DOMINO GAME APPLICATION DEVELOPMENT

Iwan Rijayana

DOI: https://doi.org/10.37178/ca-c.23.1.081

Iwan Rijayana*, Faculty of Engineering, Widyatama University, Bandung, Indonesia*
Email: iwan.rijayana@widyatama.ac.id

Abstract

Domino is a game that is very well known by the world community. The basic algorithm for creating a domino game application is described in this research. The system will provide a graphical user interface, where the user will perform all the tasks. The result of this research is the creation of an interesting domino game application for users to play.

Keywords: Algotithm, Application, Domino, Game, Play

I. INTRODUCTION

Software games that are currently developing provide many online facilities, which allow users to play with other users in a network, be it a local network or the internet. These online facilities can not only be played by 1 or 2 people, but can reach tens or even hundreds of people, depending on the system and type of game.

In everyday life, many card games are found. One of the most popular card games is dominoes. This game has 28 cards with each card having 2 values. The two values of the card are a combination of a pair of values starting from zero (empty) to six. The rules of the game are quite simple, each player is only tasked with arranging cards by starting from the same largest card owned by the player. If the player cannot walk (does not have a card to remove) then the player must take a card from the rest of the card pile until he gets a card that can be issued (if there are still cards remaining) or continue to the next player (if there are no remaining cards). The first player to spend his cards wins the game. By knowing the rules of the game from dominoes, a game software will be created that can be played by 2 people who are on a network, either local or internet.

This research belongs to the type of research and development (R&D) which is a process or steps to develop a new product or improvement of existing products [1].

II. LITERATURE REVIEW

2.1. Domino

2.1.1. History Domino

The square card game was invented in China in 1120. Several historians' calculations have proven the existence of this game, starting with a heroic warrior named Hung Ming (181 – 234). Other historians believe that Keung T'ai Kung in the 12th century BC has created the game.

The Department of Investigation on the Tradition of All Things ("Chu sz yam") states that the domino game was invented by an official in 1120[2]. It is said that the person had gifted it to King Hui Tsung, and that it had been spread abroad during the reign of Hui's son, Kao. -Tsung (1127-1163). Other experts state that this document refers to a standardization and not the creation of the game.

Michael Dummett writes briefly in the history section of his “Game of Tarot” book, that the introduction of dominoes in Europe to Italy, possibly Venice and Naples in the 18th century. Although the game of dominoes is clearly a descendant of the Chinese, there is a debate over whether European domino games came from China to Europe in the 14th century or were created separately.

A single domino was found along with the wreckage of Mary Rose in the early 16th century. In general, there is ample evidence for games in the 16th and 17th centuries and if domino games did exist then they certainly would not have gone unnoticed.

The game moved from Italy to France in the early 18th century and became a fad. In the late 18th century, France also produced domino puzzle games. This puzzle consists of two types. The first type, is given an arrangement (pattern), and the goal is to place the cards until they all match. The second type, is given an arrangement (pattern), and the goal is to place the cards according to the arithmetic properties of the seeds.

The game of dominoes arrived in England in the late 18th century from France, probably brought over by French convicts of war and quickly became popular in inns and motels at that time.

European dominoes are rectangles whose side lengths are twice the width of the sides. A simple deck of cards consists of cards that have a combination of pairs of dice ranging from one to six, plus an empty square, so that a total of 28 cards are owned. Another pile of cards with a higher number is found later, with a combination of circle pairs up to nine pieces and twelve pieces.

The word "domino" seems to be derived from the Latin "dominus" which means ruler of the house. But the word "domino" is in French which means a black and white head scarf worn by Christian priests in winter. It is possible that this word is the origin of the name of the domino game [3].

2.1.2. Rule of The Game

This domino game is played by people all over the world, and this game is very popular in Latin American countries. This domino game has 28 cards with each card having 2 values. The two values of the card are a combination of a pair of values starting from zero (empty) to six.

