Il file "`mani_poker.rdata`" è un *workspace* di **R** contenente i seguenti oggetti:

`mani`	Si tratta di una matrice le cui righe contengono le 2598960 possibili mani del poker. Questa matrice ha 5 colonne, corrispondenti alle 5 carte "estratte". La codifica delle carte è numerica; vale a dire che a ciascuna delle 52 carte del mazzo è stato attribuito un codice numerico (da 1 a 52). Cominciando con le carte di cuori, all'asso è stato attribuito il valore 1, alle 9 carte successive i valori da 2 a 10, al fante il valore 11, alla donna il valore 12 e al re il valore 13. Seguendo lo stesso criterio, alle carte di quadri sono stati attribuiti i codici da 14 a 26, alle carte di picche sono stati attribuiti i codici da 27 a 39, alle carte di fiori sono stati attribuiti i codici da 40 a 52. Data una mano di 5 carte che utilizza la codifica appena descritta, è possibile visualizzare le carte stesse sotto forma di *punto* e di *seme* impiegando la funzione **`mano`**.
`mano`	È una semplice funzione (una sola riga) che accetta come input un vettore di 5 elementi numerici (da 1 a 52) e l'oggetto di **R** **`mazzo`** (vedi sotto) e restituisce, sotto forma di matrice con 2 righe e 5 colonne, i corrispondenti elementi sotto forma di *punto* e di *seme*.
`mazzo`	È una matrice di caratteri alfanumerici (`character`) con 52 righe (le 52 carte) e 2 colonne; la prima colonna contiene il *punto*, mentre la seconda contiene il *seme*. È il secondo argomento da passare a `mano` per ottenere la decodifica.

Per eseguire gli script che seguono è necessario aver scaricato il file `mani_poker.rdata` e averlo letto eseguendo l'istruzione `load("poker.rdata")` dopo aver *settato* opportunamente la *woking directory* di **R**. Verificate di aver eseguito correttamente le istruzioni precedenti, chiedendo cosa avete nel *workspace* di **R**.

```
> ls()
[1] "mani"  "mano"  "mazzo"
```

Se avete ottenuto l'*output* precedente, potete procedere con gli esempi che seguono.

```
> mano(c(1,14,27,40),mazzo)
     punto  seme
[1,] "Asso" "Cuori"
[2,] "Asso" "Quadri"
[3,] "Asso" "Picche"
[4,] "Asso" "Fiori"
```

```
> mano(c(1:4)*13,mazzo)
     punto seme
[1,] "Re"  "Cuori"
[2,] "Re"  "Quadri"
[3,] "Re"  "Picche"
[4,] "Re"  "Fiori"
```

```
> # scegliamo una mano a caso
> out <- sample(c(1:nrow(mani)),1,replace=FALSE)
> out                 # questo e' l'ID della mano
[1] 584571

> mani[out,]          # questi sono gli ID delle carte
[1]  3 11 18 30 32

> x <- mani[out,]; mano(x,mazzo)
     punto     seme
[1,] "Tre"     "Cuori"
[2,] "Fante"   "Cuori"
[3,] "Cinque"  "Quadri"
[4,] "Quattro" "Picche"
[5,] "Sei"     "Picche"
```

Contare quante sono le mani che corrispondono a determinate caratteristiche (per esempio, le mani con un *poker*) non è difficile in linea di principio, ma può essere abbastanza lungo, soprattutto se si dispone di un *computer* non molto recente. Conoscere le regole del calcolo combinatorio consente, invece, di ottenere una risposta esatta in un tempo molto inferiore. Se comunque volete provare ad enumerare le diverse mani con **R**, potremmo sfruttare la funzione `apply`, applicando una funzione `fun` appositamente scritta per il problema in oggetto. Lo *script* che segue conta il numero di mani con un *poker*.

```
fun <- function(x)
{
    y <- x %% 13
    tbl <- table(y)
    max(tbl)==4
}
```

La funzione `fun` accetta come *input* il vettore che contiene la mano in parola, codificata con 5 numeri da 1 a 52. La prima riga, sfruttando il resto della divisione per 13, estrae il valore del *punto* (il re ha valore zero, invece che 13); la seconda crea la distribuzione di frequenza dei punti della mano; l'ultima riga restituisce `TRUE` se la *moda* della distribuzione è 4 (*poker*) e `FALSE` in tutti gli altri casi. Notate che scrittura che segue è del tutto equivalente alla precedente, anche se forse un po' meno leggibile: `fun <- function(x) max(table(x %% 13))==4`.
Si tratta ora di "applicare" questa funzione a tutte le righe di `mani` e poi di contare quanti sono i valori `TRUE`:

```
ris <- apply(mani,1,fun)
sum(ris)
```

