```
FUNCTION main
	DEFINE difficulty
	READ user input
	STORE difficulty = user input
	DEFINE struct deck[4][13]
	DEFINE struct selection[difficulty][difficulty]
	DEFINE string name
	WHILE name != string
		PRINT “Enter Username”
		READ user input
		STORE name = user input
	END WHILE

	PRINT “How to play message”

	CALL createDeck
	CALL createSelection

	DEFINE SET status = false
	WHILE check = false
		CALL printSelection
		CLICK CARD user input
		STORE row and column
		CALL swap with selection[row][column],selection[row][column]
		SET status = CALL check RETURNING status
	END WHILE

	PRINT “Winning message”
END FUNCTION
////////////////////////////////////////////////////////////
FUNCTION createDeck
	DECLARE i = 0,ii = 0
	FOR(i = 0; i < 4; i++)
		FOR(ii = 0; ii < 13; ii++)
			deck[i][ii].suit = i;
			deck[i][ii].rank = ii;
		END FOR
	END FOR
END FUNCTION
////////////////////////////////////////////////////////////
```

```
FUNCTION shuffle
        FOR(int i = 0; i < 4; i++)
                DEFINE int r = rand() % 4
                FOR(int j = 0; j < 13; j++)
                        DEFINE SET rr = rand() % 13
                        DEFINE SET struct temp = deck[i][j]
                        SET deck[i][j] =  deck[r][rr]
                        SET deck[r][rr] = temp
                END FOR
        END FOR
END FUNCTION
//////////////////////////////////////////////////////////////////////
FUNCTION createSelection
        DEFINE int j = 0,jj = 0,i = 0,ii = 0
        CALL shuffle

        FOR(i = 0; i < difficulty; i++)
                FOR(ii = 0; ii < difficulty; ii++)
                IF jj > 12
                        SET jj = 0
                        SET j = j + 1
                END IF
                SET selection[i][ii] = deck[j][jj]
                SET jj = jj + 1
                END FOR
        END FOR
END FUNCTION
//////////////////////////////////////////////////////////////////////////////
FUNCTION printSelection
        FOR(int row = 0; row < selectionNum; row++)
                FOR(int col = 0; col < selectionNum; col++)
                        PRINT "selection[row][col].suit,selection[row][col].rank”
                END FOR
        END FOR
END FUNCTION
//////////////////////////////////////////////////////////////////////////////
FUNCTION swap struct *a,struct *b
        DEFINE struct t
        SET t = *a;
        SET *a = *b;
```

```
        SET *b = t;
END FUNCTION
////////////////////////////////////////////////////////////////////////////////
FUNCTION check
        DEFINE status = true
        DEFINE int i = 0
        DEFINE row =  < Integer >
        DEFINE col = < Integer >
        FOR(i = 0; i < difficulty && status; i++)
                FOR(int j = 0; j < c && status; j++)
                        IF selection[j][i] > difficulty || selection[j][i] < 1 || selection[i][j] >
difficulty  || selection[i][j] < 1
                                SET status = false
                        ELSE
                                SET row add to selection[j][i]
                                SET col add to selection[i][j]
                        END IF
                END FOR
                IF status && row.size() != difficulty  || col.size() != difficulty
                        SET status = false
                END IF
        SET clear row
        SET clear col
        END FOR
        RETURN status
END FUNCTION
```