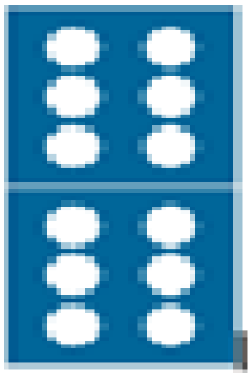


Figure 1:Example of a domino

The rules of the game are also quite simple. Each player is only tasked with arranging cards by starting from the same largest card owned by the player. If the player cannot walk (does not have a card to remove) then the player must take a card from the rest of the card pile until he gets a card that can be issued (if there are still cards remaining) or continue to the next player (if there are no remaining cards). The first player to spend his cards wins the game [4-6].

2.2. Image Format

2.2.1. BMP Image Format

The BMP file format is the standard image file format for computers running the Windows operating system (Baker, et all, 2013). This file format was developed by Microsoft to store image files (bitmaps) and allow the Windows operating system to display these images again. The structure of a BMP file consists of a Bitmapfileheader and a Bitmapinfoheader. The Bitmapfileheader structure stores information about the dimensions and color format of the device independent bitmap (DIV).

So it can be concluded that the Bitmapfileheader provides information about the file and the Bitmapinfoheader provides information about the image. Color table defined as RGBQUAD array and structure and the rest is image data. This format supports resolution and monochrome to true color (16.7 million colors).

2.2.2. GIF Image Format

GIF (Graphics Interchange Format) is a standard image format that provides a number of capabilities such as sharp image resolution and a relatively smaller image file size. Whether or not the resolution of the resulting GIF image is good also depends on the hardware (Graphics Hardware) used. The hardware must be able to produce sharp color resolution images with good color pixels (Almeer, 2013).

Pixel is the smallest element in an image. Pixels correspond to dots (dots) in the graphic screen. For example, an image has a size of 320 x 320. This means that the image has a number of pixels of 320 x 320. The more pixels contained in the image object, the smoother the image because the distance between pixels is getting closer and the file size is bigger. the. And vice versa, where the fewer the number of pixels contained in the image object, the appearance of the image looks rough because the distance between pixels is tenuous and the resulting file size is smaller.

The GIF format is intended to support current and future image formats. The GIF format has a .gif extension that is supported by graphic applications, such as Microsoft Paint, Corel Draw, Corel PhotoPaint, Adobe PageMaker and so on.

2.3. Multimedia

A software that uses more than one method of communicating with its users, such as: text, images, sound, animation, and video, or a combination of text, images, sound, animation, and video on a computer.

The emergence of multimedia is triggered by the human desire to make computers and their applications more attractive so that users feel at home and it is easier to digest information from computers. With the multimedia facilities, users can easily interact with computers [7].

2.3.1. Multimedia Elements

The following is an explanation of each multimedia element:

1. **Text**

Text is the basic element of multimedia in conveying information. Text is the simplest type of element and takes up the least amount of storage space.

2. **Pictures**

Using pictures is the easiest way to illustrate information. To explain a horse to a user, for example, it would be easier to use a picture than to describe it in words about the animal.

3. **Voice**

With the sound, we can give a concrete example of pronunciation. This is better than giving examples of pronunciation with words that are almost the same or similar to the sound in question. For example, the pronunciation of the vowel 'a', it is better to give direct examples of real sounds, although explanations with text such as the pronunciation of 'a' in 'chicken' are also very helpful.

4. **Animation**

Animation is a static image that is manipulated to produce the effect of movement. Animation is used to convey information that is felt to be quite complex which cannot be fully explained with text or images.

5. **Videos**

Video is an animation that is taken through a video camera and saved in the form of a file.

2.3. Multimedia Uses

Multimedia provides many ways so that ordinary people can get good computer usage information. The use of multimedia can be done in businesses, educational centers, public places, even at home. The following is an explanation of each place where multimedia is used:

1. ***Business***

Multimedia applications in business are used in presentations, training, marketing, product demonstrations, catalogs, and network communications. In presentations, for example, multimedia can make listeners more excited. Many presentations combine the use of sound and video. This is certainly better when compared to a presentation that only uses slides and an overhead projector which is quite boring.

2. ***Education center***

Many schools and universities have taken full advantage of multimedia technology in their environment. For example, internet facilities to search for supporting materials for lessons, recording lecturers' voices while teaching so they can repeat lessons at home, sending assignment results via e-mail directly to the teacher's mailbox.

