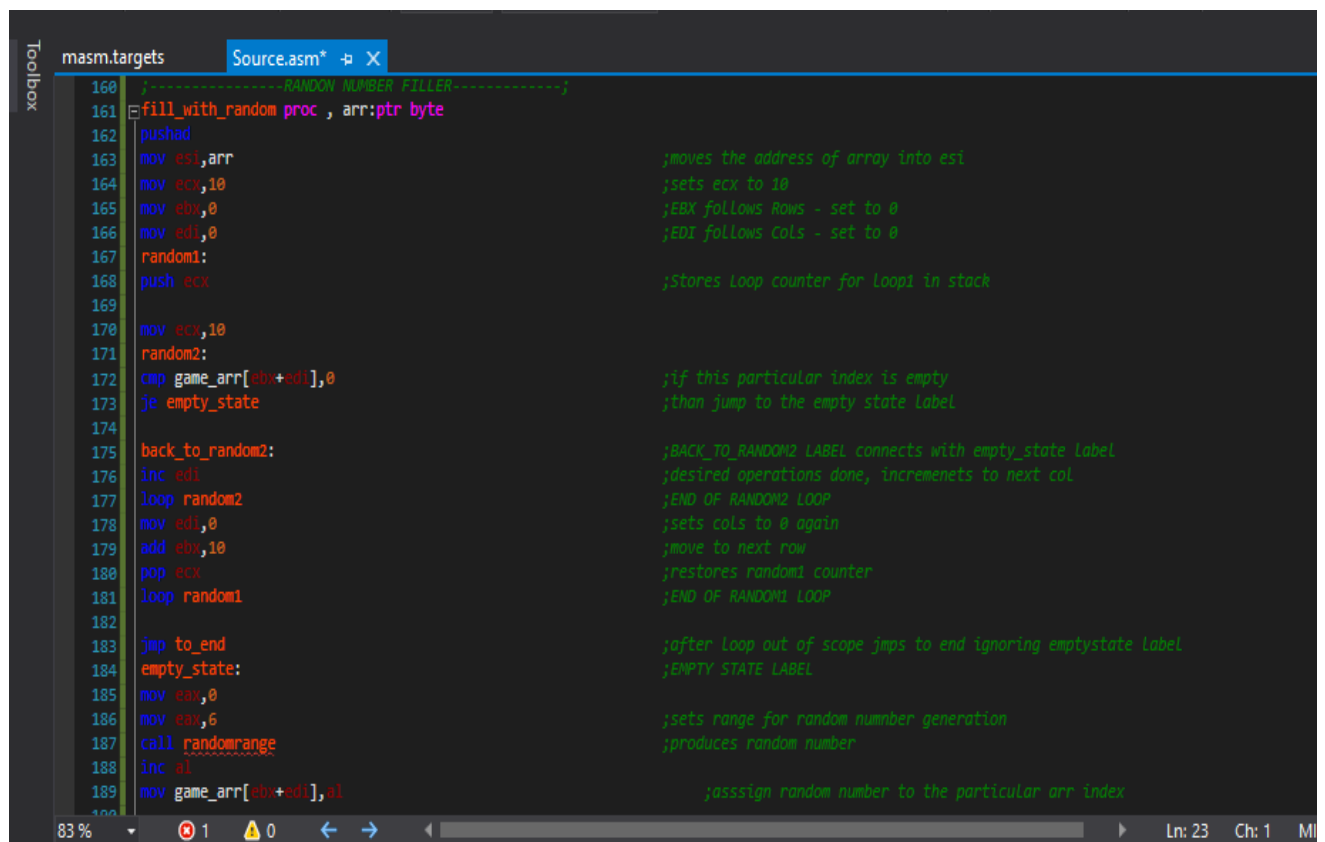
. code:


```

3
4
5 .code
6 main PROC
7
8
9
10
11 call driver_level1
12
13 call WaitMsg
14 exit
15 main ENDP
16
17
18
19

```

To fill crushed boxes with random numbers:



```

masm.targets Source.asm* X
160 ;-----RANDOM NUMBER FILLER-----;
161 fill_with_random proc , arr:ptr byte
162 pushad
163 mov esi,arr ;moves the address of array into esi
164 mov ecx,10 ;sets ecx to 10
165 mov ebx,0 ;EBX follows Rows - set to 0
166 mov edi,0 ;EDI follows Cols - set to 0
167 random1:
168 push ecx ;Stores loop counter for loop1 in stack
169
170 mov ecx,10
171 random2:
172 cmp game_arr[ebx+edi],0 ;if this particular index is empty
173 je empty_state ;than jump to the empty state label
174
175 back_to_random2: ;BACK_TO_RANDOM2 LABEL connects with empty_state label
176 inc edi ;desired operations done, increments to next col
177 loop random2 ;END OF RANDOM2 LOOP
178 mov edi,0 ;sets cols to 0 again
179 add ebx,10 ;move to next row
180 pop ecx ;restores random1 counter
181 loop random1 ;END OF RANDOM1 LOOP
182
183 jmp to_end ;after loop out of scope jumps to end ignoring emptystate label
184 empty_state: ;EMPTY STATE LABEL
185 mov eax,0
186 mov eax,6 ;sets range for random number generation
187 call randrange ;produces random number
188 inc al
189 mov game_arr[ebx+edi],al ;assign random number to the particular arr index
190

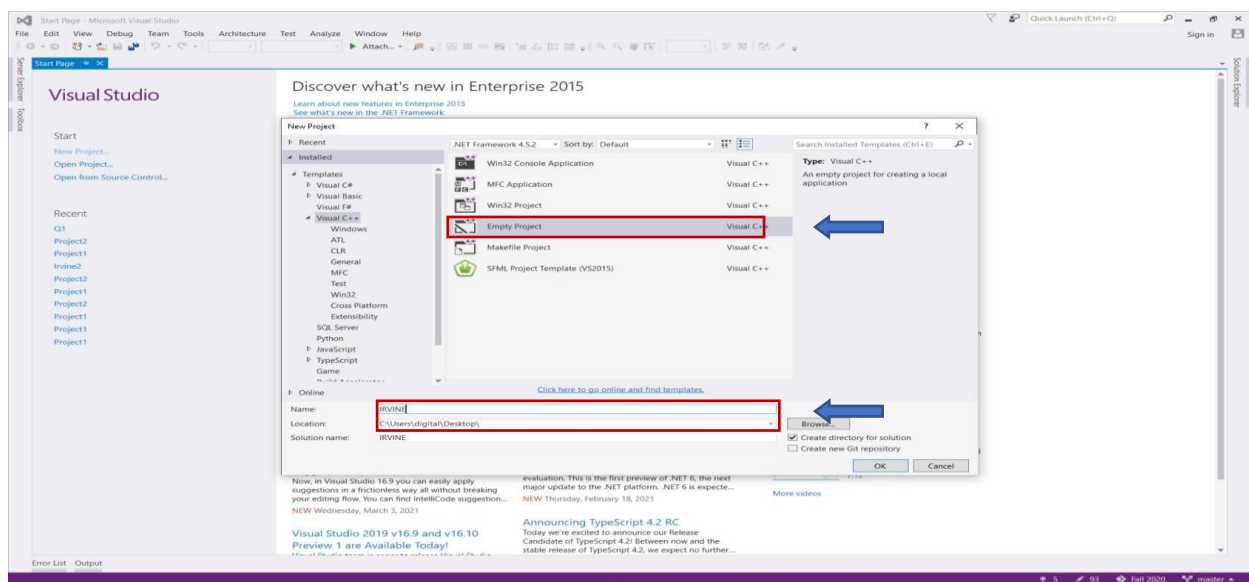
```

