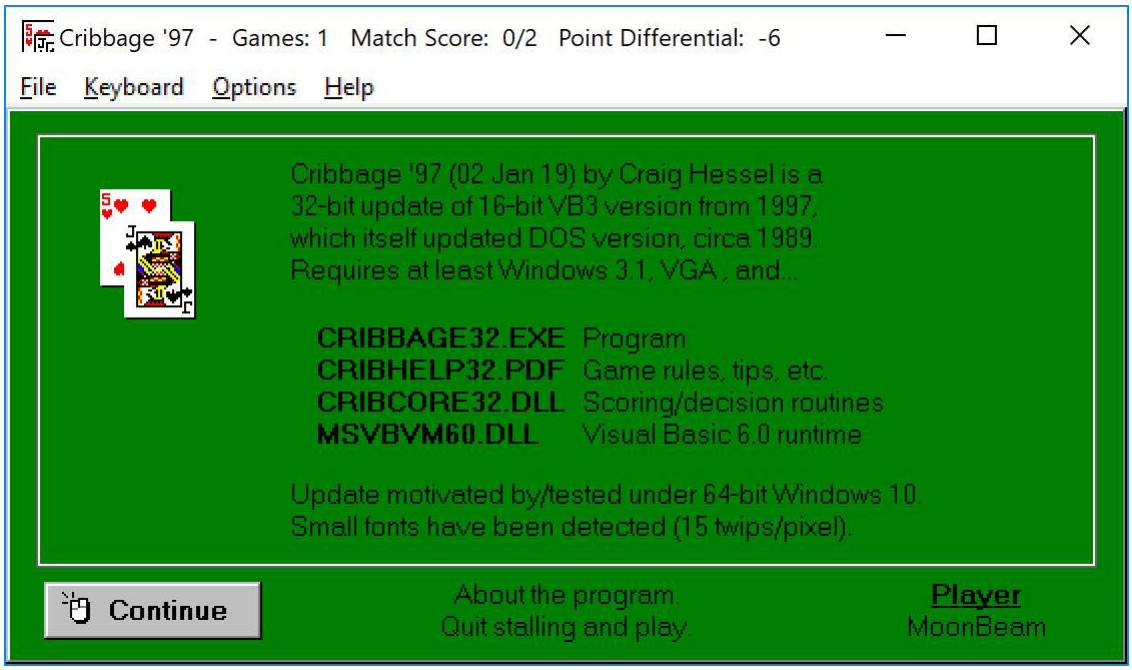
Version Notes
Why Cribbage?
File Menu
Keyboard Menu
Options Menu
Rules of the Game
Duplicate Match Play
Tips and Strategy
Cribbage Trivia
Programming Notes
Appendix: Custom DLL

Version Notes



The purpose of this update to Cribbage '97 (over 20 years later) is to enable the program to run under 64-bit (or 32-bit) Windows 10. To install program, unzip then copy files **CRIBBAGE32.EXE**, **CRIBHELP32.PDF**, and **CRIBCORE32.DLL** to a new folder on your Windows desktop. Run **EXE** file to start program. The required runtime version of

Visual Basic 6.0 (**MSVBVM60.DLL**) has been a component of all Windows operating systems since Windows 2000.

There have been three main functionality changes since the 1997 version of the program:

1. The File/Print Window option has been disabled. You may instead use the basic Windows Alt-PrintScreen feature to copy a form image to the copy/paste buffer, as was done to capture the above Help/About image.

2. Context sensitive help is no longer supported. Reference this PDF help file as needed.

3. The default INI file and player LOG files now reside in same folder as the EXE file. This behaviour may be undone a command-line argument (see Programming Notes).

Otherwise the program should behave as in 1997. In particular, the decision making routines are unchanged. A log file comparison of two 1000-game runs (from this version on my 64-bit Surface Pro 4 and from the 1997 version on an older machine) with all decisions made by the computer yielded identical output. I.e., CribCore32.dll (2019) behaves exactly as CribCore.dll (1997). This was a necessary prelude to any future changes since it provides a known 32-bit baseline for comparison testing. A potential change would be incorporating the discard decision making found in the 32-bit discard utility DS.EXE.

Why Cribbage?

If you've never played cribbage before, well.....good luck. The game is tricky to learn (see Rules of the Game). It was invented several centuries ago by an English nobleman who probably kept tweaking the rules until he was the only one of his day who could figure them out. The rules have such an odd mix of ways to score points that a beginner is apt to accuse his opponent of making them up as he goes along.

But cribbage grows on you. It's the kind of game that leads grizzled veterans to maintain meticulous records over the years of wins and losses and skunks, dollars won and dollars lost.

For the veteran player, this program provides automatic record-keeping. Not only that, it lets you play duplicate cribbage (see Duplicate Match Play). After all, the reason for record-keeping or playing for money (admit it now) is to see who's better at the game. Duplicate cribbage against a common foe -- the computer -- can help answer that question, or at least give you something new to argue about.

If you are a beginner, you can ignore these extra features and just try your luck against the computer. By default, the program gives you hints on which cards to discard or play and where to place your peg when scoring. It even provides a command button to make card plays and handle scoring automatically. You can disable the button and turn off the hints

once you get a feel for the game.

File Menu

The File Menu handles a few odds and ends.

New Match starts a new match immediately. Match score, game score, and statistics are all reset, but player options are retained.

Whenever you start the program or switch to a different player name, the program always picks up where the last session left off for that player. Use this option if you want to start a fresh match instead.

Change Player lets you select a player name from an earlier session or else type in a new name. Names may be up to 8 characters (letters/digits/hyphens allowed). The program then restores the last session for that player or starts with system defaults, if you entered a new name.

The program INI file saves restart information for up to a dozen player names. This includes option settings, match and game scores, match statistics, and current game state.

Different names allow several people to play on the same machine without mixing up the respective game states and settings. But you can also maintain several aliases, each with different groups of option settings, for yourself. E.g., you might want to keep a Match alias for a long-running ruthless match, a Replay alias for duplicate autoplay of the same match, and a Practice alias with friendly options enabled.

Show Statistics and Log shows updated match statistics and session settings for the current player. The tail of the log file (last 8K) for that player is also shown, if anything has been previously logged.

The statistics give running per-hand averages for dealer/crib/pone hand scores and dealer/pone pegging scores. Total penalty points are also shown. The final hand of each game is excluded from the statistics, since it is frequently incomplete or skewed. Some extra statistics, generally for extended autoplay sequences, are also provided.

The following sample shows statistics/settings after the 100th game of an autoplay match.

```
*********************************************************************
*    Games: 100  Match Score: 107/109  Point Differential: -178     *
*********************************************************************
First table excludes final (usually skewed and/or incomplete) hand
of each game and includes any points for jack cuts in peg averages.

            Hands        Hand Averages    Peg Averages Penalty
          Deal Pone    Deal  Crib  Pone    Deal  Pone   Points
          ---- ----    ----  ----  ----    ----  ----    ----
Moonbeam:  399  400    7.94  4.48  8.02    3.33  2.04       0 missed
Computer:  400  399    7.90  4.68  8.04    3.58  2.01       0 taken
```

```
Win Margin 1-5 -10 -15 -20 -25 -30 -35 -40 -45 -50 -55 -60  61+ Sum
           --- --- --- --- --- --- --- --- --- --- --- --- ---  ---
Moonbeam:   12   8  11   5   1   4   4   1   2   0   0   0   1   49
Computer:    4  11  10   4   8   8   4   1   0   0   0   0   1   51
Total:      16  19  21   9   9  12   8   2   2   0   0   0   2  100

Hands/Game    1-5   6   7   8   9  10  11  12  13+ Sum
              --- --- --- --- --- --- --- --- ---  ---
Moonbeam Won:   0   0   2   9  24  14   0   0   0   49
Computer Won:   0   0   5  11  19  13   3   0   0   51
Total:          0   0   7  20  43  27   3   0   0  100

Initial match seed (or file pointer, if applicable): 12345
Current session settings follow.

Hints (Mouse Icon): On
Scoring Details: On
Allow Autoplay: On
Allow Autoscore: On
Sort Hands: Off
2-3-4 Weights.
Friendly Corrections.
Normal Game (121 Points).
Loser Deals.
Default Sequence.
Disable Logging.
Default Deck.
Default Patter.
Default Sounds.
Default Green.
**********************************************************************
No log file to display.
```

Append Statistics to Log lets you append current statistics and settings to the player log file. A good time to do this, if you want to preserve the information, is at the end of a match. The statistics are cleared whenever you start a new match.

The name of log file is set under Options/Activity Log. Other automatic logging may be enabled under that menu. Appending statistics is a one-shot save, rather than automatic.

Exit quits the program. Session settings, game state, etc., are saved automatically.

Keyboard Menu

The program requires three basic actions from you during a game: selecting a card, scoring a play, and continuing after a pause. With a mouse, you take these actions by clicking on a card, dragging and dropping your back peg, and clicking on the Continue button. If Hints are enabled, you may also score by simply clicking on the target hole -- the mouse icon looks something like an upside down peg over the target.