3. ***Public places***

Uses that are placed in hotels, stations, shopping centers, museums, shops, or in other public places are usually in the form of a terminal or kiosk that provides information or assistance. At a hotel information kiosk, for example, there is information about the restaurants in the hotel, a map of the city, recreational areas, flight schedules, and various other information.

4. ***Home***

Currently, the use of multimedia at home with a computer is only limited to completing schoolwork and playing, but in the future, multimedia will reach homes through televisions that are combined with input devices.

III. ANALYSIS SYSTEM

3.1 Connection Analysis Between Two Players

As a liaison or mediator between two computers, the author uses 'Microsoft Winsock 6.0', a visual basic component that can send and receive data from other winsocks that are connected to each other. Each software has one winsock component, winsock which functions as a server will wait for connection requests (listening) until winsock which functions as a client connects itself and winsock between two players is connected. After winsock is connected, the game can be started and winsock can exchange information with each other to regulate the game according to the applicable regulations.

The algorithm for waiting for connection requests (listening) on the winsock server (with local port = “1111”) is as follows,

```
 WS.Close
WS.LocalPort = "1111"
WS.Listen
```

While the algorithm to connect the winsock client with the winsock server (with server port = "1111") is as follows,

```
WS.Connect txtIP, "1111"
```

3.2. Analysis of Domino Card Random

Domino card shuffling is done the first time the domino game starts. The card shuffling process is run on the server program, while the client program just waits for the randomization results from the server program.

The card shuffling process is done by generating a random number. The random number that is generated is the position of the nth domino in the position being filled. The contents of the card are then stored into an array. If you get a card with contents that have been obtained before, then repeat the shuffling. The randomization loop is repeated until the card array is completely filled.

After the card shuffling is complete, 7 (seven) cards are drawn from the back card position for the server player, and the next 7 (seven) cards for the client player. The server program will send the card contents of each player to the client program.

The algorithm for shuffling the dominoes and sending the randomization results to the client program is as follows:

```
nK = 0
While nK < 28
nK = nK + 1
{Generate random number}
Randomize
J = Round(Rnd * 27) + 1
{If the card has been previously obtained}
While InArray(TCard(J),PCard)
Randomize
J = Round(Rnd * 27) + 1
Wend
{Save to array}
ReDim Preserve CardP(nK)
PCard(nK) =TCard(J)
Wend {Take your own card – 7 cards from the back of the card }
ReDim Card1(7)
cCard1 = ""
For nK = 1 To 7
```

```
If cCard1 <> "" Then cCard1 = cCard1 & ","
Card1(nK) = CardP(29 - nK)
cCard1 = cCard1 & Card1(nK)
Next nK
{Take the opponent's card – the next 7 cards from the back of the card}
ReDim Card2(7)
cCard2 = ""
For nK = 1 To 7
If cCard2 <> "" Then cCard2 = cCard2 & ","
Card2(nK) = CardP(22 - nK)
cCard2 = cCard2 & Card2(nK)
Next nK
{Remaining cards – 14 cards placed in the middle}
ReDim Preserve CardP(14)
cCard = ""
For nK = 1 To 14
If cCard <> "" Then cCard = cCard & ","
cCard = cPCard & PCard(nK)
Next nK
{Send the contents of player-1 card to the client program}
frmStart.WS.SendData "CARD1~" & cCard1 & "$E"
DoEvents
{Send the contents of player card-2 to the client program}
frmStart.WS.SendData "CARD2~" & cCard2 & "$E"
DoEvents
{Submit the contents of the remaining cards – placed in the center of the game board}
frmStart.WS.SendData "CARDUP~" & cCardP & "$E"
DoEvents
{Send instructions to the client program to start the game}
frmStart.WS.SendData "START~$E"
DoEvents
{Fill card variable value}
bStart = True
YourCard = Card1
Opponent's Card = Card2
```