Converting 6 consecutive same numbers in a bomb (special move):

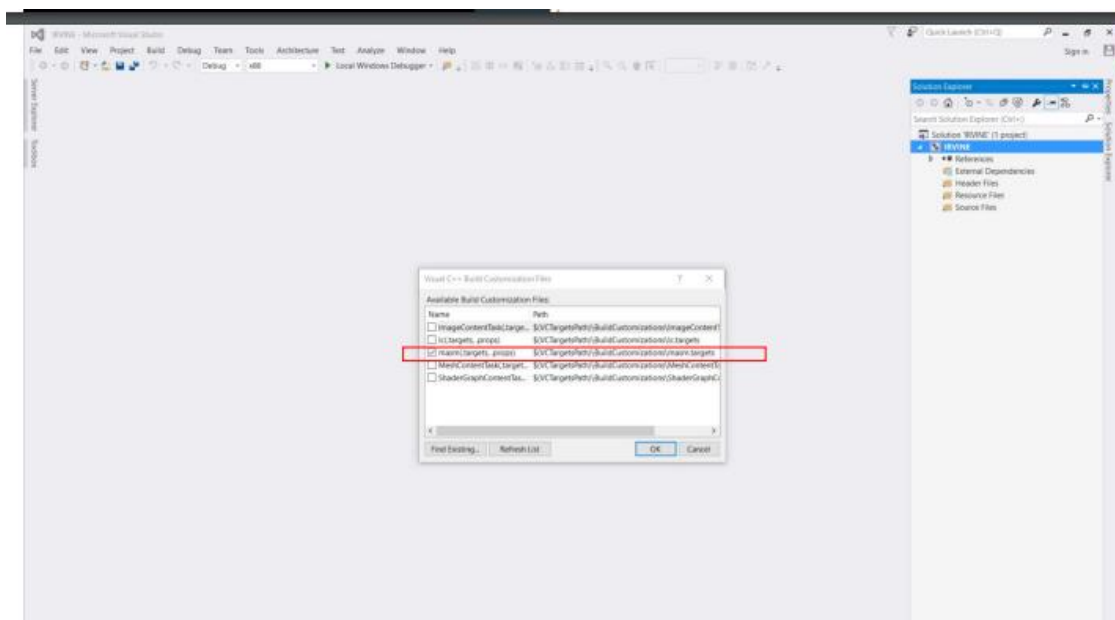
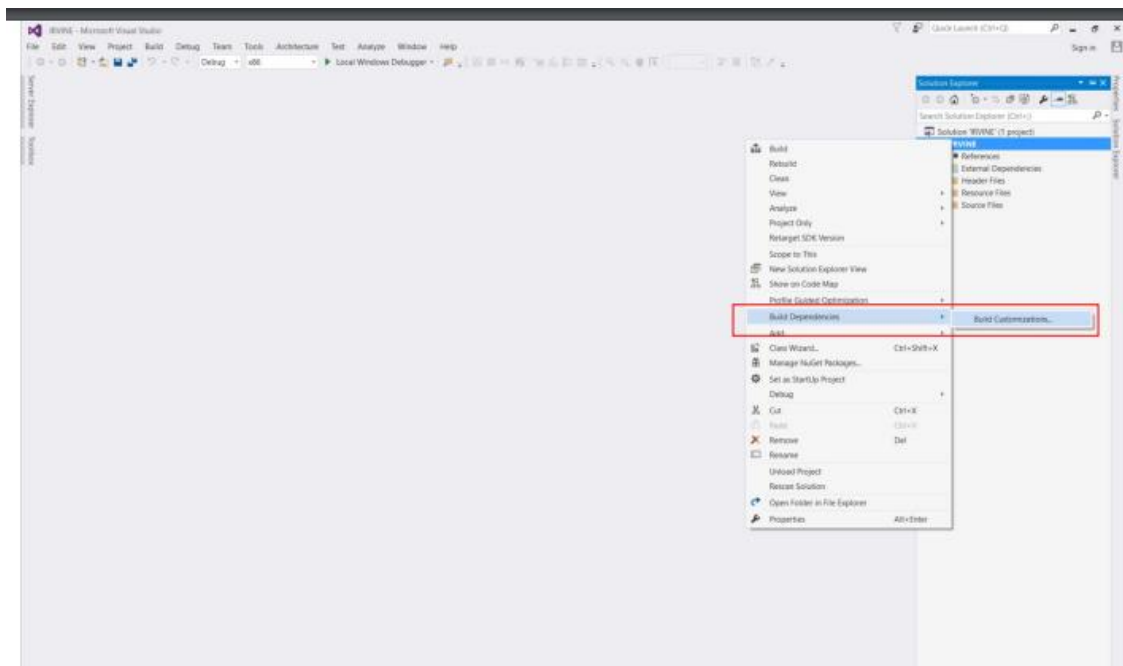
```

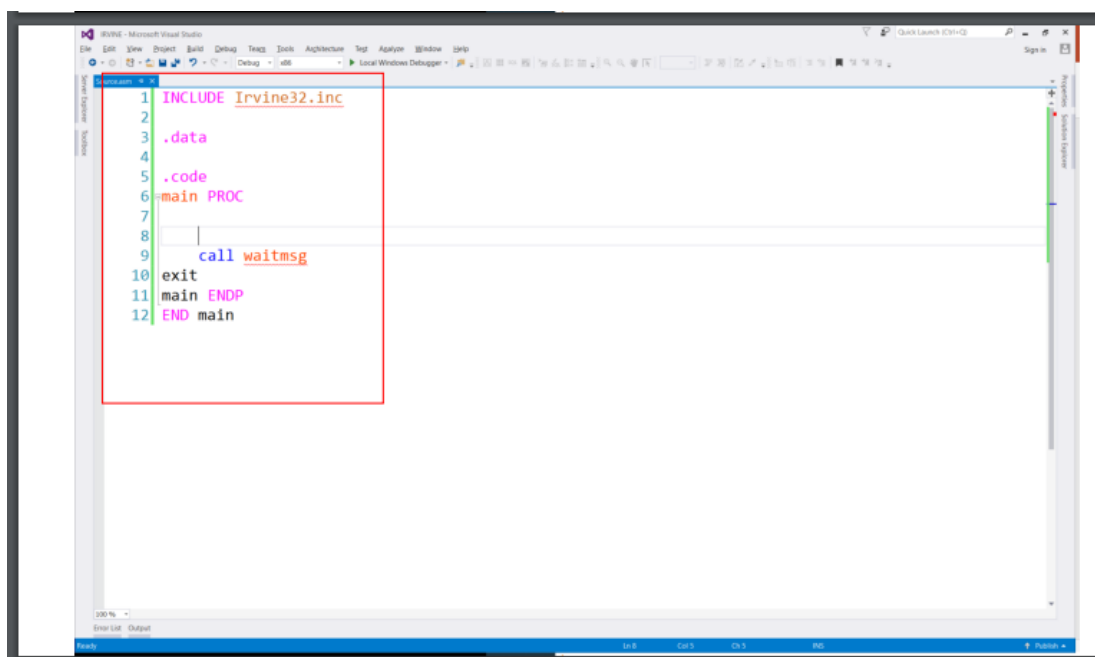
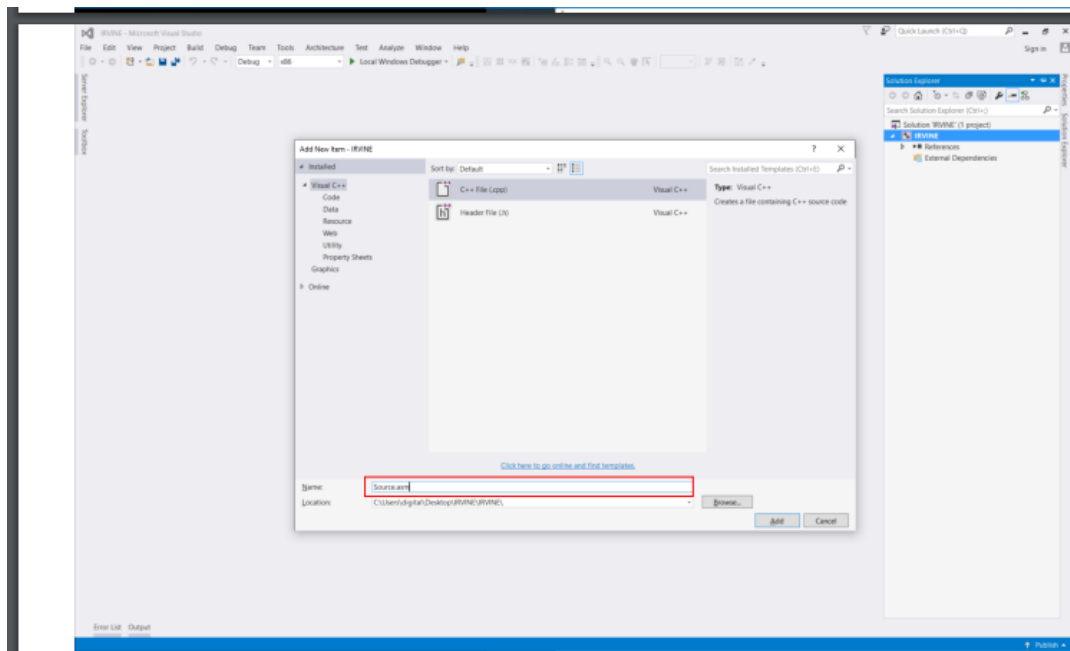
282 ; CONV 0 into bomb in the array
283 ;-----
284 null_occ proc, bomb_num:byte
285 pushad
286
287
288 mov esi,0                ;jest refresh
289 mov edi,0                ;jedi refresh
290 mov ecx,100              ;loop indexes
291
292 l1:                      ;loop l1
293 mov eax,0                ;refresh eax