For those who prefer keyboard navigation, a Keyboard Menu is provided. The menu is really only there to remind you of its shortcut keys. These are not typical of Windows shortcuts, but instead are the same as the keys used by the earlier DOS version of the program. The keyboard controls are particularly useful if you have disabled Autoplay and Autoscore (e.g., in a tournament-style game).

Card Selection Use the **Left Arrow** or **Right Arrow** to highlight a desired card, then press **Enter** to select it.

Peg Movement Use the **Up Arrow** or **Down Arrow** to move your back peg, then press **Enter** to anchor it. You always have the green pegs. At game start, the inside (upper) peg is the back peg.

Continue Just press **Enter**.

Two special keys are also recognized:

Automatic Pressing the **F10** key is the same as clicking on the game button. The button makes automatic plays for you and its caption changes according to game context. When the button is disabled (see Options Menu), the key is also disabled.

Minimize If the menu bar has focus, pressing the **Escape** key eventually takes you back to the game area. In the game area, the same key then minimizes the program to an icon or to the taskbar.

Standing on the **F10** key is a way to generate computer vs. computer games quickly. You might find the statistics from this interesting if you are a real cribbage diehard. Standing on **Escape** is a quick way to hide the program. Basically, treat this as the '"boss is comin'" key.

Options Menu

The Options Menu lets you adjust program features. Checkmarks next to menu options indicate which options are enabled. Submenus handle settings a little more involved than the first five on/off options. The defaults shown apply when you play under a name for the first time. After that, previous settings for the player name are loaded from INI file.

You may be content to play without ever changing the defaults, but I've tried to make the program flexible. In particular, you might have some fun with Card Deck Style and Game Over Sounds. On the other hand, if you are a true diehard, you can disable the frills and duke it out toe-to-toe with the machine.

Hints (Mouse Icon) On by default

With Hints on, the computer shows you what cards it would discard or play from your hand. It also shows the correct destination peg hole when you are scoring. It does this by changing the mouse icon when it is over the card or peg hole.

Scoring Details On by default

With this option on, an extra display box appears at hand scoring time. The box tells you how each of the three hands is scored. This should be helpful for beginners. You can turn off the display, and just turn it on when you are puzzled by a particular hand or when the computer detects a scoring error.

Allow Autoplay On by default

This option enables or disables the **Autoplay** button. There is only one button, but its caption changes according to game context. An automatic card play is just the discard or pegging play that the computer would make in your shoes. This matches the hints.

Allow Autoscore On by default

This option enables or disables the **Autoscore** button. Clicking on the button is the simplest way to score. With Autoplay or Autoscore disabled, you might find the keyboard controls easier to use than the mouse.

Enable autoplay and autoscore when you want to replay a match and see how the computer fares with the same cards you were dealt.

Sort Hands On by default

With this option on, all face-up hands are sorted low rank to high. In addition, the computer hand is sorted even when you see only the card backs. Your discards and the computer discards are sorted separately before being turned face-up. With sorting turned off, all cards are ordered as dealt.

Beginners prefer sorting their cards, since it makes scoring and discarding clearer. Experienced players will not sort their cards in real-life games, since observant opponents can gain useful information from this. You can gain a similar advantage when playing with sorted hands and the dog-eared deck (see Card Deck Style below). E.g., if the computer holds one dog-eared card at the far left, then shows that card to be a 5 during play, you know the remaining cards are rank five or higher. The computer has tunnel vision and does not notice sorted cards or dog ears when making its decisions. Sorting does not give an advantage to either side when the default deck or custom deck is used.

The first six submenus (grouped by menu separator bars) handle mostly technical settings. These are for folks who believe that if you keep score, then you nail down the rules and you play to win.

Match Point Weights lets you set the match point scoring weights for win-skunk-double skunk. The default is 2-3-4. Previous scoring is not affected, so you should make any changes here at the beginning of a match.

Scoring Mistakes/Friendly Corrections handles your scoring mistakes by making friendly corrections, without penalty. With this default option enabled, you are also only prompted to score when you have some points coming or for any hand scoring. This option and the next two are mutually exclusive.

Scoring Mistakes/Penalties but No Muggins is the typical way to handle scoring mistakes in tournament play. Ordinarily, you would also turn off hints and scoring details, disable autoplay and autoscore, and disallow use of the dog-eared deck. There are two differences here from friendly play.

First, penalties are in effect. If you underscore, you lose the points. If you overscore, your score is corrected and the computer takes the amount you overpegged. Note that Go points are scored separately from other pegging points during play. Don't enable penalties until you get accustomed to this.

Second, you are asked to score every possible scoring play (including game cuts, since a jack might be cut). Most cuts and pegging plays score zero, so get used to clicking on your back peg or just pressing **Enter** for zero.

Scoring Mistakes/Penalties and Muggins is the same as the previous option except that "muggins" is also enabled. That is, if you underscore, you lose the points AND the computer takes the points you missed. Muggins is a stiff penalty, so enable it only if you really like ruthless play.

Game Length/Normal Game (121 Points) sets the length of each new game at the normal 121 points. The current game is not affected whenever game length is changed. This option and the next two are mutually exclusive.

Game Length/Short Game (61 Points) sets the length of each new game at 61 points. The starting peg positions are moved ahead halfway.

Game Length/End Game Practice starts each new game with a dealer/pone score of 98/106. This is nearly an even game. It may also be the largest dealer deficit for which that statement can be made. Use this option setting to hone your endgame skills.

First Dealer for Game/Loser Deals is the normal cribbage default. The loser of a game deals first in the next game. This option and the next three are mutually exclusive.

First Dealer for Game/Alternate First Deals alternates first deal from game to game between the player and computer. This option is essential for duplicate match play. It helps insure each replayed game starts with the same dealer and same cards.

To insure same conditions, the computer also internally deals out extra dead hands to a total of 17 after each game. This guards against games that differ in length, even if the cards dealt out are otherwise the same. The conservative choice of 17 is probably overkill, since games as long as 12 hands are about as rare as double skunks. The average game is about 9 hands.

First Dealer for Game/Player Always First lets you deal first every game. This option can be used for handicapping. It can also be used to generate autoplay statistics to estimate the advantage the first deal gives.

First Dealer for Game/Computer Always First lets the computer deal first every game. As above, the option can be used for handicapping or generating statistics.

Card Source/Default Sequence sets the card source to the sequence determined by the default card generator. This option and the next are mutually exclusive.

Card Source/Card Data File sets the card source to any file you choose. This could be truly random binary data downloaded from some obscure Internet source, pseudo-random data determined by an algorithm similar to that used by the program, or else a specific card sequence you have constructed. When end of file is reached, the card source reverts to the default sequence.

Card Source/New Seed or Pointer allows you to change the random seed used by the default card generator. The initial seed for a new player is taken from the system time. The allowed range is -2147483648 to 2147483647. If you are using a data file as the card source instead, the option lets you set the file pointer to a value from 0 (beginning of file) to the file length less one (end of file). The option is included for duplicate match play/replay. The match statistics display shows the initial seed or pointer for the match.

See Programming Notes for descriptions of the methods for generating the default sequence and for converting data file bytes to a card sequence.

Activity Log/Disable Logging turns off automatic logging. This option and the next two are mutually exclusive.

Activity Log/Log Games Only automatically logs game history to the player log file. The log file name is set by option below.

Activity Log/Log Games and Messages automatically logs game history plus extra messages to the player log file. The messages mainly show option changes, with each message appearing before the hand in which the change occurred. A sample log file fragment with messages follows.

```
Resuming match from INI state for Moonbeam (seed from hand 3).

Game 19: 18/23 (-39)  Seed: 1458352401  01-02-2019 10:47:26

# Opp Com          Pegging          Opponent Dscd  Cut Computer Dscd
- --- ---  ----------------------  -------- ----  --  -------- ----
3  43  39  7h 8s Kh 6s.6d 9d 8c 7s .8s6s9d7s AhTd  6c  7hKh6d8c Kd3h
Card Data File: C:\AUTOEXEC.BAT
Hints (Mouse Icon): Off
4  53  66  Ah 3h 6d 5h 5d 5c 4s.9d  Ah6d5d9d Qc3c  7h .3h5h5c4s 7c8c
5  64  76  4c 4s 7s Jd 6s.Ah 9c 5s .4sJdAh5s 8c8s  2s  4c7s6s9c KcTs
6  74  95  3s 6d Js 4d 5s 3d.Ts 2d  3sJs5sTs JdAd  Kc .6d4d3d2d Ah6s
7  94 109  8c Ts 9c 4c.7c Ks Kc Ah .Ts4cKsAh 6s9d  Ad  8c9c7cKc 5sJd
8 104 119  Jd 4s Ah 9d 4c 2s.Js 4h  JdAh4cJs 6h9h  Kc .4s9d2s4h As7h
9 106 121  2h 7d Kc 4d 8d          .4h3d7d4d Th9d  Js  5d2hKc8d 5h6h

Game 20: 18/25 (-54)  Pointer: 377  01-02-2019 10:56:49

# Opp Com          Pegging          Opponent Dscd  Cut Computer Dscd
- --- ---  ----------------------  -------- ----  --  -------- ----
1  22   4  4s Ks 6s Ts.4c Js 6c 5s .KsTsJs5s 5c3s  Jd  4s6s4c6c KcAh
2  24  37  Ks 5c Kc 6d.Ah 4c 3s 4s  KsKcAh3s Jd6s  6c .5c6d4c4s TsJs
3  40  60  3c 9s 3h Ts 4c.6s 5s Js .9sTs6sJs AhKh  4h  3c3h4c5s Ks6d
4  44  85  3h Ah 6h Ad 7h 4h.Th Js  3h6h7hTh QhTd  Jd .AhAd4hJs 5d2h
5  57  95  2h 9h 3d 6h Jd Ah.4h Th .9h6h4hTh Kc4d  9d  2h3dJdAh 8d5h
Card Source: Reverting to default sequence (EOF reached)
6  63 108  6h Jd Js 5d.7s Ad 6d 2d  6hJs7s6d KsQd  7d .Jd5dAd2d AhTh
7  70 121  7d 4s 8h 8s.6c 8c 6s 7s .4s8s8c7s 5hTh  8d  7d8h6c6s TcJc
```