3.3. Player Turn Manager Analysis

In this software there is a function to check whether the player in question can run or not. If you can't walk, you will have two choices, whether to draw a card or miss your turn to the opposing player (pass).
The turn control algorithm is as follows,

```
 {Change turn variable}
Turn = IIf(Turn = 1, 2, 1)
{Refresh the display of player-1 and player-2 card counts}
Call RefreshAmount
{If both players cannot walk – then draw}
If (Can Walk(Your Card) Or Can Walk(Fight Card)) = False Then
   If UBound(PCard) = 0 Then
      Turn = 0
      lblDescription = "You and the Opposing Player Cannot Walk." & Chr(13) & _
                  "Game ends in a draw."
      MsgBox "You and Your Opponent Can't Walk." & Chr(13) & _
           "Game ends in draw!", vbCritical
```

```
        {Add to move - list}
        ListMove.AddItem ListMove.ListCount + 1 & "Game ends in draw."
        ListMove.ListIndex = ListMove.ListCount - 1
        ListMove.SetFocus
        Exit Sub
    End If
End If
```

{Checks the turn and whether the player can walk, if not then there are two choices, namely whether to draw a card or miss his turn on the opponent}

```
If Turn = nID Then
    ImgSelect.DragMode = vbAutomatic
    'Check can run or not
    lblDescription = "Your turn!"
    Call CekCanJalan(Your Card)
Else
    cmdDrag.Visible = False
    cmdPass.Visible = False
    ImgSelect.DragMode = vbManual
    lblDescription = "It's Opposing Player's Turn!"
End If
```

3.4. Issued Domino Check Analysis

As is well known, the domino game has game rules. One of them is that the card issued must have the same number (seed) as the number on the end of the card that was previously issued.
To be clear, consider the following picture where the domino card position can only be filled by cards that have 4 seeds on the left and 3 seeds on the right.

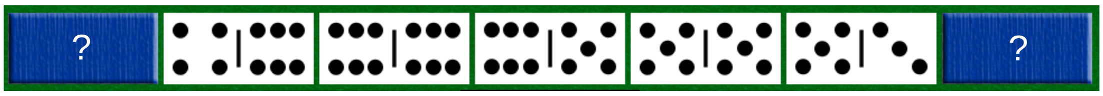


Figure 2 :
The position of domino cards that can only be filled by cards that are has 4 seeds (on the left) and 3 (on the right)

Each card issued is stored in 2 (two) variables, namely one variable as the size of the seeds from the left end of the left card and one variable as the size of the seeds from the right end of the right card. The checking process will compare whether the size of the seeds from the end of the card issued is the same as the size of the seeds on the variable according to the placement area. If they are the same, then the checking process is true and dominoes can be placed. If they are not the same, then the checking process is false and the domino card cannot be placed.

If the player who in turn does not have a domino that can be issued, then the player can draw one card from the rest of the card pile or can also pass his turn to another player (pass).

The algorithm for checking dominoes that can be issued is as follows,

```
bValid = False
If ImgDrop(pnIndex).Tag <> "" Then Exit Function
If bInitial = False Then
    Dim A() As String
    Dim B() As String
    If pnIndex < 6 Then
        {Check right card}
        C = pnIndex + 1
        If ImgDrop(C).Tag <> "" Then
            A = Split(ImgDrop(C).Tags, "-")
            B = Split(pcSourceTag, "-")
            If B(1) = A(0) Then
                bValid = True
            End If
        End If
    End If
    If pnIndex > 1 Then
        {Check left card}
        C = pnIndex - 1
        If ImgDrop(C).Tag <> "" Then
            A = Split(ImgDrop(C).Tags, "-")
            B = Split(pcSourceTag, "-")
            {Card can run}
            If B(0) = A(1) Then
                bValid = True
            End If
        End If
    End If
Else
    'Valid
    bValid = True
End If
ValidMove = bValid
```

3.5. Analysis of the Placement and Arrangement of Domino Cards on the Game Board

Due to the large limitations of the monitor, the placement of domino cards in this application is done horizontally and uses a history system. When the card placed on the game board has exceeded the monitor limit, the use of the history system will be activated.