294 mov al,bomb_num          ;moves num we want to destroy in al
295 cmp game_arr[esi],al     ;game_arr comparison with al
296 je d_c                  ;comes back to the loop
297 back:                    ;BACK LOOP
298 inc esi                  ;increments esi
299 loop l1                  ;dec l1
300
301 jmp to_end
302 d_c:
303 mov game_arr[esi],0
304 inc score
305 jmp back
306 to_end:
307
308 popad
309 ret
310 null_occ endp
311 ;-----END NULL OCCURENCE-----

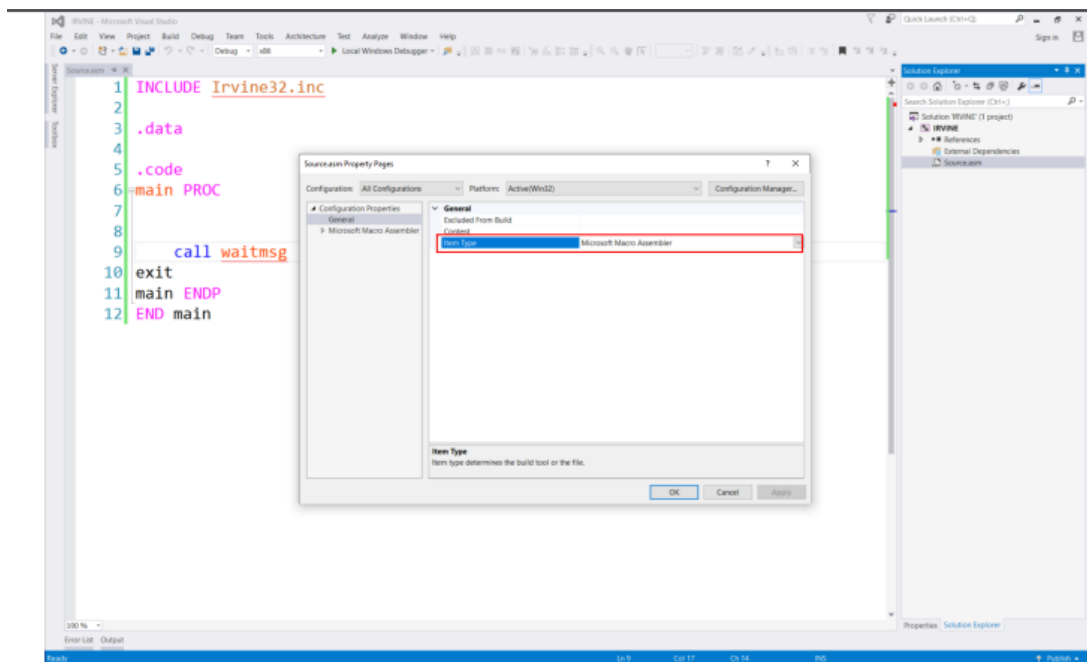
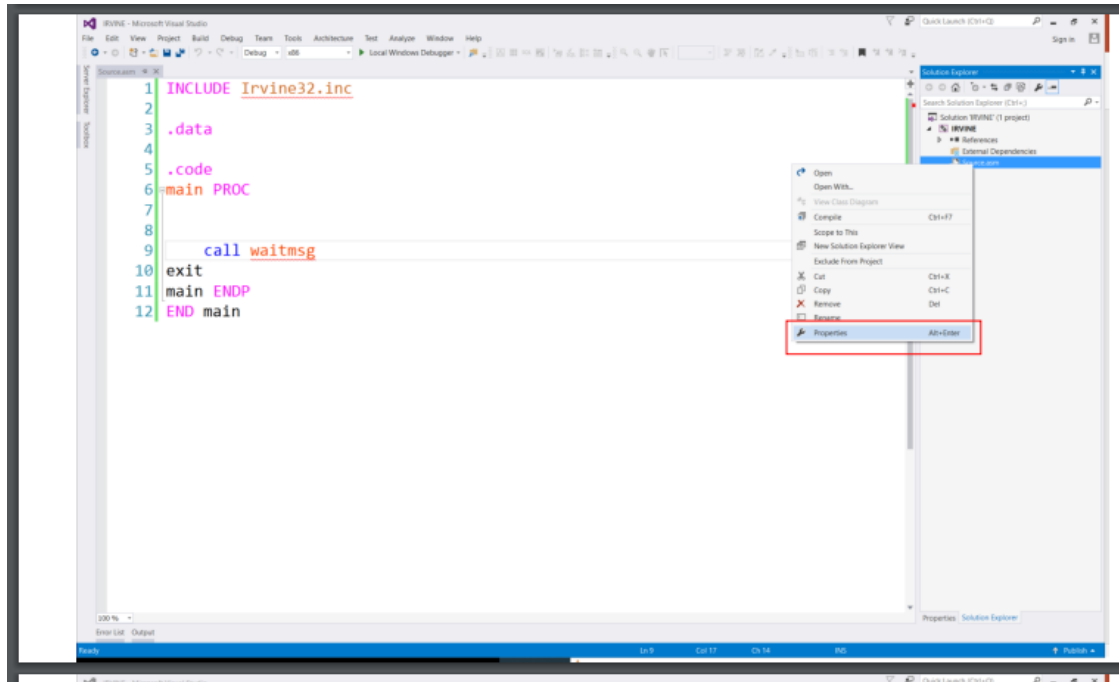
```

