```
'Memory card game using QCard32.dll
' copyright July 2011, Alyce Watson
' you may learn from this code
' you may borrow sections of code
' you may not redistribute this code -
'     do not post on a web page, message board, archive, etc.

gameWon=0   'flag that is set when all pairs are removed

[varSetup]
i=0         'i will be our counter var in for/next loops
design=6    'card back design
newIndex=0  'used when shuffling
tempCard=0  'temp var used when shuffling
clickCard=0 'index of current card clicked by user
dim card(24)'array to hold card info

gosub [fillCardArray]   'fill array with card values

nomainwin
    WindowWidth=640:WindowHeight=500
    UpperLeftX=1:UpperLeftY=1

    menu #1, "&File", "&New",[new],"&About",[about],"E&xit", [quit]
    menu #1, "&Card Back Design","&Circles",[circles],"&Blue",[blue],_
    "&Red",[red],"&Mountain",[mountain],"&Purple",[purple],
"M&usic",[music]
    graphicbox #1.g2,410,406,300,40
    graphicbox #1.g, 0, 0, 640, 480
    open "Memory Card Game" for window_nf as #1
    #1 "trapclose [quit]"
    #1.g2 "down;fill 230 230 150;backcolor 230 230 150;color brown"

    'get graphicbox handle
    hBox=hwnd(#1.g)

    'open the dll
    open "qcard32.dll" for dll as #qc
    'initialize the deck
    Call InitializeDeck hBox

[new] 'reset variables and shuffle cards for next try
    turns=0     : pairs=0
    clickCard=0 : gameWon=0
    cardOne=0   : cardTwo=0
    cardOneX=0  : cardTwoX=0
    cardOneY=0  : cardTwoY=0

    Call SetDefaultValues
    Call SetCurrentBack design

    'draw a nice background
    #1.g "down; fill 190 190 115"
    #1.g "backcolor 190 190 115"
    'trap mouse clicks:
    #1.g "setfocus; when leftButtonUp [checkIndex]"

    gosub [shuffleCards]

    'set xy location to start deal
    x=10:y=2
    for i = 1 to 24
        'set status of all cards to 0, which is face down
        Call SetCardStatus card(i), 0

        'deal cards
        Call DealCard hBox,card(i),x,y

        x=x+100
        if x>510 then   'move to next row
            x=10
            y=y+100
        end if
        playwave "card.wav",sync

       'pause 100 milliseconds between cards
        call Pause 100
        scan
    next
    wait


[checkIndex]
    clickCard=0:x=0:y=0 'reset values
    mx=MouseX   : my=MouseY 'mouse x and y location
    nCard=InitDrag(hBox, mx, my) 'discover index of card under mouse
    call AbortDrag     'release DLL mouse capture
    if nCard=0 then wait

    'Check to see if the user has already exposed this card.
    if nCard=cardOne then wait
    x=GetCardX(nCard):y=GetCardY(nCard)
    'remove card to restore tabletop
    call RemoveCard hBox, nCard

    'set status of cards to 1, which is face up
    Call SetCardStatus nCard, 1

    'deal card face up
    Call DealCard hBox,nCard,x,y

    gosub [readValue]

    'If all pairs have been removed, ask user if he
    'wants to play again.
    if gameWon=1 then
        if bestTurns=0 then
            bestTurns=turns
        else
            if bestTurns>turns then bestTurns=turns
        end if
        msg2$="Best score today: ";bestTurns
        #1.g2 "place 10 16"
        #1.g2 "\" ; msg2$; space$(100)

        msg$="You have won in ";turns;" turns.  Play again?"
