**Manual test – documentation**

**Reading and saving files functions tests:**

Every test was performed on given file:

**Input.txt**

15 16

000 000 100 400 320 320 000 000 000 000 000 200 100 000 100 030  
120 000 000 010 320 300 300 000 000 000 020 100 330 000 100 000  
200 000 130 000 000 000 030 000 000 000 200 000 100 000 500 100  
030 000 010 000 000 000 000 100 000 100 500 000 000 000 100 000  
010 000 000 000 100 530 000 000 100 010 010 000 100 000 500 300  
110 000 000 000 000 300 120 320 000 000 420 000 000 020 020 000  
000 000 000 000 000 000 000 000 400 300 000 000 100 000 000 000  
400 100 020 000 000 400 000 000 430 000 000 000 420 100 000 000  
000 000 000 300 030 000 200 020 400 000 200 030 000 100 000 400  
000 000 000 000 220 000 020 030 000 020 000 000 000 300 200 520  
000 000 000 000 030 200 320 100 400 300 110 100 020 000 000 000  
200 000 500 000 400 000 300 000 200 030 030 200 000 310 100 400  
330 100 020 010 020 300 200 000 000 200 300 020 000 400 230 000  
200 030 000 000 400 400 030 200 130 000 000 130 000 000 100 000  
000 000 000 000 500 110 300 020 020 020 100 000 100 000 000 010

Dummy 1 0

**Test 1: return get\_multi\_digit\_file (reading height and width from Input.txt)**

Expected: 15 and 16 Return: 15, 16 **PASSED**

**Test 2: return get\_tile\_file (tile [5][7])**

Expected: 320 Return: 320 **PASSED**

**Test 3: get\_board\_file (Input.txt - board)**

Expected: Saved board from Input.txt in game\_system structure

Return:

000 000 100 400 320 320 000 000 000 000 000 200 100 000 100 030  
120 000 000 010 320 300 300 000 000 000 020 100 330 000 100 000  
200 000 130 000 000 000 030 000 000 000 200 000 100 000 500 100  
030 000 010 000 000 000 000 100 000 100 500 000 000 000 100 000  
010 000 000 000 100 530 000 000 100 010 010 000 100 000 500 300  
110 000 000 000 000 300 120 320 000 000 420 000 000 020 020 000  
000 000 000 000 000 000 000 000 400 300 000 000 100 000 000 000  
400 100 020 000 000 400 000 000 430 000 000 000 420 100 000 000  
000 000 000 300 030 000 200 020 400 000 200 030 000 100 000 400  
000 000 000 000 220 000 020 030 000 020 000 000 000 300 200 520  
000 000 000 000 030 200 320 100 400 300 110 100 020 000 000 000  
200 000 500 000 400 000 300 000 200 030 030 200 000 310 100 400  
330 100 020 010 020 300 200 000 000 200 300 020 000 400 230 000  
200 030 000 000 400 400 030 200 130 000 000 130 000 000 100 000  
000 000 000 000 500 110 300 020 020 020 100 000 100 000 000 010

**PASSED**

**Test 4: get\_player\_data\_file(Input.txt)**

Expected: Saved player data form Input.txt in game\_system structure

Return: Dummy 1 0

**PASSED**

**Test 5: add\_user\_initation(Input.txt)**

Expected: Added in player\_database program as new player (DoubleA 2 0)

Return:

Dummy 1 0  
DoubleA 2 0

**PASSED**

**Test 6: read file(Input.txt)**

Expected: Saved in game\_system board and player\_data from the file and new player (DoubleA)

Return:

15 16

000 000 100 400 320 320 000 000 000 000 000 200 100 000 100 030  
120 000 000 010 320 300 300 000 000 000 020 100 330 000 100 000  
200 000 130 000 000 000 030 000 000 000 200 000 100 000 500 100  
030 000 010 000 000 000 000 100 000 100 500 000 000 000 100 000  
010 000 000 000 100 530 000 000 100 010 010 000 100 000 500 300  
110 000 000 000 000 300 120 320 000 000 420 000 000 020 020 000  
000 000 000 000 000 000 000 000 400 300 000 000 100 000 000 000  
400 100 020 000 000 400 000 000 430 000 000 000 420 100 000 000  
000 000 000 300 030 000 200 020 400 000 200 030 000 100 000 400  
000 000 000 000 220 000 020 030 000 020 000 000 000 300 200 520  
000 000 000 000 030 200 320 100 400 300 110 100 020 000 000 000  
200 000 500 000 400 000 300 000 200 030 030 200 000 310 100 400  
330 100 020 010 020 300 200 000 000 200 300 020 000 400 230 000  
200 030 000 000 400 400 030 200 130 000 000 130 000 000 100 000  
000 000 000 000 500 110 300 020 020 020 100 000 100 000 000 010