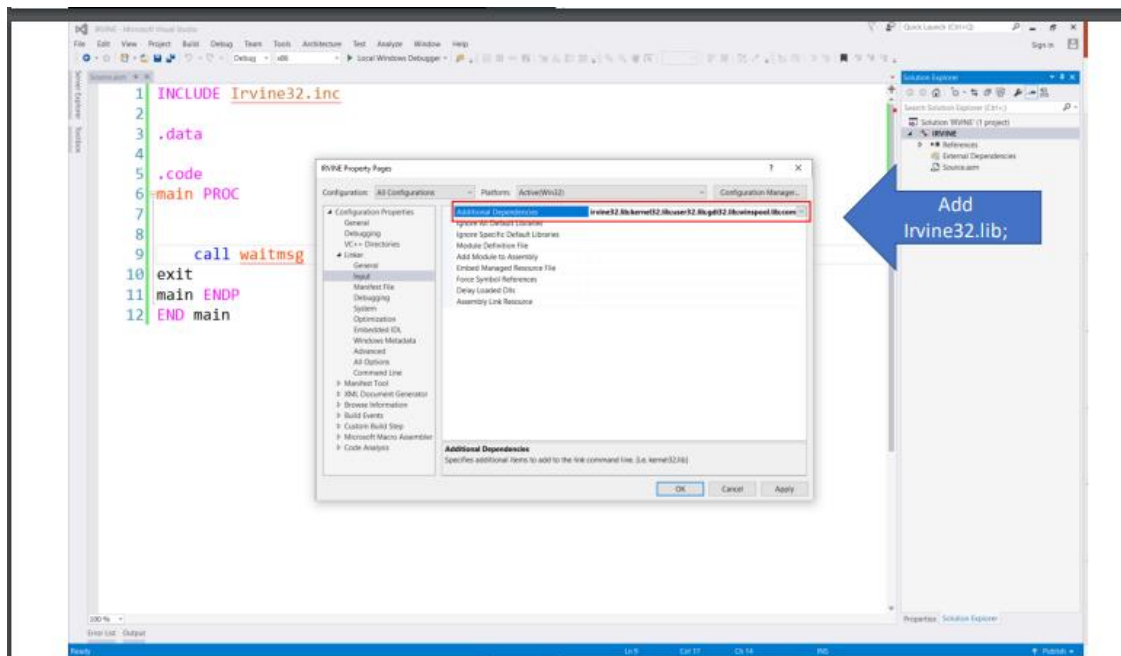
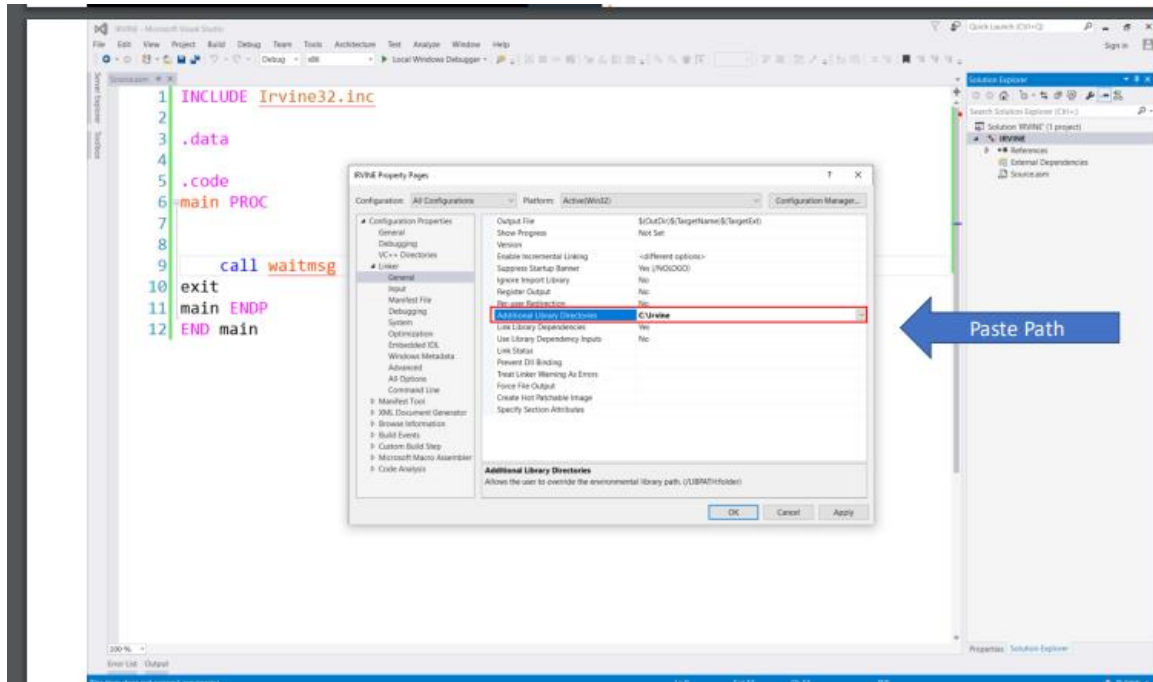