You may view the tail of the log file under File/Show Statistics and Log or view the whole file with a good text file viewer. The log may get too large for some viewers (e.g., Windows NotePad). It is ok to delete or edit the file outside the program.

Activity Log/Log File Name lets you specify the name and location of the player log file. The default file name is the player name plus file extension **LOG**. The default location is the EXE directory.

The remaining submenus handle some fun stuff.

Card Deck Style/Default Deck lets you play with a crisp new blue-backed deck. This option and the next two are mutually exclusive.

Card Deck Style/Dog Ear Deck dredges up an old tattered deck with several dog-eared cards initially. You may choose which cards are dog-eared (see below). If you are observant, this will help your game. The dolt computer does not notice the dog ears during play.

Card Deck Style/Custom Back lets you load a custom picture for the card backs. The picture will be stretched to the card size. The best size is 36x49 pixels, but almost any BMP or WMF file will do.

Card Deck Style/Save Template As lets you save the default blue card back to a bitmap file. You can start with this as a template and create your own custom card back. Use a Windows program like Paint for editing.

Card Deck Style/View or Mark Deck displays all 52 cards. When you are playing with a dog-eared deck, you may also click on the cards here (or type one of the 52 upper/lower case letters) to mark or unmark them. Up to 13 cards may have dog ears.

By the way, MoonBeam is the name of the fictitious dog next to the fire hydrant. The name is intended to suggest the evening scene. If you look closely, you'll sometimes see MoonBeam "marking" the deck, so to speak. I originally drew the picture with Dogbert sitting on the curb instead, but the consultant's fee might have been too steep.

Obnoxious Patter/Hide Patter hides the bottom line of the message display. During play, there are two lines of text prompts/comments at the bottom of the game window. The top line is neutral information. The bottom line, by default, is irritating drivel from your opponent. This option and the next two are mutually exclusive.

Obnoxious Patter/Default Text restores the bottom line of the message display to the default text. The default is what you might expect to hear from an obnoxious opponent.

Obnoxious Patter/Custom Text sets the bottom line of the message display to custom text you provide. The text file you load must be in a specific format (see the next menu option).

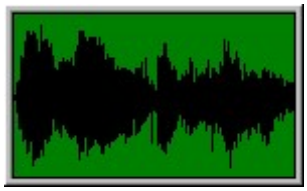
Obnoxious Patter/Save Template As saves a list of the current bottom-line messages to text file. Edit this file, then load it back into the program with the previous menu option. You may use a text editor Windows NotePad for editing. If you instead use a word processor, make sure that you save the file in text format. Comments in the template file tell you what to edit. Do not disturb other parts of the file, or else the program may not be able to load your messages successfully.

The first few lines of the patter template are shown next.

```
Patter File for Cribbage '97 (02 Jan 19)

; File header above marks this as patter file.  Edit text after
; equal signs as desired.  Ok to delete message lines.  Program
; will use its default text for any missing message markers.

ViewOrMark=You think maybe I pulled a five out?
ShowStatistics=Maybe you wanna save this gem?
SeedOrPointer=Like it's really gonna help.
About=Quit stalling and play.
NewPlayer=More cannon fodder.
MatchCut=Same stakes as last time, right?
...
```



Game Over Sounds/Disable Sounds turns off sound at end of games. This option and the next two are mutually exclusive.

Game Over Sounds/Default Sounds enables default sounds at end of games. The defaults are the first few notes of "Charge" for a computer win and of "Taps" for a loss.

Game Over Sounds/Custom Sounds substitutes your custom sounds for the defaults at end of games. See next two menu options. Pick sounds from the computer point of view, like the obnoxious patter.

Game Over Sounds/When Computer Wins lets you pick a custom sound file to be played when the computer wins a game. The sound is also demoed when you make the selection. My own favorite here is JAMESBRW.WAV ("Wow! I feel good...").

Game Over Sounds/When Computer Loses lets you pick a custom sound file to be played when the computer loses a game. As above, the sound is demoed when you make the selection. I like HALMIND.WAV here ("My mind is going. I can feel it...") from the movie 2001.

No sound (*.WAV) files are included with the program. Sorry. I didn't want any copyright hassles. But you may download a boatload of WAV files from the internet.

Background Color lets you choose green (default), cyan, or gray as the background color. Cyan is a color close to the Windows 95 desktop display default (remember -- this program dates back to 1997).

Rules of the Game

This section sticks to the rules, more or less, so it will be dry reading -- really dry reading. I've *italicized* references that are specific to the program.

Match Conditions

Matches consist either of a preset number of games or else are played to a preset number of "match points". The winner of the match is the player with the most match points. *The program just lets you keep playing as many games as you like until you choose to start a new match.*

Match points are awarded only for winning a game. There are no ties in cribbage. The

number of match points for a win varies, depending on how soundly you defeat your opponent. Several point systems are in common use. One system *(the program default)* awards 2 points for a plain win, 3 points for a "skunk", and 4 points for the rare "double skunk". Others count the double skunk as just a skunk. Still others (2-4-6 is common) give more weight to skunks and double skunks. In any case, win/skunk/double skunk have the standard meanings described next.

Game Conditions

Games are played until either player reaches 121 game points. These will just be called points from here on (to distinguish them from match points). The player reaching 121 points is the winner of the game. Scoring in cribbage is not done simultaneously, so frequently the loser of a game will have unclaimed points.

If the losing player has 60 or fewer points (rare), the game is a double skunk. If the losing player has 61 to 90 points, the game is a skunk. Otherwise, the game counts as an ordinary win for the player with 121.

The point ranges for win/skunk/double skunk are not arbitrary. Points are usually marked on a cribbage board, with the players using opposite sides of the board to keep score. Each player's side is further divided into "streets" of 30 holes each. The last street is called "home" street. Points are tracked by leap-frogging two pegs down the streets. The street boundaries coincide with the point cutoffs for win/skunk/double skunk.

There is no requirement to use a cribbage board to keep score, but it is convenient. The visual effect of the peg positions on the board is important to experienced players when they make decisions.

The program cribbage board is a 120-hole tournament board, rather than the usual 60-hole board. With the short board, players sometimes forget whether they are on the first or second pass around the board. The program also allows short 61 point games by giving each player 60 points initially.

Cut for Deal

Players cut for first deal in a match, with the low card by rank winning the deal. Aces are always low in cribbage. By standard cribbage rules, the first deal in each subsequent game is awarded to the loser of the previous game. *The program lets you optionally alternate first deals for duplicate match play or lets you handicap play so that the same player always deals first.*

The Deal

During games, the deal alternates between the players -- called "dealer" and "pone" -- on successive hands. It is important to know whether you are dealer or pone in any situation in cribbage, as many rules are applied differently depending on the roles.

To start a hand, the dealer shuffles the deck, offers to let the pone cut (he may decline), then deals six cards each, face down, alternately, to the pone and himself. These are tournament rules. In friendly games, opponents often dispense with the offer of a cut. *The program handles shuffling and dealing automatically.*

A non-standard rule *(ignored by the program and in real-life tournament play)* actually penalizes you for offering or accepting the cut.

Discarding

Each player next places two of his six cards face down into a common hand called the crib. Later, the crib will be scored by the dealer. The important choice of discards is discussed under Tips and Strategy.

The Cut

Next, the pone removes part of the remainder of the deck from the top, without exposing at any of those cards. The dealer then turns face up the top card from the stack remaining. This card is called the "cut" or "starter card". Later, the cut will be a fifth card common to the dealer hand, the pone hand, and the crib when those hands are scored.

The dealer is entitled to score 2 points now if the cut is a jack. The dealer must score the points (or announce his intention to do so) before the pone plays his first card, or else dealer misses the scoring opportunity. The pone must wait a "reasonable" amount of time to give the dealer a chance to score here.