The algorithm for laying and arranging dominoes is as follows,

```
    {Load image drop}
    ImgDrop(pnIndex).Tag = pcSourceTag
    ImgDrop(pnIndex).Picture = LoadPicture(App.Path & "\Image\S-" &
pcSourceTag & ".gif")
{Set card image into history}
If pnIndex = 0 Or pnIndex = 6 Then
    {Previously there is history}
    If ImgDrop(3).Tag = "H" Then
       {Leftmost index}
        If pnIndex = 0 Then
            {Save to history}
            C = UBound(HCard) + 1
            ReDim Preserve CardH(C)
            {Slide Card Back}
            For C = UBound(HCard) To 2 Step -1
                HCard(C) = HCard(C - 1)
            Next C
            {Fill Value}
            HCard(1) = ImgDrop(2).Tag
            {Swipe Right}
            For C = 2 To 1 Step -1
               ImgDrop(C).Tag = ImgDrop(C - 1).Tag
               Call LoadGbrDrop(ImgDrop(C))
            Next C
            {Empty zero index}
            ImgDrop(0).Tags = ""
            Call LoadGbrDrop(ImgDrop(0))
         End If
         {rightmost index}
         If pnIndex = 6 Then
{Save to history}
            C = UBound(HCard) + 1
            ReDim Preserve CardH(C)
            CardH(C) = ImgDrop(4).Tag
{Swipe left}
            For C = 4 To 5
               ImgDrop(C).Tag = ImgDrop(C + 1).Tag
               Call LoadGbrDrop(ImgDrop(C))
            Next C
{Empty zero index}
            ImgDrop(6).Tag = ""
            Call LoadGbrDrop(ImgDrop(6))
         End If
      Else
          {No previous history}
          {Leftmost index}
         If pnIndex = 0 Then
            If ImgDrop(5).Tag = "" Then
{Swipe right}
               For C = 5 To 1 Step -1
                  ImgDrop(C).Tag = ImgDrop(C - 1).Tag
                  Call LoadGbrDrop(ImgDrop(C))
               Next C
            Else
               {Save to history}
               C = UBound(HCard) + 1
               ReDim Preserve CardH(C)
               CardH(C) = ImgDrop(2).Tag
               C = UBound(HCard) + 1
               ReDim Preserve CardH(C)
               CardH(C) = ImgDrop(3).Tag
               {Slide card to the right}
               For C = 2 To 1 Step -1
                  ImgDrop(C).Tag = ImgDrop(C - 1).Tag
                  Call LoadGbrDrop(ImgDrop(C))
               Next C
               {History Card}
               ImgDrop(3).Tag = "H"
               Call LoadGbrDrop(ImgDrop(3))
            End If
            {Empty Zero Index}
            ImgDrop(0).Tags = ""
            Call LoadGbrDrop(ImgDrop(0))
```

```
        End If
        {rightmost index}
        If pnIndex = 6 Then

            If ImgDrop(1).Tag = "" Then
    {Slide the card to the left}
For C = 1 To 5
                    ImgDrop(C).Tag = ImgDrop(C + 1).Tag
                    Call LoadGbrDrop(ImgDrop(C))
                Next C
            Else
    {Save to history}
                C = UBound(HCard) + 1
                ReDim Preserve CardH(C)
                CardH(C) = ImgDrop(3).Tag
                C = UBound(HCard) + 1
                ReDim Preserve CardH(C)
                CardH(C) = ImgDrop(4).Tag
   {Swipe right}
                For C = 4 To 5
                    ImgDrop(C).Tag = ImgDrop(C + 1).Tag
                    Call LoadGbrDrop(ImgDrop(C))
                Next C
   {History Card}
                ImgDrop(3).Tag = "H"
                Call LoadGbrDrop(ImgDrop(3))
            End If
{Empty right index}
            ImgDrop(6).Tag = ""
            Call LoadGbrDrop(ImgDrop(6))
        End If
    End If
End If
{Subtract card}
If Turn = nID Then
    Card2 = YourCard
Else
    Card2 = Opponent's Card
End If
ReDim Card1(0)
For C = 1 To UBound(Card2)
    If Card2(C) <> pcSourceTag And _
       Card2(C) <> Reverse(pcSourceTag) Then
       ReDim Preserve Card1(UBound(Card1) + 1)
       Card1(UBound(Card1)) = Card2(C)
    End If
Next C
{card image}
If Turn = nID Then
    YourCard = Card1
    Call ImageYourCard
Else
    Opponent's Card = Card1
End If
{Check whether it is game or not}
```