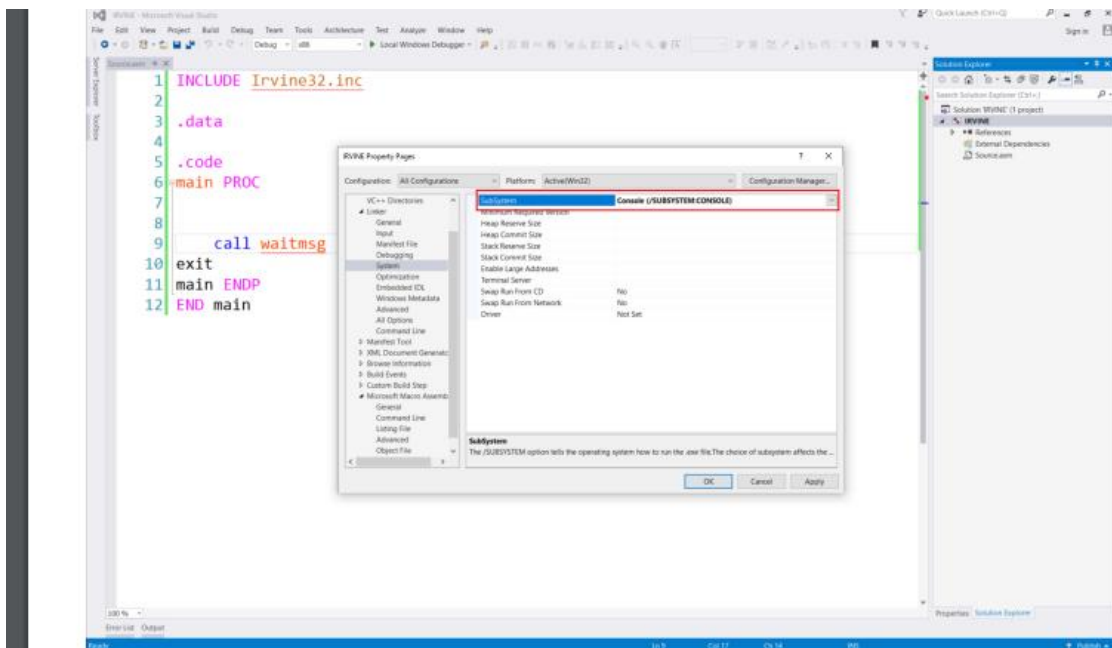
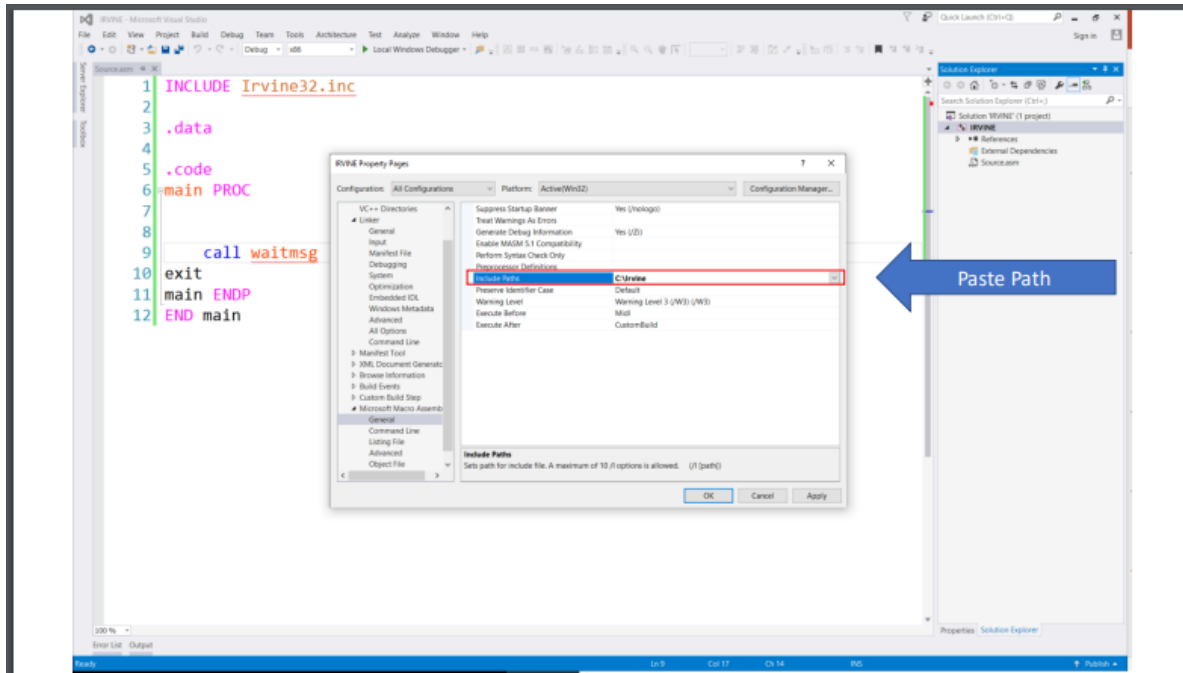
Masm And Irvine Library:

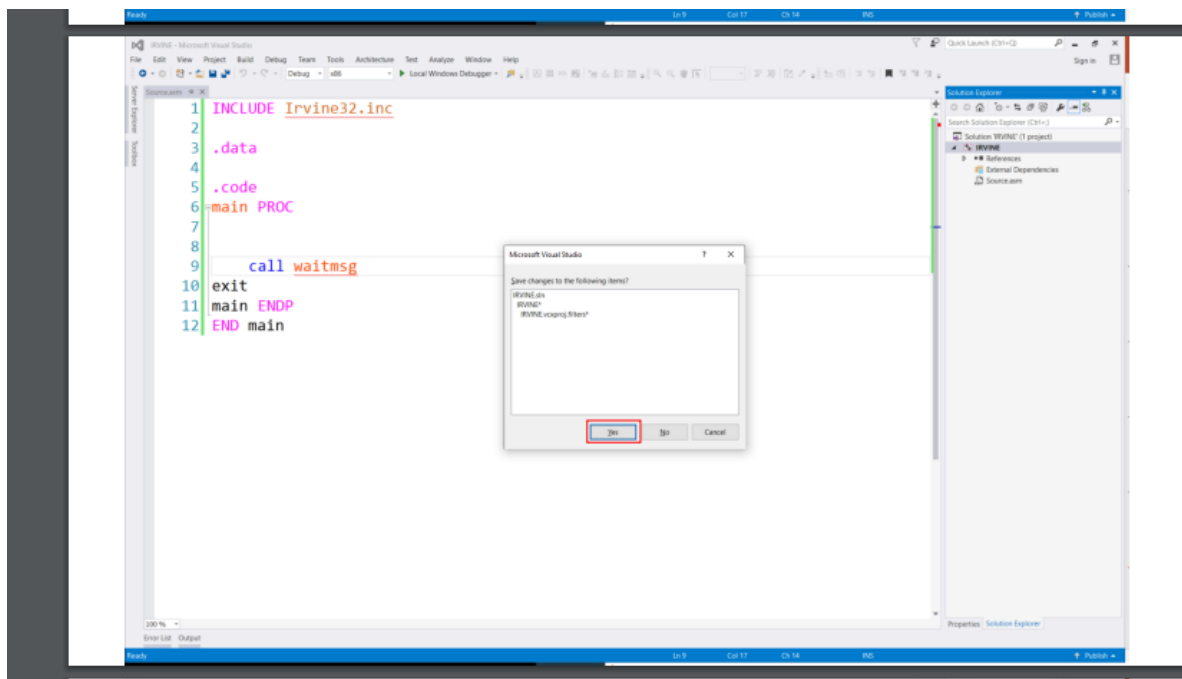
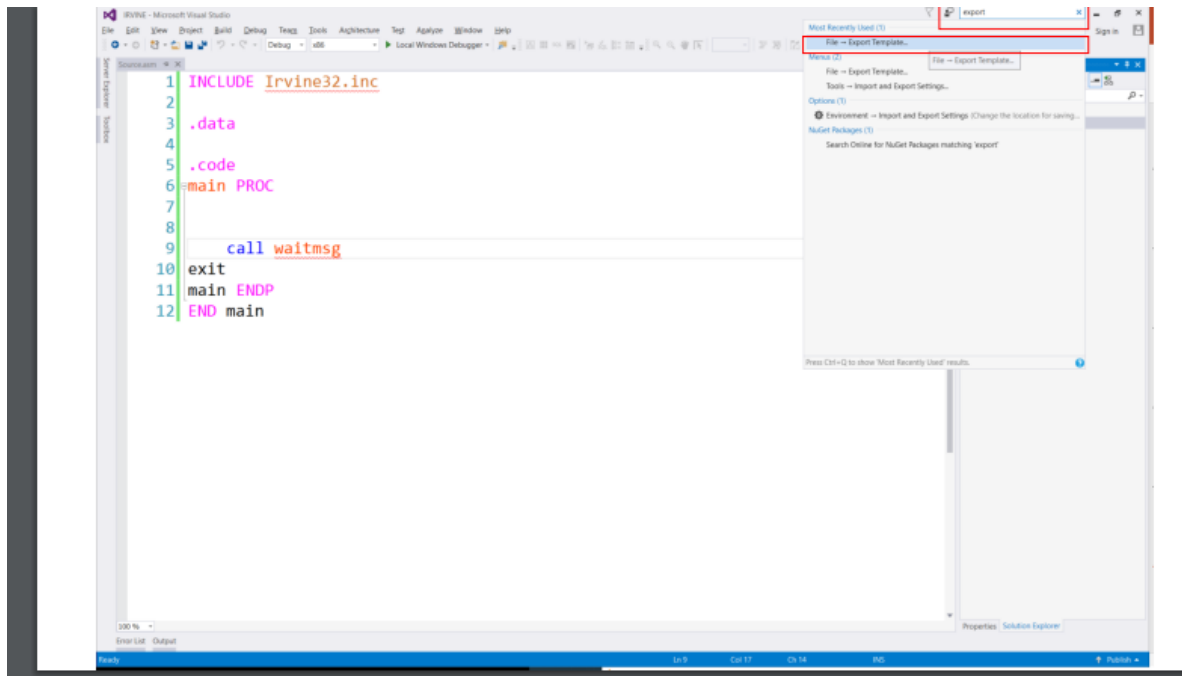