The program handles this situation when penalties are enabled by displaying a scoring prompt after every cut. You tap Enter or click the back peg to score zero, or else you take 2 points. In friendly games, the scoring prompt is only displayed when a jack is actually cut.

Pegging Play

After the cut, the pegging play takes place. The pone plays first by turning up one of his cards and calling the count. The count is a running sum of the ranks of played cards (with aces counting as 1 and face cards as 10). The play continues, alternating between the two players. The cards are not mixed, so that the hands may be scored later. If one player can no longer make a play within the count limit of 31, he announces "go" and his opponent continues alone until he too can no longer make a play. When the count has reached 31 or when both players have announced "go", the count is reset to zero. Play then continues with the opponent of the player who last played a card now going first. This is repeated until all eight cards from both hands have been played.

The program displays the eight card plays in the same row as a visual convenience and indicates that the count has been reset by flipping the exposed cards face down. The

program also calls the count and announces "go" for you automatically during the pegging play.

These are the bare mechanics of pegging play. More important, though, are the ways in which points are scored during the process. This in turn affects your playing strategy (see Tips and Strategy).

Play Scoring

In the following, a pegging sequence refers to those cards played since the count was last at zero.

Go In any pegging sequence ending with a count less than 31, the player who last played a card is entitled to 1 point, called a "go" point.

15 or 31 Playing a card which brings the count to exactly 15 or 31 entitles the player to 2 points.

Same Rank Within a pegging sequence, playing a card that matches the rank of the last 1, 2, or 3 played cards entitles the player to 2, 6, or 12 points, respectively. That is, score 2 points for a pair, 6 points for three-of-a-kind, or 12 points for four-of-a-kind.

Run Within a pegging sequence, playing a card which completes a run of three or more cards entitles the player to 1 point for each card in the run. A run is a series of cards of consecutive rank (aces are always lowest rank in cribbage), in any order.

These scoring plays can be combined on a single play. For example, the card sequence 4-6-5... at the start of pegging entitles the pone to five points when he plays the 5, since the count is at 15 and since the 5 completes a run of three cards. In cribbage jargon, this would be announced as "Fifteen two and a run of three for five".

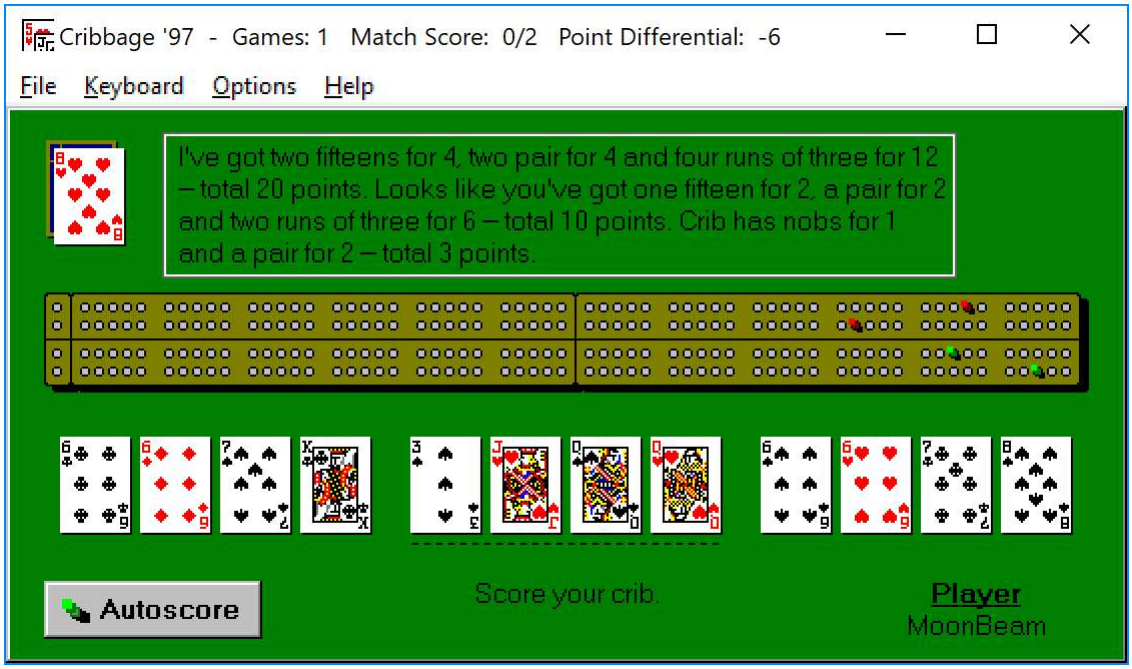
Remember -- scoring combinations cannot extend beyond one pegging sequence. Flushes (discussed in hand scoring below) do not count for anything during pegging play.

Cribbage players frequently combine "go" points with other pegging points when possible. *However, the program requires "go" points to be scored separately. Sorry about the inconvenience -- couldn't avoid it easily.*

Hand Scoring

Once pegging play is completed, the three hands -- pone hand, dealer hand, and crib -- are scored in exactly that order. The cut is a fifth card common to all three hands. Crib points are scored by the dealer. The scoring rules are identical for all three hands except

for the way flushes are scored in the crib (see below). As in the pegging, there are several ways to score points. The total score for a hand will usually be a combination of several ways.



Same Ranks Four-of-a-kind scores 12, three-of-a-kind scores 6, and each pair scores 2.

Runs Each distinct run of three or more cards scores 1 point for each card in the run. As long as two runs differ by at least one card, they are considered distinct. Again, remember that aces are always low in cribbage. A-2-3 is a run but Q-K-A is not.

Nobs If one of the four retained cards in the hand is a jack whose suit matches the suit of the cut card, then score 1 point for "nobs".

Flush If the four retained cards in the pone or dealer hand are all the same suit, score 4 points for a flush. If, in addition, the suit of the cut card also matches, score a fifth point. In order to score a flush in the crib, all five cards must be the same suit. This scores 5 points.

15's Each distinct combination of cards whose count-value sums to 15 scores 2 points. As in the pegging, the count value of a card is the same as its rank, except that face cards all have a count-value of 10. Aces have a count-value of 1.

As a short exercise, try scoring these two dealer hands: 5(cut)-5-5-5-J and A(cut)-A-6-7-8. Assume the jack scores nobs in the first hand and that the four retained cards in

the second hand are the same suit. The correct scores are in the Cribbage Trivia section.

Once the crib is scored, the hand is completed and the pone becomes dealer for the next hand, and so on, until the game is concluded.

Penalties

In friendly games, opponents will often call attention to each other's scoring mistakes and allow the player in error to make a correction. *For simplicity, the program makes corrections for you in a friendly game.*

In ruthless games, penalties are enforced for taking too many or too few points in any scoring situation. As with match scoring rules, there are a few variations in common practice. In tournament play *(and in the program when penalties are enabled)* these rules are enforced:

1. If you score too many points, your score must be corrected and your opponent takes the excess points you tried to take.

2. If you score too few points, you have in effect already penalized yourself, so nothing else is done.

Some folks are more ruthless than this, and in the second case will take the points you missed. This is called "muggins". *Including muggins is a program option.* A further tournament rule in practice *(not enforced by the program)* is that if you take too many points, and in doing so claim victory, you forfeit the game immediately.

The computer prevents several other problem situations from arising, e.g., misdealing, inadvertently exposing cards, improperly calling Go, etc.

Duplicate Match Play

Here is a suggested scenario for duplicate match play. There are no official rules for this; duplicate play is impractical without a computer. The intent of duplicate play is to get a better comparison of skill levels by removing some of the luck of the cards from the game. You'll either love or hate duplicate play. You are not playing against the computer per se, but against others playing with exactly the same cards as you. Whether you think the computer plays well or poorly, at least it plays consistently, which is exactly what you want here.

Agree beforehand on a random seed, number of games, and option settings. Typically, you should disable any program options that would help players. That is, turn off hints and scoring details, disable autoplay/autoscore, and disallow use of the dog-eared deck. Set the penalty level and match point weights as desired. Turn on logging of games and messages. Most importantly, insure that first deals are alternated. To start each match,

set the random seed as agreed on, then click on New Match. Play the agreed on number of games. Afterward, append statistics to the log file. The log files hold session histories for later comparison. The matches can be played serially on the same machine or simultaneously on several machines. Just insure that players cannot view each others' sessions.

You can also generate a session history of the computer vs. itself by turning on autoplay and autoscore, and then standing on the **F10** key for the duration of the match. This can be used as a "par" session (e.g., if you are the only player besides the computer).

The best match point differential determines the best player in this scenario. E.g., a 12-game match score of 10/14 for net of -4 is better than a 12-game score of 11/16 for a -5 net. The 10/14 match had no skunks, while the 11/16 match had several. As a tiebreaker, extend the match a few games at a time as needed. Or instead use best total game point differential. The running point differential is shown in the window title and in the log file.

Getting match scores of 10/14 *and* 11/16 with the same cards may be unlikely. But you'll be surprised at how often the match point results will vary. Close games and near-skunks are critical, as they should be. Purists will prefer not to use game point differentials as a tiebreaker, since playing strategy should be based solely on win/loss/skunk considerations.