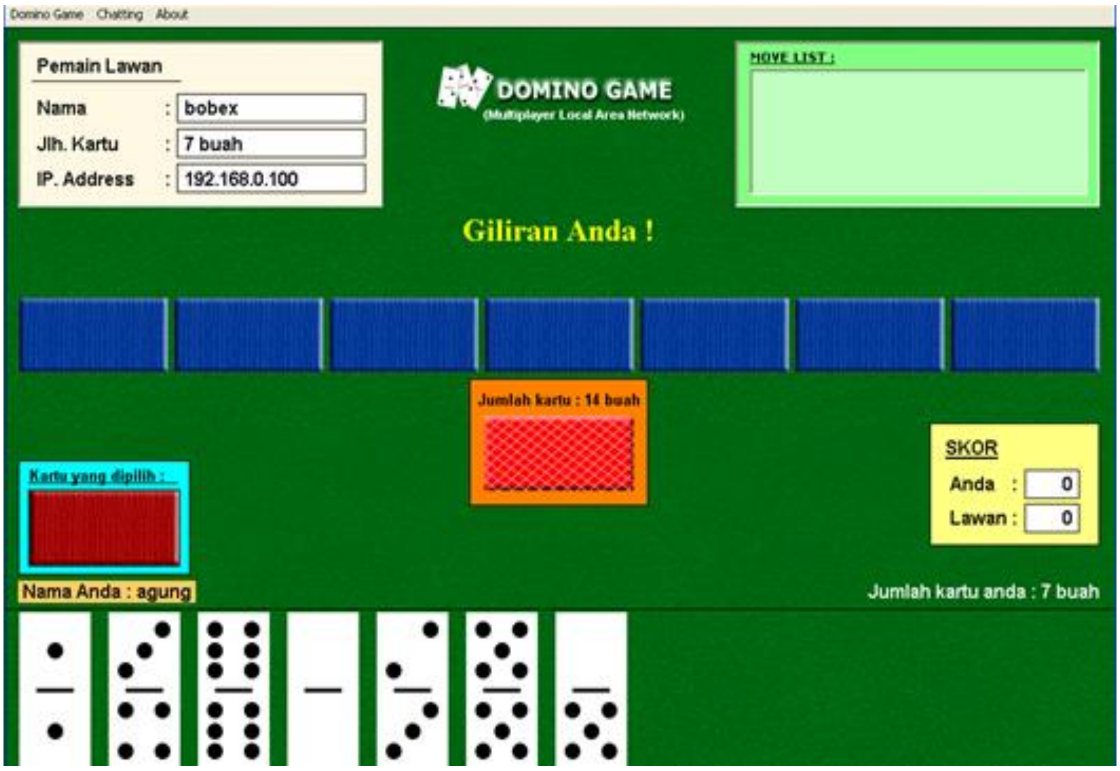
```
If UBound(Your Card) = 0 Then
   Voice Call("WIN")
   lblDescription = "You Win!"
   MsgBox "You Win!", vbInformation
   ListMove.AddItem ListMove.ListCount + 1 & ". " & _
                cName(Turn) & " WIN !"
   ListMove.ListIndex = ListMove.ListCount - 1
   ListMove.SetFocus
   ImgSelect.DragMode = vbManual
   ImgSelect.Tag = LoadPicture(App.Path & "\Image\BK-SM.gif")

   Turn = 0
   RefreshAmount
ElseIf UBound(Fight Card) = 0 Then
   Voice Call("LOSE")
   lblDescription = cName(Turn) & "Win !"
   MsgBox "You Lose!", vbInformation
   ListMove.AddItem ListMove.ListCount + 1 & ". " & _
                cName(Turn) & " WIN !"
   ListMove.ListIndex = ListMove.ListCount - 1
   ListMove.SetFocus
   Turn = 0
   RefreshAmount
Else
   {Change turn}
   Call ChangeTurn
End If
```

3.6. Winner Determination Analysis

The first player to spend his dominoes wins the game. If no player has cards that can be removed and there are no remaining piles of cards, the software will calculate the number of seeds from the remaining cards of the two players, the player with the smallest remaining number of seeds wins the game, whereas if the number of remaining seeds of the two players is the same, then the game is declared a draw.

