Due dates D1: Thursday Feb 14 + D2: Thursday March 7 + D3: Thursday March 28, 2019
Late Submissions 20% per day per late deliverable
Teams Students registered in COMP 472 can do the project in teams of at most 4.
Students registered in COMP 6721 can do the project in teams of at most 2.
Teams must submit only 1 copy of the project via the team leader.
Purpose In this project, you will implement an adversarial search algorithm to play the game **Double Card**.

1 The Game

1.1 Initial Set-Up

Double Card is 2-player game played with 24 identical cards and an 8×12 board. As shown in Figure 1, each card has 2 segments (a left and a right segment) and has 2 faces (side 1 and side 2).

Figure 1: Cards. The game is played with 24 identical cards such as this one.

Each card can be placed on the board on either side and can be rotated by 0^o, 90^o, 180^o or 270^o, for a total of 8 possible placements. This is shown in Figure 2.

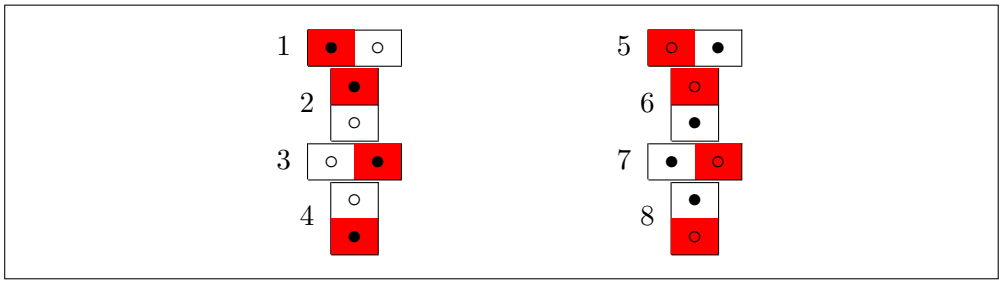


Figure 2: Card rotations. Each card can be rotated 8 ways before being placed on the board.

Initially, each player is given 12 cards. The game is played on a board of 8 (width) × 12 (height), that is initially empty. The board is numbered 1 to 12 from bottom to top, and A to H from left to right as shown in Figure 3.

top	12								
	11								
	...	...	...	...	...	...	...	...	...
	4								
	3								
	2								
	1								
bottom		A	B	C	D	E	F	G	H

Figure 3: Initial empty board. The board size is 8×12 and cells are numbered A to H and 1 to 12.

Before the game begins, each player decides what he/she will play: dots or color. The player who plays dots will try to get 4 consecutive •'s or 4 consecutive ○'s (in a row, a column or a diagonal) regardless of the colors of the cards. The player who plays colors will try to get 4 consecutive red[1] segments or 4 consecutive white segments (in a row, a column or a diagonal) regardless of the dots on the cards.

[1]If you print this document in black and white, the red segments will appear gray.

1.2 Moves and Winning

1.2.1 Regular Moves

Each players take turns placing his/her cards on the board. Cards can be rotated and placed on either side (see Figure 2), but can only be placed at row 1 of the board or on top of a card already on the board. A card cannot be placed on the board if one (or both) of its segments hangs over an empty cell. For example, Figure 4 shows a legal configuration; while Figure 5 shows an illegal configuration where the ∘ segment at `C2` is hanging over an empty cell.

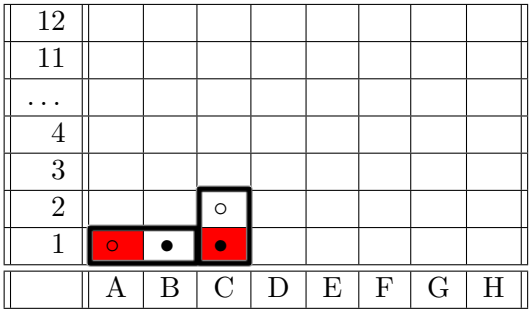


Figure 4: Example of a legal board

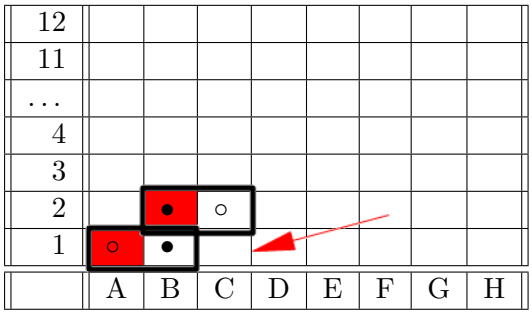


Figure 5: Example of an illegal board. The ∘ in `C2` is hanging over the empty cell in `C1`.

1.2.2 Winning

As soon as a player gets four consecutive identical dots or identical colors vertically, horizontally, or diagonally they win the game. If both players are in a winning position because the card that was just played puts both players in a winning position, then only the player who played the last card wins the game. An example of such a situation is shown in Figure 6.