If you lose a duplicate match to someone, you can argue that there is still some luck involved and that, at best, it only shows who did a good job of figuring out the computer's style of play. If you win, gloat.

These command-line parameters may help with setting up and managing duplicate matches:

/LOCK	Locks all option settings and disables New Match and Change Player.
/PLAYER=Name	Starts program with specified player name.
/DEFAULTS=Options	Sets default options (applicable only for a new player name). The options list format matches the INI file Options entry. Any space terminates the list. A short list is assumed to have trailing zeroes.

Examples:

CRIBBAGE /Lock /Player=Practice
CRIBBAGE /Lock /Player=Smith **/Defaults=**31415926,0,0,0,0,0,0,1,0,1,0,2,0,0,0,0

The first example starts the program with Practice as the player and locks all loaded settings until the session is ended.

The second example starts the program with Smith as the player and locks option settings. If Smith is a new player (not in INI file), the initial Seed is set to 31415926 and the five on/off options (0,0,0,0,0) are turned off. The next six technical settings (0,1,0,1,0,2) indicate 2-3-4 Weights, Penalties but No Muggins, Normal Game (121 Points), Alternate First Deals, Default Sequence, and Log Games and Messages (to default Smith.LOG in the EXE directory). The last four zeroes indicate Default Deck, Hide Patter, Disable Sound, and Default Green. If player Smith inadvertently closes the program then restarts with the same command-line parameters, the defaults are ignored and his session continues with the settings and state now loaded from INI file.

Note that 0/1 represent off/on settings and 0/1/2/... represent option submenu settings. The order of the settings matches the top-to-bottom menu display order. If you are unsure of the Defaults parameter, set up options from the game menu as you would like them, exit the program, and then inspect the Options entry in the INI file.

The intent here is to help a director orchestrate the chaos at tournament start and funnel most problems to that point in time. The director can step players through a last minute edit of player name and initial seed in the launching PIF or Program Properties section. These are the two values you most likely wish to hold off specifying until just before the match starts. The command-line can be set up beforehand modeled after the second example. Once the director has verified that everyone is getting the same cards on the initial match cut, things should proceed smoothly -- hopefully.

Each player should be instructed to stop and minimize the program after the agreed on number of games. Log files can be collected (after statistics are appended) for review.

These command-line parameters are my best guess as to what is needed for duplicate match play tournaments. If you try this and have any suggestions for improvement, let me know.

Tips and Strategy

This section is intended for those new to the game. Basically, we'll look at discarding and pegging play, with a hint or two about positional play.

Discarding Strategy

Beginners usually find discarding more difficult than pegging because there are more choices. There are 15 ways to discard from a hand, while you never have more that four choices when pegging. Besides that, there are lots of considerations that affect your decision. You want your final hand to score well. You want your discard to either help or hinder the crib. You want to keep good pegging cards. Finally, you want to weigh all this against the game situation, which dictates whether you should be cautious at the moment, or aggressive. This is too much to handle at once, so let's break up the problem

into manageable pieces.

For starters, let's look at discarding from the perspective of making your final hand score the best possible. This is usually the most important consideration in discarding. Hand scoring for dealer and pone (that is, excluding peg play scoring and crib scoring) accounts for over 60% of the points in cribbage.

The unknown in the final hand score is the cut card. A computer can average in the effect of the cut when it assesses a discard. For you and me it's a different story. We rely on experience and rules of thumb. A good rule of thumb here is just to look for the four cards that by themselves add up to the most points. The scoring rules covered previously for five-card hands work just as well with four-card hands. You can estimate about a quarter point per jack for possible "nobs". It still may seem like a lot of work to evaluate all the possible four-card combinations in a hand, but after playing awhile, you'll soon notice that most of the combinations aren't even worth looking at. At this point, just try to narrow down your choice to two or three possibilities that score the highest.

Next, consider the effect the discard has on the crib. As dealer, naturally you want to throw good cards, since you get to score the crib later. What are good cards? A pair or two cards that form a 15-count are good cards, since they are a sure two points. Other good cards are two cards consecutive in rank, like 6-7 or 3-4. These increase the chance of runs in the crib. Also, a 5 with any other card, a 2-3 combination, and an A-4 combination are all good discards. 5's are easily the most important cards in the game of cribbage. The abundance of 10-count cards (face cards and tens) in the deck means that 5's will be part of 15-count card combinations (which are worth two points) more frequently than any other rank.

As pone, you take the opposite view. You want to avoid throwing the card combinations that help the crib. If you can, you would like to throw cards like 10-K, 9-K, or 9-Q. As a rule of thumb, almost never throw a 5 into your opponent's crib.

For a detailed comparison of discards, see the tables in the Programming Notes section.

Now look back at the two or three four-card combinations that score highest and check the corresponding two-card discards for their effect on the crib. If you still have two choices that seem pretty even, a good rule of thumb for breaking the tie is to save the hand with lower ranking cards. These tend to improve pegging potential. Another tiebreaker is opponent tendencies. If your opponent is very conservative, and rarely throws good cards in your crib, you should tend to avoid wasting your own good cards by throwing them there also.

Another good rule of thumb when discarding is never to break up a "double run". A double run is a combination like 2-3-4-4 or 9-10-10-J, i.e., a combination of two runs and a pair. This guarantees you at least eight points and has potential to score much higher with a favorable cut.

Let's look at a few examples, starting with an easy hand. Suppose as dealer you hold 2-3-4-5-9-10. There are three ways to save five points: 2-3-4-9, 2-3-4-10, and 3-4-5-10. The corresponding discards are 5-10, 5-9, and 2-9. Since it's your crib, the 5-10 discard is best, since it's two sure points and 5's are great crib helpers. If you are pone with the same hand, then the 2-9 discard is best.

Now suppose as dealer you hold 2-3-4-4-7-7. Keeping either 2-3-4-4 or 4-4-7-7 saves eight points. The corresponding discards are 7-7 and 2-3. These are both good discards. Save the double run. It's a good rule of thumb. But what if you are pone? You hate these kinds of decisions. The safest discard is 4-7. In situations where extreme caution is dictated, many players will choose this, even though it means breaking up a double run. If it's early in the game, throw the 7's.

Finally, a trick hand. As dealer or pone, you hold 2-2-3-3-6-6. It seems clear cut, since 3-3-6-6 is worth eight points. You might be surprised to find that experienced players will frequently discard the 6's. The rule of thumb fails us here, since 2-2-3-3 has tremendous scoring potential. Sometimes two birds in the bush are better than one in the hand. If cribbage were too easy, it wouldn't be nearly as interesting.

A last word of encouragement on discarding. If you are a beginner, it can be frustrating to play an opponent who never needs more than a couple seconds to discard. First, he has probably been playing for years. He should be a little faster at it. Second, realize that this can work to his DISadvantage. There may be some type of hands he discards one way just because he's always done it that way, even though it's not the best way. You are new to the game and still think before acting. Make it work to your advantage.

Pegging Strategy

Although fewer choices make pegging seem simpler, good rules of thumb are harder to come by. Opponent tendencies should have a larger influence on pegging decisions than on discarding decisions. Keep this disclaimer in mind when reading on here.

Pegging play, by the way, is the weakest part of the program's game. It is a little too predictable and does not follow the above advice -- it does not try to adjust its style according to your playing tendencies.

A 4 is generally considered the best lead by pone. This prevents the dealer from reaching a count of 15 on the next play and forces him to make a play that gives the pone the opportunity to reach 15. As pone, however, you should occasionally decline to lead a 4, to keep your opponent off guard.

Leading from a 5-combination as pone is also common (e.g., leading a 3 if you also have a 2). Almost a third of the deck is made of 10-count cards, so dealer likely has one or more. If he plays one, pone can score two points. The dealer knows this, however, and will often decline to play a 10-count card next.

A 5 is a poor lead. You might be tempted to lead a 5, for example, from 5-J-Q-K at the end of a game if you are 11 points from going out. You are trying to lure your opponent into playing a face card so you can pair up. The unusual lead instead will trigger all kinds of warning signals in your opponent (even if you "cleverly" pretend the 5 slipped out accidentally). Instead, play one of the face cards, and hope that your opponent will routinely (and incorrectly in this instance) play a 5 to score two points. Then you may pair the 5 for your eventual winning points.

If you are dealer, you should generally assume that pone has a 5 and try to trap it. The pone almost never throws a 5 in your crib, so he is more likely to have a 5 than any other card by rank. He will hold a 5 after discarding about four hands out of every ten. As pegging proceeds, the play of 10-count cards, 4's, or 6's by pone should reinforce this initial assumption, while other plays weaken the assumption.

Though obvious to say so, remember what you discarded and what was cut. Those are three known cards your opponent cannot hold. You'll sometimes hear a real-life opponent say "Pair the deck" as his first lead pairs ups the cut card. This has several possible implications. He may be playing the percentages to reduce your chance of pairing his play. That is, he is including the extra known cards in his play decision. Or, psychologically, he may be trying to heighten your anxiety that he got a favorable cut. On the other hand, because he could have made the lead without comment, he may actually be trying to lure you into pairing the card. This should sound a little like the poison duel in "Princess Bride". Just make the simple assumption that he is playing percentages.