        confirm msg$;answer$
        if answer$="yes" then
            'start a new game
            goto [new]
        else
            'disable mouse event trapping and wait
            #1.g "when leftButtonUp"
        end if
    end if
    wait


[readValue]
    'check whether this is first or second card
    if cardOne=0 then
        cardOne=nCard
        cardOneX=GetCardX(cardOne)
        cardOneY=GetCardY(cardOne)
        return  'leave first card up and return
    else
```

```
        cardTwo=nCard
        cardTwoX=GetCardX(cardTwo)
        cardTwoY=GetCardY(cardTwo)
    end if

    #1.g "when leftButtonUp" 'turn off mouse event while pausing
    call Pause 2000     '2 second pause to view cards
    #1.g "setfocus; when leftButtonUp [checkIndex]"

    oneVal = GetCardValue(cardOne)
    twoVal = GetCardValue(cardTwo)
    'ace=1,deuce=2....jack=11,queen=12,king=13
    oneSuit = GetCardSuit(cardOne)
    twoSuit = GetCardSuit(cardTwo)
    'returns 1=Clubs, 2=Diamonds, 3=Hearts, 4=Spades.

    'Remove cards from table --
    'they will be redealt if they don't match.
    call RemoveCard hBox, cardOne
    call RemoveCard hBox, cardTwo
    call SetCardDisabled cardOne, 1
    call SetCardDisabled cardTwo, 1
    turns=turns+1

    'See if cards match each other in suit and value.
    'If they don't match, turn them face down and redeal them.
    if (oneVal<>twoVal) or (oneSuit<>twoSuit) then
        'set status of cards to 0, which is face down
        Call SetCardStatus cardOne, 0
        Call SetCardStatus cardTwo, 0

        'deal card face down
        Call DealCard hBox,cardOne,cardOneX,cardOneY
        Call DealCard hBox,cardTwo,cardTwoX,cardTwoY
        call SetCardDisabled cardOne, 0
        call SetCardDisabled cardTwo, 0
    else
        'If cards match, increment pairs/score and don't
        'replace them on the table.
        call DrawSymbol hBox,3,cardOneX,cardOneY
        call DrawSymbol hBox,3,cardTwoX,cardTwoY
        pairs=pairs+1
    end if

    cardOne=0   : cardTwo=0
    cardOneX=0  : cardTwoX=0
    cardOneY=0  : cardTwoY=0  'reset for next try

    msg$="Score ";turns;"        Pairs ";pairs
    #1.g "place 10 420"
    #1.g "\" ; msg$; space$(100)
    if pairs=12 then gameWon=1 'flag that all pairs are removed
    RETURN


'setting new card back doesn't restart game,
'so new back won't show until new game is started:
[circles] design=1:goto [setDesign]
[blue] design=2:goto [setDesign]
[red] design=3:goto [setDesign]
[mountain] design=4:goto [setDesign]
[purple] design=5:goto [setDesign]
[music] design=6:goto [setDesign]

[setDesign]
    Call SetCurrentBack design
    'design can be 1,2,3,4,5,6 for 6 possible designs
    wait


[fillCardArray]
    'fill card array
    'cards 1 to 52 are in the first deck
    'cards 53 to 104 are in the second deck
    'use cards Jack through King in each suit, first deck
    card(1)=11  'jack of clubs
    card(2)=12  'queen
    card(3)=13  'king
    card(4)=24  'jack of diamonds
    card(5)=25  'queen
    card(6)=26  'king
    card(7)=37  'jack of hearts
    card(8)=38  'queen
    card(9)=39  'king
    card(10)=50  'jack of spades
    card(11)=51  'queen
    card(12)=52  'king

    'now use second deck, to fill second half of array
    for i = 1 to 12
        card(i+12)=card(i)+52
    next
    RETURN


[shuffleCards]

    playwave "shuffle.wav",async

    'now shuffle cards
    for i = 1 to 24
        newIndex=int(rnd(0)*24)+1
        tempCard=card(i)  'temp var to allow switching values
        card(i)=card(newIndex)
'this index now contains value from random index
        card(newIndex)=tempCard
'random index now contains value from other index

'now card(i) has switched values with a random card in the array
    next
    playwave "shuffle.wav",sync
    RETURN

[quit]
    for i = 1 to 24
        'remove cards from table
        call RemoveCard hBox,card(i)
    next

    gosub [fillCardArray]

   2'set xy location to start deal
    x=10:y=2
    for i = 1 to 24
        'deal cards, no shuffle
        Call SetCardStatus card(i), 1
        Call DealCard hBox,card(i),x,y
        playwave "Card.wav"
        x=x+100
        if x>510 then   'move to next row
            x=10
            y=y+100
        end if
    next
    call Pause 500     '.5 second pause

    'animation to end game
    for j = 1 to 24
```

```
        by=2:bx=10
        call ReturnDrag hBox,card(j),bx,by
        call Pause 100     '.1 second pause
     next
    call Pause 1000
close #qc:close #1:end

[about]
    notice "Memory Card Game ";chr$(169);" July 2011, Alyce Watson"
    wait
''''''''''''''''''''
'subs and functions:
Sub Pause ms
    'pause ms number of milliseconds
    calldll #kernel32,"Sleep",_
    ms as long, re as void
    End Sub

Function GetCardSuit(nC)
    'returns 1=Clubs, 2=Diamonds, 3=Hearts, 4=Spades.