Figure 6: Two possible winners. After moving the card ∘ ● at cells `A3 B3`, we have 4 ● in a row (at cells `B3` to `E3`) so the player playing the dots can win. However, we also have 4 red segments in a row (also at cells `B3` to `E3`), so the player playing the color can also win. In this case, who ever played the last card (the one at cells `A3 B3`) wins the game.

1.2.3 Recycling Moves

If all 24 cards have been played and neither player has won the game, then players proceed to using *recycling moves* rather than the regular moves explained in Section 1.2.1. A recycling move consists in taking one card off the board and placing it somewhere else on the board. A player cannot move the card that the other player just moved/placed and must ensure that the resulting board will be legal. Players keep recycling one card at a time until either one player wins the game or a total of 60 moves have been played. If after 60 moves (regular + recycling), no player has won, then the game ends in a draw.

2 Your Task

In this project, you will implement an adversarial search to play the game of Double Card automatically.
Students in COMP 472 will implement a minimax algorithm and 1 heuristic.
Students in COMP 6721 will implement a minimax algorithm + alpha-beta pruning and 2 heuristics.

2.1 Play Modes

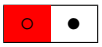
Your program should be able to run in manual mode and in automatic mode. This means that you should be able to run your program with:

1. manual entry for both players (i.e. 2 humans playing against each other)
2. manual entry for one player, and automatic moves for the other (i.e. one human playing against your AI)

After each move, your program must display the new configuration of the board.

2.2 Programming Details

1. To program the game, you must use Python.
2. Your program must be able to run on the lab machines and must decide on the next move to take in at most 3 seconds (real time in the lab).
3. It is not necessary to have a fancy user-interface. A simple command-line interface with a text-based output is sufficient.
4. In order to facilitate the running of the tournament (see Section 3.3), we will use the following standards:
   (a) The player (human or AI) will indicate its move by entering the coordinate of the card to play followed by the coordinate or where that card should go on the board.
      i. The card to play will be indicated by:
         A. For regular moves: a 0 followed by a digit from 1 to 8 (that indicates the rotation of the card using the numbering system of Figure 2).
         B. For recycling moves: the coordinates of the card on the board, followed by a digit from 1 to 8 (that indicates the rotation of the card using the numbering system of Figure 2).
      ii. The position of where the card will be placed on the board will be indicated by its bottom, left coordinates on the board.

   For example, the input `0 5 A 2` will indicate that we have a regular move and that card  will be placed at positions `A2` and `B2` on the board.
   The input `F 2 F 3 3 A 2` will indicate that we have a recycling move and the card at position `F2 F3` will be rotated as  will be placed at positions `A2` and `B2` on the board.
   (b) If a human player enters an illegal move (e.g. `0 5 H 1`), then the player will only be warned and be given a chance to enter another move with no penalty[2]. However, if your program generates an illegal move, then it will automatically lose the game.

[2]The point of this rule is to avoid losing a game in the tournament because a human did not transcribe your AI's move correctly.

3 Deliverables

The project will be divided into 3 deliverables and one tournament.

3.1 Evaluation Scheme

Students will be given individual grades that will be a function of the team grade and the peer-evaluation.

At the end of the project, all team members will fill-in a peer-evaluation form to evaluate the contribution of each team member. The grade of a student will be a function of his/her team grade and the peer evaluation received.

3.1.1 Team grades for COMP 472

Deliverable	**Functionality**	**%**	**Evaluation Criteria**
Deliverable 1 + Demo 1	manual game (no minimax, no heuristic)	20%	Manual entry, functionality of the rules of the game (e.g. visual trace after each move, detection of illegal moves, ...), programming quality, demo ...
Deliverable 2 + Demo 2	same as above + minimax with a simple heuristic	20%	Automatic mode (e.g. functionality of the minimax, use of a naive heuristic, generation of a legal move ...), programming quality, demo ...
Tournament *(nothing to submit)*	same as above + informed heuristic	15%	Participation & Results at the tournament
Deliverable 3	same as above + Report	45%	Clarity of the report, justification of the heuristic, depth of the analysis, presentation, grammar, ...

3.1.2 Team grades for COMP 6721

Deliverable	**Functionality**	**%**	
Deliverable 1 + Demo 1	manual game (no minimax, no heuristic)	10%	Manual entry, Functionality of the rules of the game (e.g. visual trace after each move, detection of illegal moves, ...), programming quality, demo ...
Deliverable 2 + Demo 2	same as above + minimax + alpha-beta pruning with a simple heuristic	20%	Automatic mode (e.g. functionality of the minimax, use of a naive heuristic, generation of a legal move ...), programming quality, demo ...
Tournament *(nothing to submit)*	same as above + 2 informed heuristics	15%	Participation & Results at the tournament
Deliverable 3	same as above + Report	55%	Clarity of the report, justification and comparison of the heuristics, analysis of alpha-beta, depth of the analysis, presentation, grammar, ...

3.2 Demos

Deliverables 1 and 2 will be demonstrated on the lab machines. You will not be able to demo on your laptop. Regardless of the demo time, you will demo the program that was uploaded as the official submission on or before the due date. The schedule of the demos will be posted on Moodle. No special preparation is necessary for the demo (no slides or prepared speech). Your TA will ask you questions on your code, and you will have to answer him/her. Part of your marks will be based on your demos.