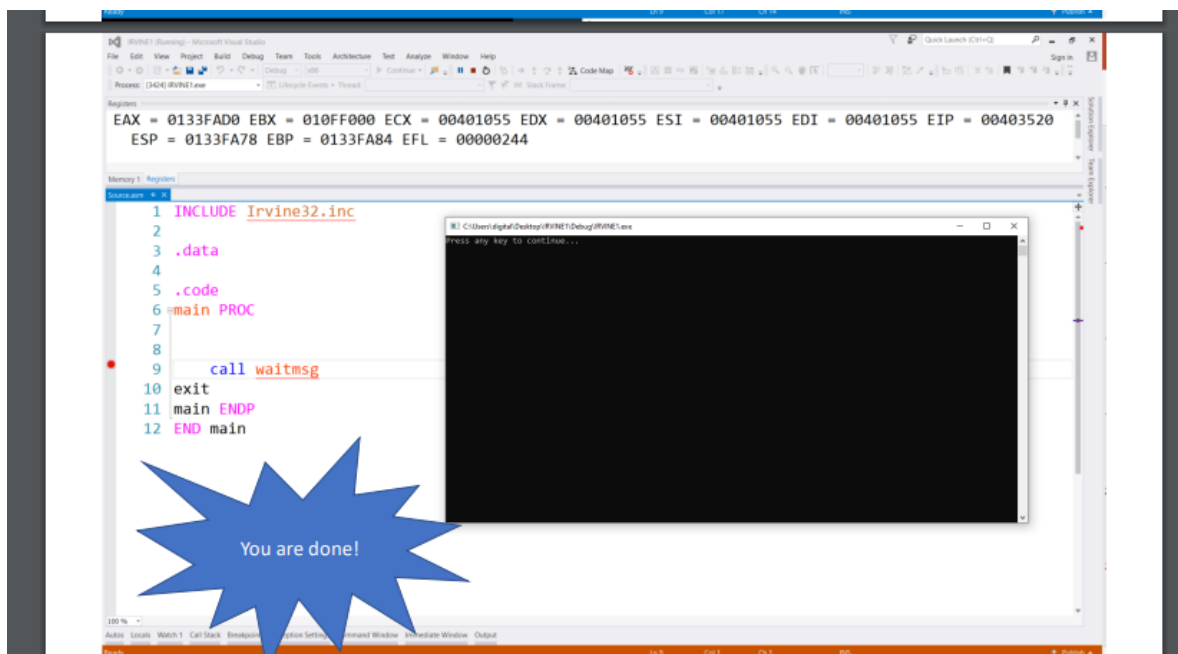
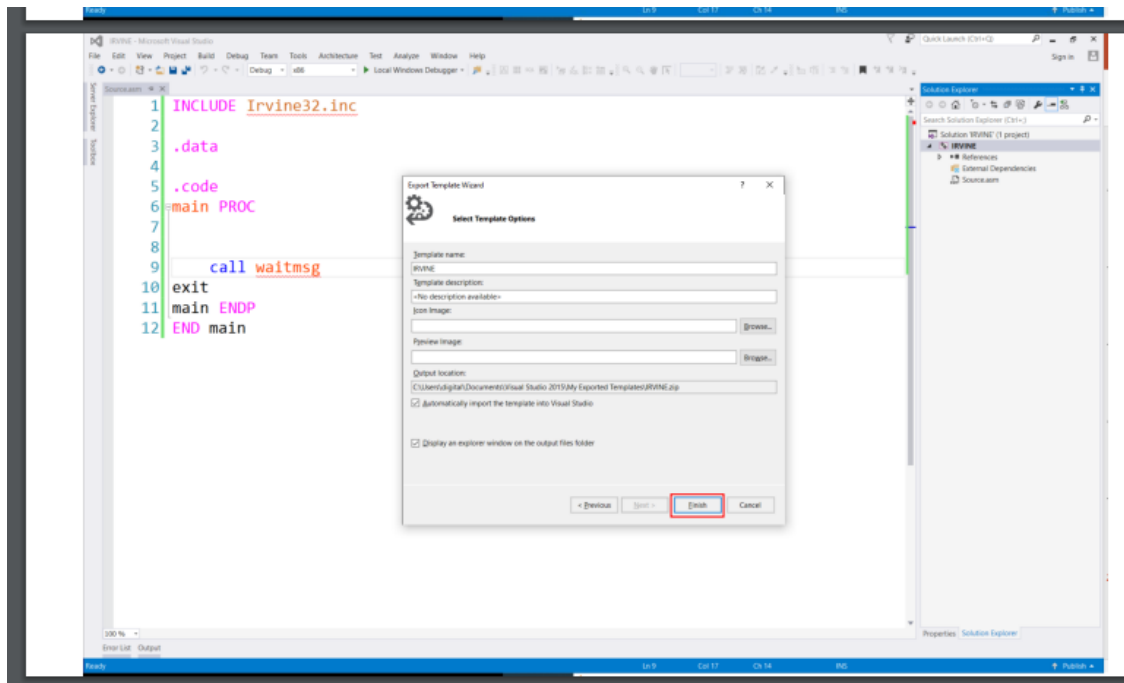