As the pegging winds down, and you have seen a few opponent cards, you should assume the opponent's remaining card or cards fit best with what he has shown. E.g., if your opponent has shown 6-8-9 already, guard against a 7 first, then against a 6 or a 9 next. If you play a 2, and he surprises you by pairing the 2, don't consider that as losing points. When the hands are scored, his total will be poor and that will more than offset those two pegging points.

From the opposite point of view, if you've got a discouraging busted hand, stay sharp in the pegging. Because your cards don't fit well together, your opponent likely will be mis-guessing what you have. You'll have a good chance to salvage some points.

A good piece of pegging advice for beginners is NOT to make scoring plays automatically. While you may end up making the play anyway, you should always do a damage assessment first and consider how your opponent will follow up. Give your opponent credit. Particularly early on in the pegging, when he has more freedom in card selection, he will be trying to guard against your possible scoring plays.

The toughest decision in pegging is whether or not to pair an opponent's card. You stand to gain two points, but risk giving up six points. Four points is a painful swing in the pegging. Tournament players are overly cautious in this respect, probably thinking that skill will carry the day anyway. Beginners, on the other hand, frequently pair up without a second thought. Probably the best advice here is to watch your opponent's style of play.

If your opponent tends to play from his pair early in the pegging, then you should tend to decline the early pairing opportunity. Positional strategy, discussed next, affects both discarding and pegging.

Positional Strategy

Experienced cribbage players often use the phrase "playing position" (or "playing shape"). This means playing aggressively or cautiously depending on the game score, i.e., on the peg positions on the board. I'll use an analogy here. As a driver, you should know enough not to keep your eyes on the bumper of the car in front of you. You should look well ahead for potential accident situations.

In cribbage, just watching to see whether you are a couple points behind or ahead of your opponent is like staring at the bumper in front of you. You may think you're in a close game and you may even gain a couple points as you turn the corner onto home street, but then kapow! An accident. There you are at 103 points and your opponent is at 99. But he's dealing next and you are pone. Think about it. Who's probably going to win?

The dealer on a typical hand scores 16 combined points. The pone typically scores 10 combined points. So he is likely to be around 115 as pone next hand, while you sit glumly dealing at 113. Then you think back a couple hands to the furious pegging play where you both scored a couple of times on run combinations and you had a net gain of a couple points. What if you had blocked the runs by playing your king? Take away 10 points from your score and 8 points from his and the score would be 103 to 107. Then you could have had a real battle down home street.

This is what positional play is all about. Think ahead. Try to maneuver so you will be dealing when you have 95-100 points. Sometimes it will take risky pushes in the pegging. Sometimes it will mean breaking up your hand to throw real clunkers into your opponent's crib.

The final tip here comes from computer simulations. Most cribbage games require 8, 9, or 10 hands to play, with 9 hands as easily the most common. Not surprisingly, the first dealer has a significant statistical edge in 9-hand games, since he has the extra deal. This suggests that the first pone in a game should do his level best to steer the game toward an 8- or 10-hander by risky or cautious play. The very best players probably do this instinctively.

Cribbage Trivia

Some of these are obvious, but others may surprise you.

1. The highest scoring hand in cribbage is four 5's and a jack, when the jack counts for nobs. This scores 29. Tournaments often pay out bonus money for 29-hands.
2. The most frequently mis-scored hand in cribbage is probably A-A-6-7-8. This scores

13 (or 17 with a flush). The A-A-6-7 combination for 15 is often overlooked.

3. The only cribbage hand that cannot be improved by the cut is four aces.
4. The dealer in cribbage is guaranteed a point in the pegging (unless pone goes out first).
5. Any cribbage hand containing a 5 (or cards that add up to 5) is guaranteed to score at least 2 points.
6. No matter what you discard, there is some possible three-card combination that can bring the crib score to at least 12 points.
7. The lowest hand point total impossible to achieve is 19 points. Worthless hands are sometimes called "19-pointers".
8. The largest hand improvement a cut can produce is 20 points, i.e., when a 5 is cut to 4-4-6-6.
9. The largest improvement a cut can produce for a hand which is worth nothing initially is 14 points, e.g., when a 5 is cut to the crib holding of 3-4-6-7 and all five cards are the same suit.
10. Any pairing cut to the hand 6-7-8-9 brings the hand score to 16 points (20 if the holding is a flush).

The following results are from computer simulations, but seem reasonable.

11. The most common pone hand is 5-10-J-Q, with 5-J-Q-K, 5-9-10-J, and 5-6-7-8 as the next three runners-up. The dealer top four in order are 5-10-J-Q, 6-7-8-9, 5-J-Q-K, and A-6-7-8. Together, these account for about 3% of all pone/dealer hands.
12. On average, the best discard by pone (and worst by dealer) is 10-K.
13. Dealer and pone hands each average about eight points (pone average is slightly higher), while the crib averages about four and three-quarter points.
14. Dealer averages about one and a quarter points better per hand than pone during pegging. A combined average of five and a half pegging points per hand is typical, but may be lower in cautious tournament play.
15. The player dealing first in a game wins 55% to 60% of the time, if the players are evenly matched.

Programming Notes

The original DOS version of the program (a 15K COM file, with TXT file for documentation) was written in assembly language in 1989. Most of the effort then went into the discarding and pegging decision routines.

For the 1997 Windows version, the core routines were rewritten in C and compiled/linked as **CRIBCORE.DLL** (20K). The display interface was rewritten in Visual Basic 3.0 (VB 4.0 seemed like overkill) and compiled as **CRIBBAGE.EXE** (140K). This in turn required **VBRUN300.DLL** (400K), the Microsoft Visual Basic 3.0 runtime library. The help file **CRIBBAGE.HLP** (60K) was created using the Microsoft Help Compiler.

For this 2019 version, the display interface was upgraded to Visual Basic 6.0 as

CRIBBAGE32.EXE (308K). The core DLL routines were upgraded from 16-bit to 32-bit C language as **CRIBCORE32.DLL** (60K). The help file **CRIBHELP32.PDF** (1010 KB) was produced as a PDF print-to-file from a WordPad edit of the 1997 source RTF help file. The Visual Basic 6.0 runtime file **MSVBVM60.DLL** (1354 KB) is not distributed with the progam since it has been included with every Windows operating system since Windows 2000.

If you are a C programmer with some idle time, you may also create a custom **CRIBALT32.DLL** to handle discarding/pegging decisions (see Appendix: Custom DLL).

Startup:

The main EXE file contains about 15K in bitmaps/icons. The 52 card images are constructed at program startup from a 16-color bitmap consisting of the card numbers, suit pips, and face card images. The board and default card back are drawn rather than loaded from bitmap. Image controls are used instead of PictureBox controls for the most part.

The program manually redraws its own bitmap images as needed to simulate transparent regions. You may, however, load a picture with true transparent regions as a custom card back. E.g., try loading an Icon file or a Windows metafile as a card back.

On startup, the program also scales the client area (including most controls, font sizes, etc.) to a fixed pixel size, independent of the Windows large/small/custom font size. This seemed to be the best way to preserve program appearance, but required undoing the scaling that VB does automatically. Font appearance can vary, since font increments are not continuous and since font availability depends on your machine environment.

API Calls:

Direct Windows API calls are used for card construction, common dialog interface, INI file read/write, program instance check, Windows version check, About info, and sound playing.

INI File:

By default, **CRIBBAGE32.INI** and player log files are located in same folder as **CRIBBAGE32.EXE**. You can force reversion to 1997 behaviour by passing parameter **/OLDINIPATH** as an argument to the EXE on startup. In 1997, this would place the INI file in the C:\Windows folder. However, more recent versions of Windows will place the file in folder c:\users\username\AppData\Local\VirtualStore\Windows. The AppData tree might not be easily visible to you. The OldINIPath behaviour is reasonable if multiple people execute the same copy of **CRIBBAGE32.EXE.** But if you install the program files in a folder on your Windows desktop, this is not an issue.

The **CRIBBAGE32.INI** file contains a Global section and one section for each player. The file is read at program start and written to at program end, with additional read/writes on player changes. If option settings are locked (see Duplicate Match Play), state updates are written to file more frequently to help guard against power loss problems (inadvertent or intentional).

The player sections may each have Options, State, Statistics1, Statistics2, DogEars, CardBack, DataFile, PatterFile, WinSound and LossSound entries. These preserve Options Menu settings, match/game state, match statistics, dog-eared card list, and most recent file names. The Options entry begins with the Seed or File Pointer. The Options Menu settings then follow in order. The State entry is preceded by a check value, so that an edit of the State entry or a change in the reconstructed hand (from the Seed or Pointer/DataFile entry, as applicable) will be detected. If an edit is detected, the State entry is ignored. This protects the program from the possibility of hanging when restoring the match/game state.