    calldll #qc, "GetCardSuit",nC as long,_
    GetCardSuit as long
    End Function

Function GetCardValue(nC)
    'ace=1,deuce=2....jack=11,queen=12,king=13
    calldll #qc, "GetCardValue",nC as long,_
    GetCardValue as long
    End Function

Function GetCardX(nC)
    calldll #qc, "GetCardX",_
        nC as long,_    'index of card
        GetCardX as long 'x location of upper corner
    end function

Function GetCardY(nC)
    calldll #qc, "GetCardY",_
        nC as long,_    'index of card
        GetCardY as long 'y location of upper corner
    end function

Sub InitializeDeck hndle
    calldll #qc, "InitializeDeck",_
    hndle as long,r as long
    End Sub
```

```
Sub SetCardStatus nC,face
    'nC is number of card - 1-52 in first deck and
    '53-104 in second deck, if used
    'face:  0=facedown,1=faceup
    calldll #qc, "SetCardStatus",nC as long,_
    face as long,r as void
    End Sub

Sub DealCard hndle,nC,x,y
    'places card on window whose handle is hndle at x,y
    'nC is number of card - 1-52 in first deck and
    '53-104 in second deck, if used
    calldll #qc, "DealCard",hndle as long,nC as long,_
    x as long,y as long,r as void
    End Sub

Sub SetCurrentBack nV
    'nV can be 1,2,3,4,5,6 for 6 possible designs
    calldll #qc, "SetCurrentBack",nV as long,r as void
    End Sub

Sub SetDefaultValues
    'reset all card properties back to their default values.
    calldll #qc, "SetDefaultValues",r as void
    End Sub

Sub RemoveCard hndle,nC
    'removes a card from screen that was
    'drawn with DealCard, replacing screen background
    calldll #qc, "RemoveCard",hndle as long,_
    nC as long,r as void
    End Sub

Sub ReturnDrag hndle,nC,nx,ny
    calldll #qc, "ReturnDrag",_ 'automatic dragging
        hndle as ulong,_    'handle of graphicbox
        nC as long,_        'card to drag
        nx as long,_        'x location to drag to
        ny as long,_        'y location to drag to
        re as void          'no return
    end sub

Function InitDrag(hndle, x, y)
    calldll #qc, "InitDrag",_
        hndle as ulong, x as long, y as long,_
        InitDrag as long
    end function

Sub AbortDrag
    calldll #qc, "AbortDrag",re as void
    end sub

Sub DrawSymbol hndle,nV,nx,ny
    calldll #qc, "DrawSymbol",_
        hndle as ulong,_    'handle of graphicbox
        nV as long,_        '1=X 2=O 3=place holder
        nx as long,_        'x location
        ny as long,_        'y location
        re as void          'no return
end sub

sub SetCardDisabled nC, nV
    calldll #qc, "SetCardDisabled",_
        nC as long,_    'card to set
        nV as long,_    '1=disable,0=not disabled
        re as void      'no return
    end sub
```