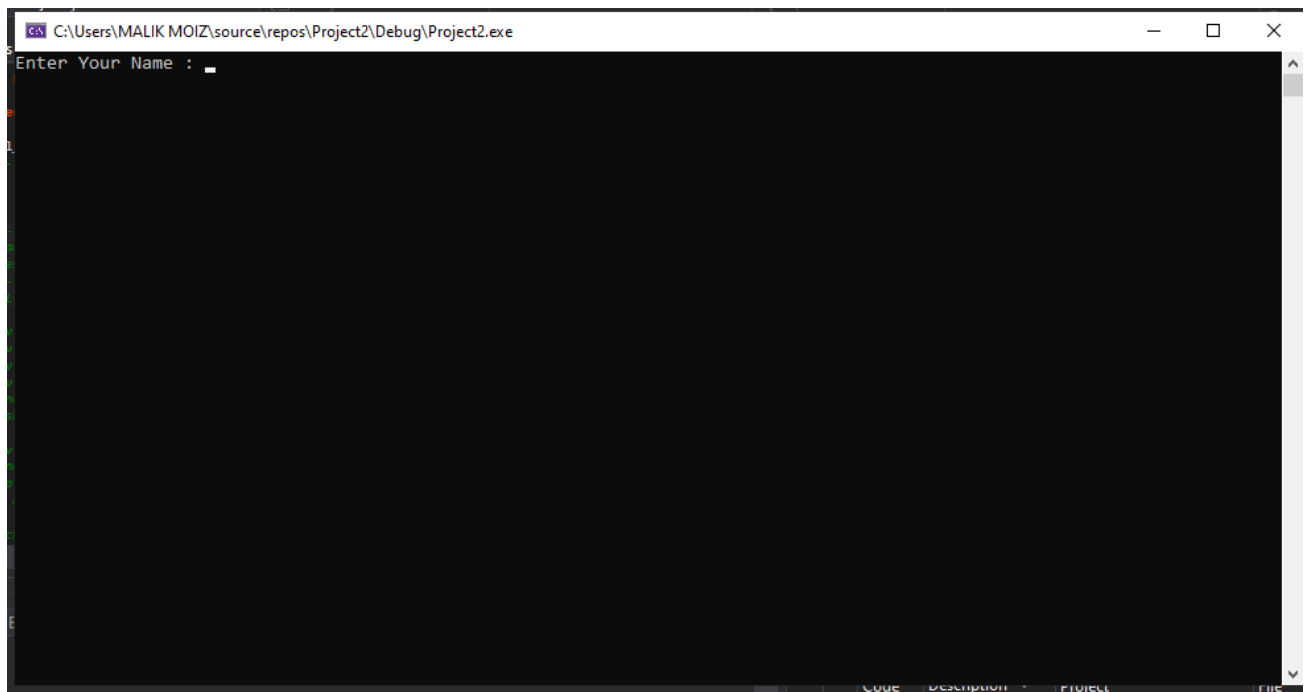




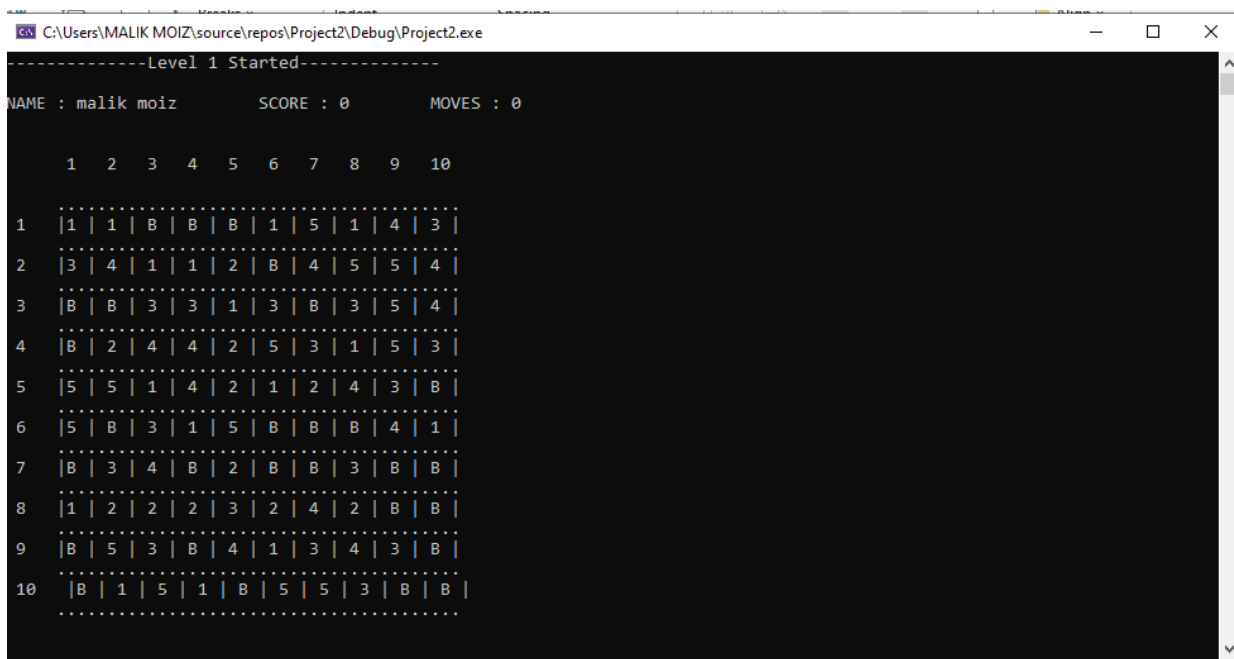




output:



```
C:\Users\MALIK MOIZ\source\repos\Project2\Debug\Project2.exe
Enter Your Name : 
```



```
C:\Users\MALIK MOIZ\source\repos\Project2\Debug\Project2.exe
-----Level 1 Started-----
NAME : malik moiz      SCORE : 0      MOVES : 0

  1  2  3  4  5  6  7  8  9 10
1  |1|1|B|B|B|1|5|1|4|3|
2  |3|4|1|1|2|B|4|5|5|4|
3  |B|B|3|3|1|3|B|3|5|4|
4  |B|2|4|4|2|5|3|1|5|3|
5  |5|5|1|4|2|1|2|4|3|B|
6  |5|B|3|1|5|B|B|B|4|1|
7  |B|3|4|B|2|B|B|3|B|B|
8  |1|2|2|2|3|2|4|2|B|B|
9  |B|5|3|B|4|1|3|4|3|B|
10 |B|1|5|1|B|5|5|3|B|B|
```

```

C:\Users\MALIK MOIZ\source\repos\Project2\Debug\Project2.exe
2 | 3 | 4 | 1 | 1 | 2 | B | 4 | 5 | 5 | 4 |
3 | B | B | 3 | 3 | 1 | 3 | B | 3 | 5 | 4 |
4 | B | 2 | 4 | 4 | 2 | 5 | 3 | 1 | 5 | 3 |
5 | 5 | 5 | 1 | 4 | 2 | 1 | 2 | 4 | 3 | B |
6 | 5 | B | 3 | 1 | 5 | B | B | B | 4 | 1 |
7 | B | 3 | 4 | B | 2 | B | B | 3 | B | B |
8 | 1 | 2 | 2 | 2 | 3 | 2 | 4 | 2 | B | B |
9 | B | 5 | 3 | B | 4 | 1 | 3 | 4 | 3 | B |
10 | B | 1 | 5 | 1 | B | 5 | 5 | 3 | B | B |

SELECT YOUR BOX 1
Enter Row(1) : 1
Enter Col(1) : 3
SELECT BOX TO SWAP
Enter Row(2) : 2
Enter Col(2) : 3
-----BOMB-----
Press any key to continue...