The Global section contains at least Version and Players entries. The Players entry lists up to a dozen players, with the names (most recent is leftmost) serving as section headers. Beyond a dozen, the least recent entry (rightmost) is removed from the list and its section is cleared. The Version entry allows this and future program versions to recognize and handle wrong-version INI files. Currently, a wrong version is handled by clearing known INI entries to guard against possible conflicts and then by stamping in the current version. Three additional Global section entries that you may manually adjust in the INI file are described in the Appendix: Custom DLL sample code.

Card Generation:

The default random number stream is generated by `Seed = (Seed + 1) * 0x711CB3A5`. The constant must end with ...101 binary for the generator to cycle through all possible 4-byte values. The high 6 bits of Seed determine the next card, with the top 2 bits determining the suit and the next 4 bits determining the rank. If the rank is not 0-12, the next value of Seed is fetched. Hands are dealt by generating 13 distinct cards, throwing out duplicates. The 13th card is the cut and the pone/dealer hands are the odd/even cards, respectively, from the first 12.

A data file stream is handled similarly, except that the low 6 bits of each byte are used rather than the high 6 bits of each 4-byte Seed. The data file is scanned sequentially. If end of file is exceeded (or on any open/read error), the program reverts to the default random seed method. The last file pointer becomes the initial seed at that point. If the file is not very random, it may be necessary to scan quite a few bytes to complete a deal. True random file data should average about 18.2 bytes per deal and 17 times that per game (including the internal dead hands dealt out to allow for duplicate match play/replay).

Decision Routines:

The discard decision routine is fairly straight-forward. The program cuts the 46 outstanding cards to each possible 4-card holding to assess its average score and maximum/minimum possible scores. A precomputed table (one for pone and one for dealer) with 91 entries holds the average expected crib score for the 91 possible discards by rank. The expected crib score for the discard is fetched from the table. A separate routine computes a risk value based on the peg positions. The expected crib score is weighed by the risk value, and then added to (dealer) or subtracted from (pone) the average score for the 4-card holding. The maximum (if reckless risk) or minimum (if cautious risk) score is risk-weighed and added in also. A small adjustment is made for pegging potential. The discard with the best result is chosen. Near game end, if any choice guarantees enough points to go out, only those choices are assessed, with the assessment based only on safe pegging considerations.

The pegging decision routine is recursive, with consideration of own play, opponent next play, then own next play at a minimum. If opponent play scores, lookahead is extended to own play following non-scoring opponent play. Choice is determined by summing own play scores and subtracting risk-weighed opponent scores, with opponent plays further weighed by estimated likelihood that opponent holds card of that rank. The probability of each rank for opponent is estimated from 1820-entry table (one for pone and one for dealer), with adjustments for inferences made during pegging. Table holds precomputed frequencies for the 1820 possible 4-card opponent holdings by rank.

Tables:

The tables mentioned above were constructed independently of the random card routine. I avoided Monte Carlo techniques so that the decision strategy would not be biased toward the very random number generator used in actual play of the game.

A simplified discard decision routine (assuming neutral risk and ignoring flushes, pegging potential, and endplay) was used to discard as pone and as dealer from all possible 6-card hands by rank. This determined the 1820-entry frequency tables (for held cards) and new 91-entry average crib score tables (for discards). The process was repeated iteratively until the average crib score tables and the hand frequency tables stabilized. Some horsepower was needed for this -- several hours on a Pentium 166 MHz. For the earlier DOS version on a much slower machine, some further simplifications had yielded decent, but cruder tables. The current discard tables from the Pentium calculations are shown here. The program copy of the table data is rounded to 1/32nd of a point and stored with a -2 point delta..

Computed average crib score when pone discards AA-KK (flushes ignored):

A	6.07							Average Crib Score For **PONE** Discards					
2	5.07	6.43											
3	5.17	7.34	6.78										
4	5.74	5.44	6.10	6.59									
5	6.06	6.17	6.85	7.46	**9.39**								
6	4.93	5.13	4.92	5.47	7.66	7.17							
7	4.95	5.12	5.16	4.91	7.08	6.64	7.25						
8	4.92	5.03	5.08	5.02	6.36	6.05	7.88	6.76					
9	4.66	4.82	4.82	4.75	6.22	6.31	5.46	5.97	6.44				
10	4.46	4.64	4.70	4.55	7.46	4.41	4.44	5.02	5.52	6.11			
J	4.72	4.91	4.97	4.80	7.75	4.61	4.73	4.65	4.98	5.60	6.56		
Q	4.41	4.60	4.66	4.49	7.42	4.29	4.44	4.38	4.14	4.65	5.55	5.89	
K	4.34	4.53	4.59	4.43	7.31	4.25	4.38	4.31	4.13	**3.99**	4.89	4.56	5.72
	A	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**J**	**Q**	**K**

Computed average crib score when dealer discards AA-KK (flushes ignored):

A	5.26							Average Crib Score For **DEALER** Discards					
2	4.18	5.67											
3	4.47	6.97	5.90										
4	5.45	4.51	4.88	5.65									
5	5.48	5.44	6.01	6.54	**8.95**								
6	3.80	3.87	3.72	3.87	6.65	5.74							
7	3.73	3.81	3.67	3.74	6.04	4.94	5.98						
8	3.70	3.58	3.84	3.84	5.49	4.70	6.58	5.42					
9	3.33	3.63	3.66	3.69	5.47	5.11	4.06	4.74	5.09				
10	3.37	3.51	3.61	3.62	6.68	3.15	3.10	3.86	4.27	4.73			
J	3.65	3.79	3.88	3.89	7.04	3.40	3.43	3.39	3.98	4.64	5.37		
Q	3.39	3.52	3.62	3.63	6.71	3.08	3.17	3.16	2.97	3.36	4.90	4.66	
K	3.42	3.55	3.66	3.67	6.70	3.13	3.21	3.20	3.05	**2.86**	4.07	3.50	4.62
	A	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**J**	**Q**	**K**

The discard routine is pretty good. The pegging routine needs work, since it still makes some bonehead plays, is too predictable, and pushes too often. But overall, I think the program will challenge even the experienced player. This was my original goal.

Credits:

Special thanks here go to my dad, who taught me the game long before computers were around to amuse and entertain us. Belated thanks go to Ted Blackney, Eddie Green, and Joe Wergin from the Madison, WI, area. About 40 years ago they dragged me off to the National Open in Raleigh, NC, where I duly served as cannon fodder. But this gave me a chance to watch the action in later rounds. That opened my eyes to the importance of positional play for the first time. Eddie finished second in the tournament. Ted and Joe didn't fare as well, but both were already renown Skat players. Joe later became president of the American Cribbage Congress. For the 1997 version, thanks go to Scott Numbers for indispensable advice on Windows API and DLL construction. Thanks also go to my brother Mark and co-workers Bern Knutsen and Steve Ducharme for helpful suggestions during beta testing. Mark endured a barrage of ZIP files e-mailed across country as I kept tinkering with things. For this 2019 version, thanks go especially to my wife Betsy for accomodating the time spent on this project. We are both retired and time is now our most precious commodity.

As of: 02 Jan 2019
Email: craig_hessel@hotmail.com
Web Site: http://www.geocities.ws/craig_hessel/

