Author's Preface

Welcome to Chinese Puzzle. This is a Solitaire/Patience game that I first learned many years ago as a single deck game, and later adapted to a two-deck game to provide more opportunity for choice of move, and hence skill. (I have even tried three decks, but just dealing the cards seems to take *forever*).

Please don't read anything into the name—it's just the name I first learned the game under. The only other name I have ever seen for this game is the somewhat uninspired “Gaps”, and I’ve never been able to come up with anything better.

This program was written primarily for my own amusement, and as a means of testing some ideas on animation on the Macintosh for another program I am working on. The program was by no means a major effort on my part, and I enjoyed writing it (most of the time - see below), and as such I don’t expect (or want) payment for it. I enjoy playing it, and if you do too I ask only that you make a small contribution to the charity of your choice. If you have no charity-of-choice, then allow me to suggest that you contribute to any agency which helps deaf children—as I am semi-deaf from a childhood illness myself, this is a subject dear to my heart.

“Why only most of the time?” In the course of writing this program, I was infested by the “Scores” virus, forcing me to shell out good bucks for a recovery utility (I consider that the bum that wrote that owes me $90.00), and had the power supply on my Mac burn out—literally just as I had finished the program (but not saved the source-code).

Feel free to distribute this program by any means you want, except:

a) You may not charge for it without my permission.

b) You may not alter this program in any way, and then redistribute it without my permission.

c) This documentation must accompany the program.

Fair enough?

I can be contacted at the EMAIL address A.Kirby1 on GEnie (preferred) or 72560,373 on CompuServe (checked less frequently)

Rules of the game

Setup

Chinese puzzle is played with two full (52 card) decks. Shuffle two decks together thoroughly, and then deal all of the cards into eight rows of thirteen cards each leaving enough space to the left of the first card in each row for one additional card. Take each of the eight Aces, and place them in any of the spaces to the left of the rows. Each Ace can go in any vacant space in the extreme left column.

(Variant one) Each of the four suits must be represented in the first four rows—rows five through eight must repeat the same order of aces.

(Variant two) The aces must be in (ascending) Bridge order—counting down from row one, they must be “Clubs, Diamonds, Hearts, Spades, Clubs, Diamonds, Hearts Spades”.

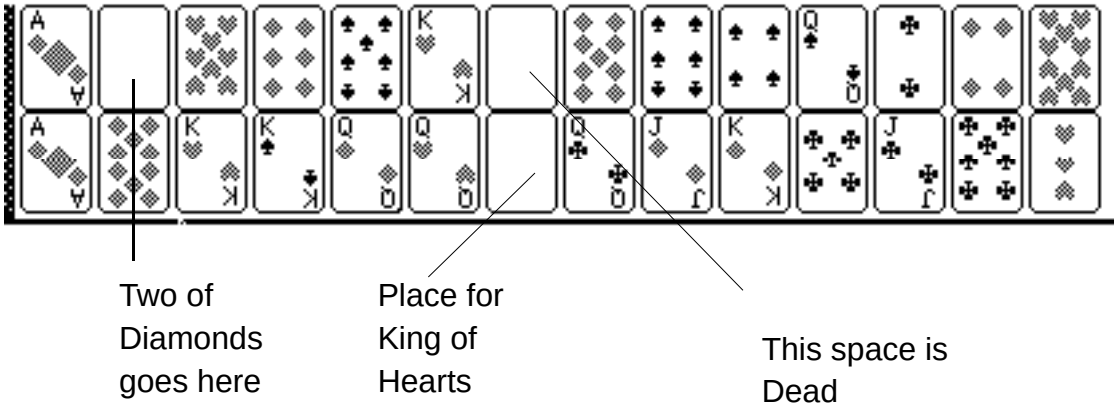
Neither of these two variations is imposed by the program, but you can impose your own restriction if you want.

Object

The object is to get all eight rows into sequence(Ace, Two,...Queen, King) and suit behind the Aces.

Play

Moving the eight aces will leave eight gaps in the layout. You fill these gaps by moving in either of the two cards which are next in sequence and suit to the card to the *left* of the gap. There is no card after a king, so a gap after a king cannnot be filled, and is dead…



(the number of dead spaces is shown by the count under the gravestone marker on the right of the screen)

Every time you move a card, it will leave behind a gap, which can be filled in the same way. In the above illustration, moving the King of hearts to follow the Queen would leave two spaces behind the seven of spades. You could then move the eight and nine of spades there.

Once a card is in a continuous sequence of rank and suit from the Ace it is in its final position, and cannot be moved again. In the above illustration, if you moved either two of diamonds to the spot behind the Ace, then it would be in its final position. If the following card were the three of diamonds, then it would also be in its final position—this is shown by the counts on the right of the rows.

Re-deals

Once there are eight dead spaces, you cannot continue, and all cards not in their final positions are picked-up and re-shuffled. They are laid out again in the eight original rows, leaving one space to the right of the last “locked” card in each row. This produces eight new gaps which control the movement of cards as before.

You are allowed to re-deal the cards either once or twice (the choice is yours—see the Control Menu). The game finishes when you either run out of deals (you lose), or get all 104 cards into their final positions (you win - visit the refrigerator).

Using the program

Starting-up

Start the program by double-clicking it or by selecting “Open” from the Finder's File menu in the usual way. The program will start-up and deal the initial layout for you. Two defaults are already selected by the program: you will be given two chances to re-deal, and the method for moving cards is dragging (see “Control Menu”, and “Moving the Cards” for details).