```

Here as we have swapped R1 C3 with R2 C3 a bomb is formed as it's a special power the 3 consecutive 1's along with the bomb is crushed and score is added.

```

C:\Users\MALIK MOIZ\source\repos\Project2\Debug\Project2.exe
-----CRUSH ROW-----
NAME : malik moiz      SCORE : 22      MOVES : 1

  1  2  3  4  5  6  7  8  9 10
1 | 0 | 0 | 0 | B | B | 0 | 5 | 0 | 4 | 3 |
2 | 3 | 4 | 0 | 0 | 2 | B | 4 | 5 | 5 | 4 |
3 | B | B | 3 | 3 | 0 | 3 | B | 3 | 5 | 4 |
4 | B | 2 | 4 | 4 | 2 | 5 | 3 | 0 | 5 | 3 |
5 | 5 | 5 | 0 | 4 | 2 | 0 | 2 | 4 | 3 | B |
6 | 5 | B | 3 | 0 | 5 | B | B | B | 4 | 0 |
7 | B | 3 | 4 | B | 2 | B | B | 3 | B | B |
8 | 0 | 0 | 0 | 0 | 3 | 2 | 4 | 2 | B | B |
9 | B | 5 | 3 | B | 4 | 0 | 3 | 4 | 3 | B |
10 | B | 0 | 5 | 0 | B | 5 | 5 | 3 | B | B |

Press any key to continue...

```

After crushing the numbers the crushed location is initialized with zeros

```
C:\Users\MALIK MOIZ\source\repos\Project2\Debug\Project2.exe
-----RANDOM NUMBER-----
NAME : malik moiz      SCORE : 25      MOVES : 1

  1   2   3   4   5   6   7   8   9  10
1  |3|1|3|2|3|5|5|1|5|1|
2  |5|1|5|1|B|3|4|5|5|3|
3  |3|3|4|1|2|2|B|5|4|4|
4  |B|4|5|B|2|B|3|3|4|4|
5  |B|B|3|B|2|3|2|4|3|3|
6  |5|2|4|3|5|5|B|B|4|B|
7  |5|5|3|4|2|B|B|3|B|B|
8  |B|B|4|4|3|B|4|2|B|B|
9  |B|3|3|B|4|2|3|4|3|B|
10 |B|5|5|B|B|5|5|3|B|B|
Press any key to continue...
```

Zeros are replaced with random numbers by Irvine Library

```
C:\Users\MALIK MOIZ\source\repos\Project2\Debug\Project2.exe
2 | 3 | 4 | 1 | 1 | 2 | B | 4 | 5 | 5 | 4 |
3 | B | B | 3 | 3 | 1 | 3 | B | 3 | 5 | 4 |
4 | B | 2 | 4 | 4 | 2 | 5 | 3 | 1 | 5 | 3 |
5 | 5 | 5 | 1 | 4 | 2 | 1 | 2 | 4 | 3 | B |
6 | 5 | B | 3 | 1 | 5 | B | B | B | 4 | 1 |
7 | B | 3 | 4 | B | 2 | B | B | 3 | B | B |
8 | 1 | 2 | 2 | 2 | 3 | 2 | 4 | 2 | B | B |
9 | B | 5 | 3 | B | 4 | 1 | 3 | 4 | 3 | B |
10 | B | 1 | 5 | 1 | B | 5 | 5 | 3 | B | B |
SELECT YOUR BOX 1
Enter Row(1) : 8
Enter Col(1) : 5
SELECT BOX TO SWAP
Enter Row(2) : 8
Enter Col(2) : 6
```

```
C:\Users\MALIK MOIZ\source\repos\Project2\Debug\Project2.exe
SWAPPED SUCCESS
Press any key to continue...
```

```
C:\Users\MALIK MOIZ\source\repos\Project2\Debug\Project2.exe
-----CRUSH ROW-----
NAME : malik moiz      SCORE : 4      MOVES : 1

      1  2  3  4  5  6  7  8  9  10
1  |1|1|B|B|B|1|5|1|4|3|
2  |3|4|1|1|2|B|4|5|5|4|
3  |B|B|3|3|1|3|B|3|5|4|
4  |B|2|4|4|2|5|3|1|5|3|
5  |5|5|1|4|2|1|2|4|3|B|
6  |5|B|3|1|5|B|B|B|4|1|
7  |B|3|4|B|2|B|B|3|B|B|
8  |1|0|0|0|0|3|4|2|B|B|
9  |B|5|3|B|4|1|3|4|3|B|
10 |B|1|5|1|B|5|5|3|B|B|

Press any key to continue...
```

```
C:\Users\MALIK MOIZ\source\repos\Project2\Debug\Project2.exe
-----PERC UP-----
NAME : malik moiz      SCORE : 7      MOVES : 1

      1  2  3  4  5  6  7  8  9  10
1  |1|0|0|0|0|1|5|1|0|3|
2  |3|1|B|B|B|B|4|5|0|4|
3  |B|4|1|1|2|3|B|3|0|4|
4  |B|B|3|3|1|5|3|1|4|3|
5  |5|2|4|4|2|1|2|4|3|B|
6  |5|5|1|4|2|B|B|B|4|1|
7  |B|B|3|1|5|B|B|3|B|B|
8  |1|3|4|B|2|3|4|2|B|B|
9  |B|5|3|B|4|1|3|4|3|B|
10 |B|1|5|1|B|5|5|3|B|B|

Press any key to continue...
```



```
Project2.exe
C:\Users\MALIK MOIZ\source\repos\Project2\Debug\Project2.exe

1 | 1 | 3 | 5 | 5 | 3 | 4 | 5 | 1 | 2 | 3 |
2 | 3 | 1 | 1 | 3 | 3 | B | 4 | 5 | 5 | 1 |
3 | B | 4 | B | B | 5 | 3 | B | 3 | 5 | 1 |
4 | B | B | 3 | 3 | 5 | 5 | 3 | 1 | 3 | 3 |
5 | 5 | 2 | 4 | 4 | B | 1 | 2 | 4 | 3 | B |
6 | 5 | 5 | 1 | 4 | 5 | B | B | B | 4 | 1 |
7 | B | B | 3 | 1 | 2 | B | B | 3 | B | B |
8 | 1 | 3 | 4 | B | 3 | 2 | 4 | 2 | B | B |
9 | B | 5 | 3 | B | 4 | 1 | 3 | 4 | 3 | B |
10 | B | 1 | 5 | 1 | B | 5 | 5 | 3 | B | B |

SELECT YOUR BOX 1
Enter Row(1) :
Enter Col(1) :
You've Chosen an Invalid Box
SELECT YOUR BOX 1
Enter Row(1) :
```

The End!

