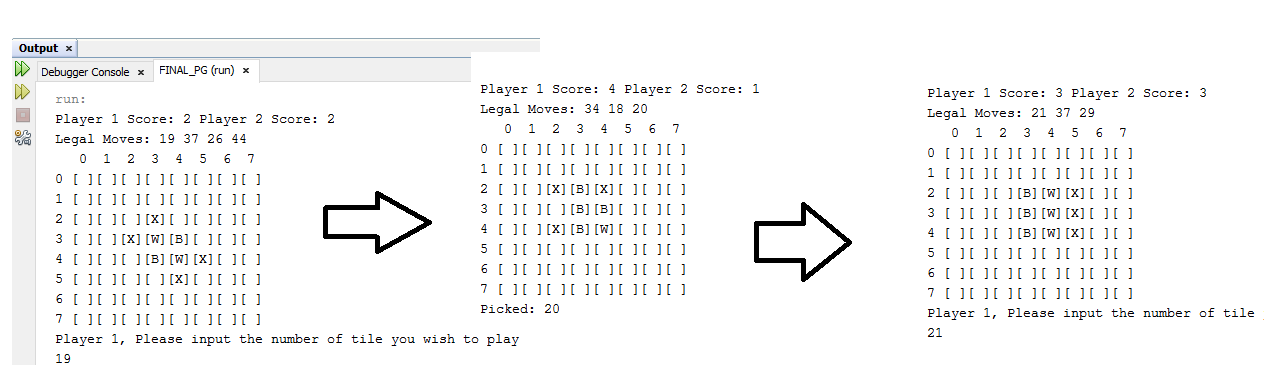
**Final\_PG:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Test | Reason | Output |
| 1. | Input a piece in [2,3] (the program recognizes it as 19) | Ensure that the possible moves are feasible | Possible moves are updated and marked with [X]s |
| 2. | Input a move which isn’t legal (not marked by x) | Make sure that the game’s flow cannot be violated | Illegal move is flagged and user is asked to re-input |
| 3. | Input a corner piece | Strategizing against the minMax ‘computer opponent’ | The computer comes up with the best possible move (another corner piece) |
| 4. | Input a piece in [5,0] for follow up | Test how the minMax algorithm will follow up to further moves | It equalizes the score, showing that it is attempting to find optimal strategies |
| 5. | Use different starting strategies | Ensuring that minMax responses are not random. | The algorithm picks an adequate comeback to yet again equalize the scores |

**Test Case 1:**



**Test Case 5:**