3.3 Tournament

Part of the evaluation will be made via an in-class tournament; where pairs of projects will play against each other. COMP 472 teams will play against COMP 472 teams, and COMP 6721 teams will play against COMP 6721 teams. If, at any time, your AI takes more than 3 seconds (real time) to decide its next move, it will automatically be eliminated from the game, and your opponent will win. Part of your marks will be based on your participation and results at the tournament. More details about the tournament will be provided later in the semester.

3.4 Report

Note that this is a project, not an assignment. The difference is that it is open-ended and in addition to the required work stated above, you are expected to be creative, perform additional experiments, comparisons and analysis. This means that in addition to stating *what* you did, you should also describe *why* you did it that way and what other options were available and why they were not retained.

Your report should have a reference section (not included in the page count) that properly cites all relevant resources that you have consulted (books, Web sites ...), even if it was just to inspire you. Failure to properly cite your references constitutes plagiarism and will be reported.

3.4.1 Report for COMP 472

In COMP 472, your report should be 4-5 pages (without references and appendix) and use the template provided on Moodle. The report should contain at least the following:

- $\frac{1}{2}$ *to 1 page:* Introduction and technical details.
- *1 to* $1\frac{1}{2}$ *page:* A description and justification of your heuristic. In particular, describe how you came up with it, what features of the game state it considers and why you chose them; how you balanced speed of the evaluation function with performance, and generally why you settled on the evaluation function you did.
- *1 page:* A description and analysis of your results either at the tournament or when you played against it (why your heuristic or search makes good or bad decisions).
- $\frac{1}{2}$ *to 1 pages:* A description of any difficulties that you have encountered and how you addressed them.
- $\frac{1}{2}$ *page:* In the case of team work, a description of the responsibilities and contributions of each team member.
- not included in the page count: References, Appendix, Figures, Tables as needed.

The report must:

☐ follow the Word or LaTeX template provided on Moodle or an equivalent format.
☐ be submitted in PDF format
☐ be called `472_P1_Report_StudentID1_StudentID2_...pdf`

3.4.2 Report for COMP 6721

In COMP 6721, your report should be 6-7 pages. In addition to following the same criteria as in COMP 472 (see Section 3.4.1), your report must:

- *1 page:* Compare and contrast the strengths and weaknesses of your 2 heuristics.
- *1 page:* Analyze the effectiveness of alpha-beta pruning.

The report must be in PDF format and must be called `6721_P1_Report_StudentID1_StudentID2.pdf`

4 Submission

4.1 Deadlines

Deliverable	Due Date	Submit
Deliverable 1	Thursday Feb 14, 2019	on EAS as project 1
Deliverable 2	Thursday March 7, 2019	on EAS as project 2
Tournament	Monday March 18 & Thursday March 21	nothing to submit
Deliverable 3	Thursday March 28, 2019	on EAS as project 3 + paper submission of the report

Deliverables 1, 2 and 3 are due at midnight on the due date.

4.2 What to Submit

4.2.1 Deliverables 1 and 2

- ☐ Read and sign the expectation of originality form (available on Moodle; or at: `http://www.encs.concordia.ca/documents/expectations.pdf`) and submit it in your package.
- ☐ Create a `README.txt` which will contain specific and complete instructions on how to run your program on the desktops in the computer labs. If the instructions in your readme file do not work or are incomplete, you will not be given the benefit of the doubt.
- ☐ Create one zip file, containing all your code, the `README.txt` file and the expectation of originality.
- ☐ Name your zip file: `472_P1_Dn_studentID1_studentID2_....zip` or `6721_P1_Dn_studentID1_studentID2_....zip` where n=1 or 2 and studentID1 is the team leader. For example, `6721_P1_D1_12345678.zip` or `472_P1D2_12345678_87654321.zip`
- ☐ Have the Team Leader upload the zip file from his/her account at: `https://fis.encs.concordia.ca/eas/` as project1 or project2.

4.2.2 Deliverable 3

Deliverable 3 is your final code and the report.

- ☐ Read and sign the expectation of originality form (available on Moodle; or at: `http://www.encs.concordia.ca/documents/expectations.pdf`) and submit it in your package.
- ☐ Make sure that your report follows the format indicated in Section 3.4.
- ☐ Create a `README.txt` which will contain specific and complete instructions how to run your program on the desktops in the computer labs. If the instructions in your readme file do not work or are incomplete, you will not be given the benefit of the doubt.
- ☐ Create one zip file, containing all your code, the `README.txt` file, the report and the expectation of originality.
- ☐ Name your zip file: `472_P1_D3_studentID1_studentID2_....zip` or `6721_P1_D3_studentID1_studentID2_....zip` where studentID1 is the team leader.
- ☐ Have the Team Leader upload the zip file from his/her account at: `https://fis.encs.concordia.ca/eas/` as project3.
- ☐ In addition, print your report and submit a paper copy in the appropriate assignment box in EV 3.177.

Have fun!