IV. IMPLEMENTATION SYSTEM

The software will shuffle the cards, and then the game is played in turns. The domino cards that are issued will be arranged on the game board. The card in the middle will be saved as a history card.



Pemain Lawan

Nama	:	bobex
Jlh. Kartu	:	7 buah
IP. Address	:	192.168.0.100

Giliran Anda !

Jumlah kartu : 14 buah

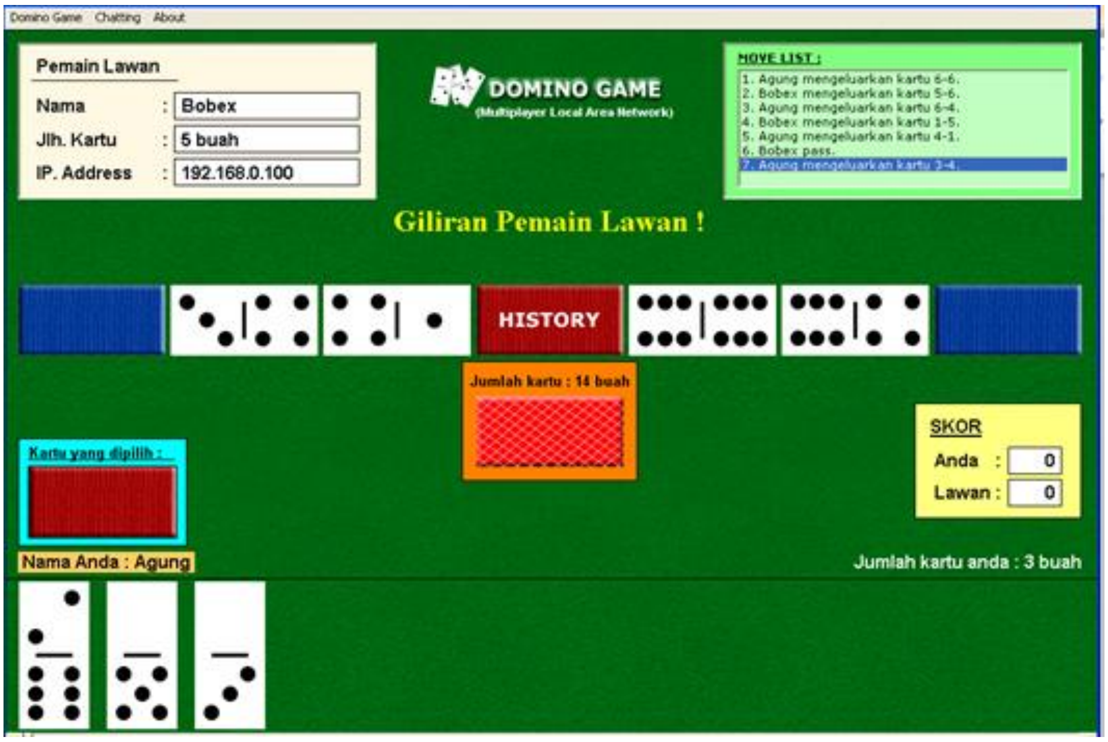
Kartu yang dipilih :

Anda	:	0
Lawan	:	0

Nama Anda : agung

Jumlah kartu anda : 7 buah

Figure 3 Game Form Display



Pemain Lawan

Nama	:	Bobex
Jlh. Kartu	:	5 buah
IP. Address	:	192.168.0.100

1. Agung mengeluarkan kartu 6-6.
2. Bobex mengeluarkan kartu 5-6.
3. Agung mengeluarkan kartu 6-4.
4. Bobex mengeluarkan kartu 1-5.
5. Agung mengeluarkan kartu 4-1.
6. Bobex pass.
7. Agung mengeluarkan kartu 3-4.