Dummy 1 0  
DoubleA 2 0

**PASSED**

From now on this data is stored now in program in game\_system structure in GS

**Test 7: add\_size\_file(GS)**

Expected: Height and Width from GS

Return: 15 16

**PASSED**

**Test 8: add\_tile\_file**

Expected: Board from GS

Return:

000 000 100 400 320 320 000 000 000 000 000 200 100 000 100 030  
120 000 000 010 320 300 300 000 000 000 020 100 330 000 100 000  
200 000 130 000 000 000 030 000 000 000 200 000 100 000 500 100  
030 000 010 000 000 000 000 100 000 100 500 000 000 000 100 000  
010 000 000 000 100 530 000 000 100 010 010 000 100 000 500 300  
110 000 000 000 000 300 120 320 000 000 420 000 000 020 020 000  
000 000 000 000 000 000 000 000 400 300 000 000 100 000 000 000  
400 100 020 000 000 400 000 000 430 000 000 000 420 100 000 000  
000 000 000 300 030 000 200 020 400 000 200 030 000 100 000 400  
000 000 000 000 220 000 020 030 000 020 000 000 000 300 200 520  
000 000 000 000 030 200 320 100 400 300 110 100 020 000 000 000  
200 000 500 000 400 000 300 000 200 030 030 200 000 310 100 400  
330 100 020 010 020 300 200 000 000 200 300 020 000 400 230 000  
200 030 000 000 400 400 030 200 130 000 000 130 000 000 100 000  
000 000 000 000 500 110 300 020 020 020 100 000 100 000 000 010

**PASSED**

**Test 9: add\_player\_data\_fike**

Expected: player\_data from GS saved in file

Return:

Dummy 1 0  
DoubleA 2 0

**PASSED**

**Test 10: save\_data\_file(GS)**

Expected: game\_state from the GS saved in a file

Return:

15 16

000 000 100 400 320 320 000 000 000 000 000 200 100 000 100 030  
120 000 000 010 320 300 300 000 000 000 020 100 330 000 100 000  
200 000 130 000 000 000 030 000 000 000 200 000 100 000 500 100  
030 000 010 000 000 000 000 100 000 100 500 000 000 000 100 000  
010 000 000 000 100 530 000 000 100 010 010 000 100 000 500 300  
110 000 000 000 000 300 120 320 000 000 420 000 000 020 020 000  
000 000 000 000 000 000 000 000 400 300 000 000 100 000 000 000  
400 100 020 000 000 400 000 000 430 000 000 000 420 100 000 000  
000 000 000 300 030 000 200 020 400 000 200 030 000 100 000 400  
000 000 000 000 220 000 020 030 000 020 000 000 000 300 200 520  
000 000 000 000 030 200 320 100 400 300 110 100 020 000 000 000  
200 000 500 000 400 000 300 000 200 030 030 200 000 310 100 400  
330 100 020 010 020 300 200 000 000 200 300 020 000 400 230 000  
200 030 000 000 400 400 030 200 130 000 000 130 000 000 100 000  
000 000 000 000 500 110 300 020 020 020 100 000 100 000 000 010

Dummy 1 0  
DoubleA 2 0

**PASSED**

**Automated move in Gamemaster.exe tests:**

**Test 1: placement phase**

Expected: Our program will find on board tiles with horses and put pawns in straight path to them.

000 100 200 300 000 000 000 000 000 300 000 100 000 200  
000 100 000 000 200 020 000 100 020 300 000 020 000 200  
000 100 100 020 000 230 000 000 300 000 000 020 300 100  
020 100 000 020 000 000 200 000 010 030 100 000 100 100  
000 420 220 000 000 200 200 000 300 000 300 200 020 500  
100 100 410 400 030 000 200 000 000 000 200 200 000 300  
000 000 000 400 000 000 300 410 030 200 000 300 100 010  
100 030 230 000 000 300 000 500 100 100 400 000 020 000  
100 100 300 000 000 000 000 200 000 400 020 200 200 300  
000 420 030 000 010 030 100 000 430 200 400 100 500 300  
000 000 000 300 200 000 000 500 000 000 100 000 000 000  
300 000 020 100 020 400 100 300 000 000 020 000 300 000  
120 010 100 000 020 020 100 020 300 100 000 000 330 000  
000 200 000 000 000 300 100 000 000 000 000 400 000 500