Appendix: Custom DLL

You may use the following sample C code as a starting point to create your custom **CRIBALT32.DLL** to handle decision-making chores for the program. This code has been compiled/linked as is in Microsoft Visual Studio C++ 6.0 for a workable demo.

```
//--------------------------------------------------------------------------
// CRIBALT32.C contains stubs for alternate decision routines for the Visual
// Basic 6.0 version of Cribbage 97.
//
// In MS Visual C++ 6.0 IDE, create new Dynamic Link-Library Project (empty).
// Add files CRIBALT32.C and CRIBALT32.DEF to project.  Build from there.
//
// ;CRIBALT32.DEF:
//    LIBRARY       CribAlt32
//    EXPORTS
//          Discard     @1
//          MakePegPlay @2
//
// CRIBCORE32.DLL has several other exports.  Your alternate decision routines
// are called only if Global/UseCribAltDLL entry of CRIBBAGE32.INI specifies:
//
// UseCribAltDLL=0 -- Ignore CRIBALT32.DLL (default)
//               =1 -- Use CRIBALT32 for player decisions only (hints/autoplay)
//               =2 -- Use CRIBALT32 for computer decisions only
//               =3 -- Use CRIBALT32 for all decisions
//
// These combinations of settings allow you to compare your routines with the
```

```
// default routines by running duplicate autoplay matches.  Two additional
// Global section entries may be helpful for testing/evaluation:
//
// AllowMultipleInstances=1 -- Allows multiple instances of program
// AllowCtrlF10=1           -- Enables Ctrl-F10 to toggle continuous play
//
// Continuous play halts after 1000 games, if not toggled off sooner.  The
// Autoplay/Autoscore buttons must both be enabled for Ctrl-F10 to work.  On
// my Surface Pro 4, it takes about 18.5 minutes for a 1000-game run with all
// decisions made by CRIBCORE32.DLL.
//
// It is also a good idea to select Alternate First Deals and Disable Sounds
// when automating multi-game runs to compare your decision routines to the
// native program routines.
//-----------------------------------------------------------------------------

#include <windows.h>
#include <ole2.h>

#define SuitOf(Card)  ((Card) & 0xF0)
#define RankOf(Card)  ((Card) & 0x0F)
#define CountOf(Rank) ((Rank) > 10 ? 10 : (Rank))

typedef struct tagHandStruct
  {LPSTR Hand;
   int   Len;
   int   Ndx;
  } HANDSTRUCT;

typedef HANDSTRUCT FAR* LPHANDSTRUCT;

//-----------------------------------------------------------------------------
// You can ignore args here and this DLL entry point as long as, e.g., no
// global memory is allocated for DLL instances.  Also avoid global variables,
// since another instance may overwrite your data.  Local data is stored on
// the caller's stack, so is specific to each instance.
//-----------------------------------------------------------------------------
BOOL WINAPI DllMain (
    HINSTANCE const instance,  // handle to DLL module
    DWORD     const reason,    // reason for calling function
    LPVOID    const reserved)  // reserved
  {
   return TRUE;
  }

//-----------------------------------------------------------------------------
// Call this or similar function if argument error detected in any of exported
// functions.  CRIBCORE32 does the same for its exports.  VB6 calling program
// does not check for error return.  It will continue sending good arguments
// so long as you continue returning good data.  This display is mainly for
// debugging.
//
// You can write your routines without further calls to Windows functions.
// This is helpful to know if your C experience is mainly in the warm and
// fuzzy DOS environment (as it was for me).  The CRIBCORE32 code does not
// call any other external functions at all.  E.g., no floating point
// functions are called, although that may change in the future.
//-----------------------------------------------------------------------------
int ArgErrMsg(LPSTR FuncName)
  {char Buf[48];

   wsprintf(Buf, "CRIBALT32.DLL: %s() argument error", FuncName);
   MessageBox(GetActiveWindow(), Buf, "Call Terminated", MB_ICONSTOP);
   return -1;
  }

//-----------------------------------------------------------------------------
// Copy cards to your own buffer (do not tamper with VB6 input strings), with
// preliminary error check.  You do not have to convert the card format, as is
// done here, but it was convenient in CRIBCORE32.
//-----------------------------------------------------------------------------
int CopyHand(LPSTR Tgt, LPSTR Src, int Length)
  {int n, m, Card, Rank, Suit;
```

```
   for (n = 0; n < Length; n++)
     {Card = Src[n];
      if (Card < 0 || Card > 51)                  return 1;
      for (m = 0; m < n; m++) if (Card == Src[m]) return 1;

      Rank = Card % 13 + 1;      // 1-13
      Suit = 16 << (Card / 13);  // 16, 32, 64, 128
      Tgt[n] = (char)(Rank | Suit);
     }

   return 0;
  }

//---------------------------------------------------------------------------
// Return 1-based index of first legal peg play, or 0 if none.
//---------------------------------------------------------------------------
int GetFirstPlay(LPSTR Hand, int Length, int Start, int PegCnt)
  {int n;

   for (n = Start; n < Length; n++)
      if (PegCnt + CountOf(RankOf(Hand[n])) <= 31) return n + 1;

   return 0;
  }

//---------------------------------------------------------------------------
// Check for legal pegging reconstruction and return peg count (-1 if error).
//---------------------------------------------------------------------------
int ChkPeg(LPSTR Cards, int Length, int CardsLeft, int OwnTurn, int IsRestart)
  {int n, PegCnt = 0, GoCnt = 0; HANDSTRUCT Own, Opp; LPHANDSTRUCT Cur;

   Own.Hand = Cards;   Own.Len = 4-CardsLeft; Own.Ndx = 0;
   Opp.Hand = Cards+7; Opp.Len = Length-7;    Opp.Ndx = 0;

   // Replay pegging to own turn, aborting on any inconsistency.

   while (Own.Ndx < Own.Len || Opp.Ndx < Opp.Len || !OwnTurn)
     {Cur = OwnTurn ? &Own : &Opp;

      if (Cur->Ndx >= Cur->Len) n = 32;
      else n = PegCnt + CountOf(RankOf(Cur->Hand[Cur->Ndx]));

      if (n > 31)

         // Insure 'go' not mistake, i.e., check if later card is
         // playable -- unknown opponent cards are at worst face cards.

        {n = OwnTurn ? 4 : Cur->Len;
         if (GetFirstPlay(Cur->Hand, n, Cur->Ndx, PegCnt)) return -1;
         if (!OwnTurn && Cur->Len < 4 && PegCnt <= 21)     return -1;
         ++GoCnt;
        }
      else
        {PegCnt = n; GoCnt = 0; Cur->Ndx++;
        }

      if (PegCnt == 31 || GoCnt > 1) PegCnt = GoCnt = 0;

      OwnTurn = !OwnTurn;
     }

   // If restarting and peg count non-zero, insure legal restart.

   if (IsRestart && PegCnt)
     {if (GoCnt && Opp.Ndx < Opp.Len)                return -1;
      if (GetFirstPlay(Own.Hand, 4, Own.Ndx, PegCnt)) return -1;
     }

   return IsRestart ? 0 : PegCnt;
  }

//---------------------- EXPORTS: MODIFY THESE FUNCTIONS ----------------------
```

```
//---------------------------------------------------------------------------
// Discard decision function.
//
// Input:
//    Hand  -- Exactly 6 cards (0-51 each).
//    DlScr -- Dealer game score (0-120).
//    PnScr -- Pone game score (0-120).
//
// Output:
//    On argument error, return -1.  Otherwise return discard indexes as if
//    dealer in high byte nibbles and discard indexes as if pone in low byte
//    nibbles.  Indexes are 0-5, with nibble order irrelevant.
//
// Game scores here and in next function let you include endgame strategy,
// positional adjustments, etc., in your decision.  Match score and opponent
// tendencies from prior hands are not available, however.  The corresponding
// CRIBCORE32 exports have same names and arguments.
//
// Both exports by CRIBCORE32 are deterministic.  That is, the same input always
// produces the same output.  This is necessary for duplicate play.  Your
// routines should behave similarly.
//
// If you need some randomness to break decision ties, rely on the order of
// the input cards rather than, e.g., a clock function call.  The VB6 calling
// program passes along the input hand here and the unplayed input cards in
// the next function in the same "random" order as the cards were dealt (even
// if the VB6 display of those cards is sorted).
//---------------------------------------------------------------------------
short _stdcall Discard(LPSTR Hand, WORD DlScr, WORD PnScr)
  {char Hnd[6]; int DlPicks, PnPicks;

   // Convert hand to useful format, with preliminary error check.

   if (DlScr > 120 || PnScr > 120 || CopyHand(Hnd, Hand, 6))
      return (short)ArgErrMsg("Discard");

   // SUBSTITUTE YOUR CODE HERE -- Sample code shows pone discard as first two
   // cards and dealer discard as last two cards.  Reference Hnd[] from now
   // on, not Hand[].

   PnPicks = 0x01;
   DlPicks = 0x45;

   return (short)(DlPicks << 8 | PnPicks);
  }

//---------------------------------------------------------------------------
// Peg play decision function.
//
// Input:
//   Cards     -- Own played cards in order, then own cards left, then own
//                discards, then cut, then opponent played cards in order.
//   Length    -- Number of cards (7-11), implying opponent cards left.
//   OwnLeft   -- Number of own cards left to play (0-4).
//   IsDealer  -- True (non-zero) if own cards (leftmost) are dealer cards.
//   IsRestart -- True (non-zero) if current peg count is zero.
//   OwnScr    -- Own game score (0-120).
//   OppScr    -- Opponent game score (0-120).
//
// Output:
//   On argument error, return -1.  Otherwise, return 0 for go or index 1-4 of
//   of card play.  Index is 1-based from start of Cards.
//
// The IsRestart argument is needed to distinguish go play from card play that
// follows count reset.  The VB6 program currently pre-checks for go, so does
// not bother to call function in this case.  But include go check anyway.
//
// You will probably want to create a function like ChkPeg() to reconstruct
// the played card sequence.  The card sequence and peg count are implied.
//---------------------------------------------------------------------------
short _stdcall MakePegPlay(LPSTR Cards, WORD Length, WORD OwnLeft,
               short IsDealer, short IsRestart, WORD OwnScr, WORD OppScr)
```

```
  {char Cds[11]; int PegCnt;

   // Convert cards to useful format, with preliminary error check.

   if (OwnScr > 120 || OppScr > 120 || Length < 7 || Length > 11 ||
       OwnLeft > 4 || CopyHand(Cds, Cards, Length) ||
       (PegCnt = ChkPeg(Cds, Length, OwnLeft, !IsDealer, IsRestart)) == -1)
      return (short)ArgErrMsg("MakePegPlay");

   // SUBSTITUTE YOUR CODE HERE -- Sample code finds first legal play.
   // Reference Cds[] from now on, not Cards[].

   return (short)GetFirstPlay(Cds, 4, 4 - OwnLeft, PegCnt);
  }

// End CRIBALT32.C
```