Moving the cards

Cards are moved either by dragging them (the default), or by clicking once to pick them up and a second time to put them down again. If you use dragging then the card will only stay “attached” to the cursor while you hold down the mouse button; if you use the two click method then the card will stay attached to the cursor until you click the mouse for a second time.

When you are moving a card, if it is in a legal destination then the spot will highlight to show you that you can drop the card here. When you “drop” the card (by releasing the mouse button, or by clicking a second time), the card will settle into its new spot if it was in a legal space, or will return to its original space if not.

If you try to move a card that cannot move the program will “Beep” - make the same mistake enough times, and the program will display an alert telling you exactly why the card cannot move.

Hints

Sometimes it can be hard to track down a sequence of moves: this program can give you some help. Suppose that your next move depends on shifting the Ten of Clubs from its present position, but that you can't locate the nine that you would need to move it to (pretty easy with tiny cards like these) - hold down the Option key and click on the card, and the cards before it in sequences will flash.

Equally, sometimes you might need to find a card that can move to an empty gap - if you “option-Click” on a gap then the card(s) that can move there will flash.

The hint option will never flash an illegal move—if you click on a locked card or a dead space, then the program will simply “Beep”. If you option-Click on a space, and only one card can move there because the other is in its final position then only the moveable card will flash.

Menus

Your usual assortment of desk-accessories. “About Chinese Puzzle…” will give you edited highlights of this documentation along with other credits.

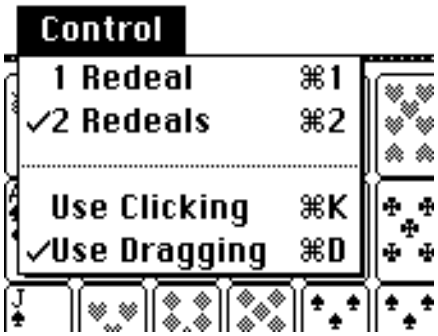
Only one item - Quit. Saving/loading games in progress may be added later.

This menu is only there for DA's to use, which is why it is dimmed when the program is active, and highlighted when a DA is active.

This menu contains only two items:

New Game — Allows you to abandon the current game and start another. You will be asked for confirmation before proceeding in case you selected this item accidentally.

ReDeal — Allows you to redeal the unplaced cards before you have eight dead spaces, as you will be able to tell that you are about to get stuck before the program can. You will be asked for confirmation before proceeding in case you selected this item accidentally.



This menu cantains two groups of either/or options:

1/2 Redeals — Select how many times the program should let you redeal the cards when you get stuck. These choices are only available when you start a new game, and will be disabled as soon as you place any cards

Click/Drag — Lets you select the kind of card-movement method you prefer. Clicking is easier to use if (like me) you have an old mouse that doesn't want to move sideways sometimes. You can change this option anytime.

Technical Notes

Compatability

I developed and tested this program on a Mac Plus. I have no way of telling for certain whether it will work on a 512K, but it should as I don't use any new ROM calls. It definitely works on a 512KE, and definitely does not work on a 128K Mac as it requires 256K to work comfortably. It does work under Switcher (v 5.1 was tested) and MultiFinder™, but please don't reduce the minimum size too much. This is a very small program, but it allocates a lot of memory for off-screen bit-maps.

Note to Switcher users: I tend to use Cmd-\ as an escape-from-program sequence - there may be some conflict with Switcher here.

My current system software is System 4.2, Finder 6.0, and everything works fine. I also have some INIT's which might be a threat to flaky software such as Shield™, AutoMac™, SuitCase™, Pyro™ and SuperSpool. The program works fine with all of these.

Techniques

The animation of the card movement is done by having three off-screen bitmaps allocated: ScreenPic (a copy of the current screen), BackPic (the current screen without the moving card), and CardPic (the cards themselves). The animation is done by (essentially):

```
CopyBits (EmptyCard, ScreenPic, srcCopy)
While StillDown do begin
        if CardHasMoved then begin
                CopyBits(BackPic, ScreenPic { cardWas }, SrcCopy);
                CopyBits(CardMask, ScreenPic, { CardIs }, SrcBic);
                CopyBits(CardPic, ScreenPic { cardIs  }, SrcOr);
                CopyBits(ScreenPic, ScreenBits { SectRect(CardWas, CardIs) },
        SrcCopy);
                CardWas := CardIs;
        end;
end;
if CardIs is validDestination then
        CopyBits(CardPic, ScreenPic, newDestination, SrcCopy)
else
        CopyBits(CardPic, ScreenPic, StartPosition, SrcCopy);
CopyBits (ScreenPic, ScreenBits, SrcCopy)
```

The Algorithm is complicated by the fact that there is the highlighting of the destinations to take care of, but that is the main part of the code

For Hackers...

Obviously you can fiddle with the program with ResEdit as much as you want (but please observe the rules listed on page 1). Note though that for reasons not unconnected with laziness, not all strings are defined in the resource fork—some are hard-coded in the program—so not everything is easily modifiable.

The most obvious choice for change is the card picture. All 52 cards are drawn in the one picture (in an off-screen bit map initially), as well as stuff like the gravestone, card mask, etc. If you decide to change the cards, then copy PICT id 257 from the resource file, and paste it into xxxxxPaint. Make any changes you want there (but every one of the individual pictures must stay in the exact same place), and then paste it back into the program with ResEdit.

Warning 1: Make sure that the resource id is correct when you paste it back.
Warning 2: Make sure that the picture is the same size as before (don't include any extra white space around the picture), as the bitmap that the picture is drawn into is the *minimum* size that can hold the picture. I created the card picture in SuperPaint, and used the Shift-Marquee to make sure that I only got what I wanted.
If the picture is too large, you'll get a bomb message - be careful!