Return: Program in three moves put three pawns in line to horse artefact

000 100 200 300 000 000 000 000 000 300 000 100 000 200  
000 100 000 000 200 020 000 100 020 300 000 020 000 200  
000 100 100 020 000 230 000 000 001 000 000 020 300 100  
020 100 000 020 000 000 200 000 010 030 100 000 100 100  
000 420 001 000 000 200 200 000 300 000 300 200 020 500  
100 100 410 400 030 000 200 001 000 000 200 200 000 300  
000 000 000 400 000 000 300 410 030 200 000 300 100 010  
100 030 230 000 000 300 000 500 100 100 400 000 020 000  
100 100 300 000 000 000 000 200 000 400 020 200 200 300  
000 420 030 000 010 030 100 000 430 200 400 100 500 300  
000 000 000 300 200 000 000 500 000 000 100 000 000 000  
300 000 020 100 020 400 100 300 000 000 020 000 300 000  
120 010 100 000 020 020 100 020 300 100 000 000 330 000  
000 200 000 000 000 300 100 000 000 000 000 400 000 500

**PASSED**

**Test 2: movement phase**

Expected: Program will move one of the pawn form the previous board to get horse artefact as long as it is possible and after it use last move to get most points possible

Return:

000 100 200 300 000 000 000 000 000 300 000 100 000 200  
000 100 000 000 200 020 000 100 020 300 000 020 000 200  
000 100 100 020 000 230 000 000 001 000 000 020 300 100  
020 100 000 020 000 000 200 000 010 030 100 000 100 100  
000 420 001 000 000 200 200 000 300 000 300 200 020 500  
109 100 000 400 030 000 200 000 000 000 200 200 000 300  
000 000 000 400 000 000 300 410 030 200 000 300 100 010  
100 030 230 000 000 300 000 500 100 100 400 000 020 000  
100 100 300 000 000 000 000 200 000 400 020 200 200 300  
000 420 030 000 010 030 100 000 430 200 400 100 500 300  
009 000 000 300 200 000 000 001 000 000 100 000 000 000  
300 000 020 100 020 400 100 300 000 000 020 000 300 000  
120 010 100 000 020 020 100 020 300 100 000 000 330 000  
000 200 000 000 000 300 100 000 000 000 000 400 000 500

Program moved one pawn and shoot multiple arrows and got available tile with most points

**PASSED**

**AC (Anti-Cheat) system tests:**

Every test was performed on exemplary files:

**Input.txt**

3 3

009 209 002  
320 009 001  
000 002 000

Player1 1 -3  
DoubleA 2 8

**AC.txt**

3 3

001 209 002  
320 002 500  
000 000 000

Player1 1 0  
DoubleA 2 0

**Test 1: return value of ac\_used\_horses**

Expected: 0 Return: 0 **PASSED**

**Test 2: return value of ac\_used\_broken\_arrows**

Expected: 0 Return: 0 **PASSED**

**Test 3: return value of ac\_points\_check**

Expected: Two messages that Player1 lost points and that player DoubleA received more points than it was possible

Return:

Player Player1 lost points this turn!  
Player DoubleA achieved this turn more points than it is possible!

**PASSED**

**Test 4: ac\_arrow\_check**

Expected: No message because on board appeared only two arrows

Return: -No message-

**PASSED**

**Test 5: return value of ac\_pawn\_positions(Player1)**

Expected: 1 – Only one pawn of Player1 on the board

Return: 1

**PASSED**

**Test 6: return value of ac\_horizontal\_move(Player1)**

Expected: 1 – Pawn of Player1 moved horizontally

Return: 1

**PASSED**

**Test 7: return value of ac\_vertical\_move(DoubleA)**

Expected: 0 – pawn made impossible move

Return: 0

**PASSED**

**Test 8: return value of ac\_diagonal\_move(DoubleA)**

Expected: 0 – pawn made impossible move

Return: 0

**PASSED**

**Test 9: ac\_check\_move (Player1 & DoubleA)**

Expected: No message for Player1 and Warning for player DoubleA

Return:

Player DoubleA made impossible move with his amazon!

**PASSED**

**Test 10: cmp\_coordinates (x1, x2) and (x3, x4)**

Input: x1= (3,4), x2= (3,4), x3= (5,8), x4= (8,5)

Expected: 1 – the same coordinates and 0 – different coordinates

Return: 1, 0

**PASSED**