Math / Stat 341 : Fall 2015 : sjm1@williams.edu

```
(* Computing a 5-0 trump split among two hands *)
deck = {}; (* initialize deck to empty *)
 (* assign five 1s to the deck; the 1s represent the trump suit *)
(* then we assign 21 0s, these are the non-trump *)
(* taking time and coding well can save you a LOT of trouble *)
For[n = 1, n ≤ 5, n++, deck = AppendTo[deck, 1]];
For[n = 6, n ≤ 26, n++, deck = AppendTo[deck, 0]];
Length[deck] (* makes sure got 26 cards *)
(* should have this in the program so we make sure we use the right deck,
and thus will paste it below! *)
```

Out[8]= 26

```
In[13]:= trumpsplit[numdo_] := Module[{},
    count = 0;
    deck = {}; (* initialize deck to empty *)
    For[n = 1, n ≤ 5, n++, deck = AppendTo[deck, 1]];
    For[n = 6, n ≤ 26, n++, deck = AppendTo[deck, 0]];
    For[n = 1, n ≤ numdo, n++,  (* main loop of code *)
     {
      hand = RandomSample[deck, 13]; (* randoml1y choose 13 cards *)
      numtrump = Sum[hand[[k]], {k, 1, 13}];
      (* note numtrump is 0 or 5 if we have a 5-0 split *)
      If[numtrump == 0 || numtrump == 5, count = count + 1];
      (* count is our counter, counts how often have 5-0 *)
      (* we use || for or;
      would use && for and use two equal signs for comparison*)
     }]; (* end of n loop *)
    Print["Two theories: 2(1/2)^5 gave ", 6.25, "%, other gave 3.913%."];
    Print["We observe ", 100. count / numdo, "."];
   ];

In[10]:= Timing[trumpsplit[1 000 000]]
```

Two theories: 2(1/2)^5 gave 6.25%, other gave 3.9%.

We observe 3.9166.

Out[10]= {11.2945, Null}

```
In[20]:= (* Getting exactly two kings *)
twokings[numdo_] := Module[{},
   deck = {}; (* initialize deck to empty *)
   (* 1 is a king, 0 non-king *)
   For[n = 1, n ≤ 4, n++, deck = AppendTo[deck, 1]];
   For[n = 5, n ≤ 52, n++, deck = AppendTo[deck, 0]];
   count = 0; (* initialize num of successes to 0 *)
   For[n  = 1, n ≤ numdo, n++,
    {
     hand = RandomSample[deck, 5]; (* 5 card hand *)
     numkings = Sum[hand[[k]], {k, 1, 5}];
     If[numkings == 2, count = count + 1];
    }]; (* end of n loop *)
   Print["Theory predicts prob exactly two kings is ",
    100.0 Binomial[4, 2] Binomial[48, 3] / Binomial[52, 5], "."];
   Print["Observed probability is ", 100.0 count / numdo, "."];
  ];

In[22]:= Timing[twokings[1 000 000]]

Theory predicts prob exactly two kings is 3.99298.

Observed probability is 3.9965.

Out[22]= {6.94204, Null}

In[19]:= Length[deck]

Out[19]= 52
```

In[28]:=
```
(* calculating probability of a full house, queens and kings *)
(* probability is VERY small so must do a lot of simulations! *)
(* sadly the more you want to compute, the worse Mathematica is *)
(* this is not a hard code, don't really need the special fns here *)
(* would want to shift to another language that is better *)
fullkingqueens[numdo_] := Module[{},
   deck = {}; (* initialize deck to empty *)
   (* 10 is a queen, 1 is a king, 0 non-king *)
   For[n = 1, n ≤ 4, n++, deck = AppendTo[deck, 1]];
   For[n = 5, n ≤ 8, n++, deck = AppendTo[deck, 10]];
   For[n = 9, n ≤ 52, n++, deck = AppendTo[deck, 0]];
   count = 0; (* initialize num of successes to 0 *)
   For[n  = 1, n ≤ numdo, n++,
    {
     hand = RandomSample[deck, 5]; (* 5 card hand *)
     numkings = Sum[hand[[k]], {k, 1, 5}];
     (* want full house of Qs and Ks *)
     (* sum is either 23 or 32! *)
     If[numkings == 32 || numkings == 23, count = count + 1];
    }]; (* end of n loop *)
   Print["Theory predicts prob full house (Qs and Ks) is ",
    100.0 Binomial[2, 1] Binomial[4, 3] Binomial[4, 2] / Binomial[52, 5], "."];
   Print["Observed probability is ", 100.0 count / numdo, "."];
  ];
```

In[30]:=
```
Timing[fullkingqueens[10 000 000]]
```

```
Theory predicts prob full house (Qs and Ks) is 0.00184689.
Observed probability is 0.00168.
```

Out[30]=
```
{71.9165, Null}
```

In[31]:=
```
Timing[fullkingqueens[40 000 000]]
```

```
Theory predicts prob full house (Qs and Ks) is 0.00184689.
Observed probability is 0.0018925.
```

Out[31]=
```
{298.945, Null}
```