Prima di lanciare tale richiesta, riteniamo sia opportuno avere una stima del tempo che impiegherà **R** a soddisfarla; a tale scopo potremmo vedere quanto **R** impiega con un numero più limitato di mani:

```
tmp <- mani[c(1:10000),]
system.time(ris <- apply(tmp,1,fun))
   user  system elapsed
  31.99    0.03   32.59
```

Su un *computer* un po' "vecchio" **R** impiega circa 33 secondi per analizzare 10000 mani; poiché le mani totali sono 2598960, **R** dovrebbe impiegare circa 8578 secondi, vale dire oltre due ore e 20 minuti! È certamente più rapido usare le regole del calcolo combinatorio.

Modificando opportunamente la funzione `fun` è possibile contare le mani relative ad altri problemi. Ecco alcuni esempi:

Una mano con tutte carte diverse	`fun <- function(x) max(table(x %% 13))==1`
Una mano con un tris	`fun <- function(x) max(table(x %% 13))==3 & min(table(x %% 13))==1`
Una mano con un full	`fun <- function(x) max(table(x %% 13))==3 & min(table(x %% 13))==2`
Una coppia	`fun <- function(x) max(table(x %% 13))==2 & length(table(x %% 13))==4`
Una doppia coppia	`fun <- function(x) max(table(x %% 13))==2 & length(table(x %% 13))==3`

Ovviamente, potremmo far sì che `fun` valuti contemporaneamente tutte queste possibilità, restituendo come risultato una stringa. L'esecuzione sarà più lunga, ma potremo enumerare tutti i casi che ci interessano una volta per tutte:

```
fun <- function(x)
{
    y <- x %% 13
    tbl <- table(y)
    out <- "Boh"
    if (max(tbl)==4) out <- "Poker"
    if (max(tbl)==1) out <- "Tutte diverse"
    if (max(tbl)==3 & min(tbl)==2) out <- "Full"
    if (max(tbl)==3 & min(tbl)==1) out <- "Tris"
    if (max(tbl)==2 & length(tbl)==4) out <- "Coppia"
    if (max(tbl)==2 & length(tbl)==3) out <- "Doppia coppia"
    out
}

ris <- apply(mani,1,fun)
frq(ris)
```

L'*output* che abbiamo ottenuto è il seguente:

```
              x         n              f
+---------------+---------+--------------+
| Coppia        | 1098240 |  42.25690276 |
| Doppia coppia |  123552 |   4.75390156 |
| Full          |    3744 |   0.14405762 |
| Poker         |     624 |   0.02400960 |
| Tris          |   54912 |   2.11284514 |
| Tutte diverse | 1317888 |  50.70828331 |
+---------------+---------+--------------+
                  2598960   100.00000000
Osservazioni mancanti: 0
```

Come si vede, abbiano abbiamo classificato in modo esaustivo le 2598960 mani del poker. Più di metà delle volte le carte sono tutte diverse fra loro. Una coppia ha una probabilità un po' superiore al 40%, una doppia coppia capita un po' meno di 5 volte su 100, mentre il tris poco più di 2 volte su 100. Un *full* capita fra 1 e volte su 1000, mentre il *poker* fra 2 e 3 volte su 10000. Ovviamente questi risultati sono relativi alla prima mano e non contemplano quanto può accadere scartando una o più carte.
In modo analogo possiamo rispondere a domande riguardanti il *seme* delle carte. In queste caso, per una decodifica possiamo usare il risultato della "divisione intera" (mentre nel caso precedente avevamo sfruttato il resto della divisione). Ecco la funzione.

```
fun <- function(x)
{
    y <- ((x-1) %/% 13) + 1
    tbl <- table(y)
    if (length(tbl)==1) out <- "Tutti colori uguali"
    if (length(tbl)==2) out <- "Due colori"
    if (length(tbl)==3) out <- "Tre colori"
    if (length(tbl)==4) out <- "Quattro colori"
    out
}

ris <- apply(mani,1,fun)
frq(ris)
```

Ed ecco i risultati ottenuti.

```
                    x         n             f
+-------------------+---------+-------------+
| Due colori        |  379236 |  14.5918367 |
| Quattro colori    |  685464 |  26.3745498 |
| Tre colori        | 1529112 |  58.8355342 |
| Tutti colori uguali |   5148 |   0.1980792 |
+-------------------+---------+-------------+
                      2598960   100.0000000
Osservazioni mancanti: 0
```

Il *workspace* di **R** è a vostra disposizione per ulteriori "esperimenti".