Giliran Pemain Lawan !

Jumlah kartu : 14 buah

Kartu yang dipilih :

Anda	:	0
Lawan	:	0

Nama Anda : Agung

Jumlah kartu anda : 3 buah

Figure 4: Display of the Game Form during the game

The history card in the middle can be displayed by clicking on the card.

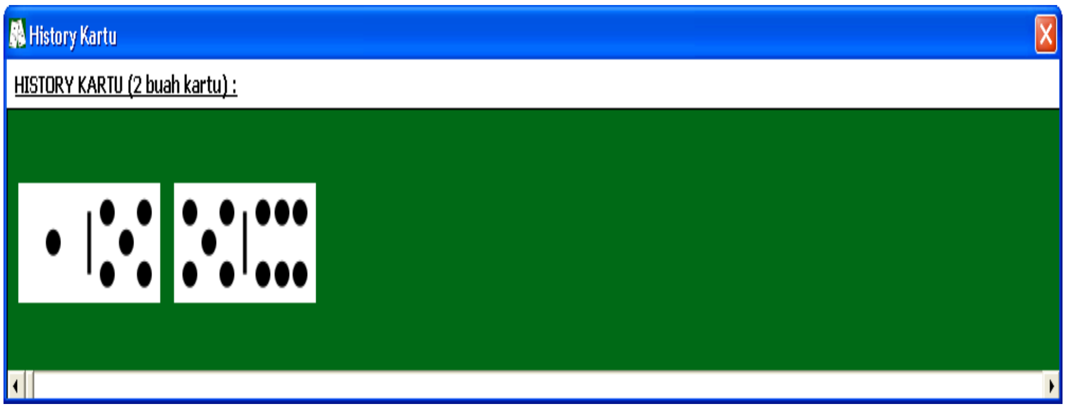


Figure 5 :Display Form History

If one of the players has finished his dominoes, then that player comes out as the winner.

V. CONCLUSIONS AND RECOMMENDATIONS

5.1. Conclusion

After completing the design of the domino card game software in multiplayer, the following conclusions can be drawn,

1. This program can be played in a computer network system with a computer as a server and another computer as a client.
2. This program can be used to play dominoes on a network without facing each other physically.
3. Software that has online features is considered attractive because it is dynamic.

5.2 Recommendation

Suggestions that can be submitted for further development include,

1. Software can be added for individual games (single player) with opposing players in the form of a computer (applying the concept of Artificial Intelligence (AI).
2. The game can be developed using 4 users.
3. The software can be developed with a wider game board and domino card layout that can be adjusted by the user, whether the cards will be placed horizontally or vertically.

REFERENCES

1. Gillies, R.M., *Cooperative learning: Developments in research.* International Journal of Educational Psychology, 2014. **3**(2): p. 125-140.
2. Wilkinson, W.H., *Chinese origin of playing cards.* American Anthropologist, 1895. **8**(1): p. 61-78.DOI: https://doi.org/10.1525/aa.1895.8.1.02a00070.
3. Davis, B.L. and D. Maclagan, *The card game SET.* The Mathematical Intelligencer, 2003. **25**(3): p. 33-40.DOI: https://doi.org/10.1007/BF02984846.
4. Isma, Y.E.N. and R. Hidayah, *Development of Domino Chemistry Game Card Media to Practice Analytical Thinking Skills of Students in Chemical Bonding Topic of Class X Semester 1.* Journal of Chemical Education, 2015: p. 386-392.
5. Okpube, N.M. and M.N. Anugwo, *Card Games and Algebra Tic Tacmatics on Achievement of Junior Secondary II Students in Algebraic Expressions.* International Journal of Evaluation and Research in Education, 2016. **5**(2): p. 93-100.DOI: https://doi.org/10.11591/ijere.v5i2.4527.

6. Vun, L., et al., *Educational DNA card game for the understanding of DNA and biotechnology.* Int J Educ Res, 2013. **1**: p. 1-6.
7. Bürger, T. and E. Simperl. *A conceptual model for publishing multimedia content on the semantic web, 101-113*. Springer.