I stimulated the situation using a geometric distribution for N, the number of flips of the second coin, and using a binomial distribution for Y, the number of heads the second coin will flip. For a geometric variable N, E[N] is 1/p, and for a binomial E[Y] is np or (1/p)q or (q/p). Then, for a geometric variable Var(Y) will be np(1-p) or (q/p)(1-q). This was stimulated in C++ with the output and code below:

```
mean
 q: 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
p -----------------------------------------------------------------------
0.1000 |    1.0000  2.0000  3.0000  4.0000  5.0000  6.0000  7.0000  8.0000  9.0000
0.2000 |    0.5000  1.0000  1.5000  2.0000  2.5000  3.0000  3.5000  4.0000  4.5000
0.3000 |    0.3333  0.6667  1.0000  1.3333  1.6667  2.0000  2.3333  2.6667  3.0000
0.4000 |    0.2500  0.5000  0.7500  1.0000  1.2500  1.5000  1.7500  2.0000  2.2500
0.5000 |    0.2000  0.4000  0.6000  0.8000  1.0000  1.2000  1.4000  1.6000  1.8000
0.6000 |    0.1667  0.3333  0.5000  0.6667  0.8333  1.0000  1.1667  1.3333  1.5000
0.7000 |    0.1429  0.2857  0.4286  0.5714  0.7143  0.8571  1.0000  1.1429  1.2857
0.8000 |    0.1250  0.2500  0.3750  0.5000  0.6250  0.7500  0.8750  1.0000  1.1250
0.9000 |    0.1111  0.2222  0.3333  0.4444  0.5556  0.6667  0.7778  0.8889  1.0000

variance
 q: 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
p -----------------------------------------------------------------------
0.1000 |    0.9000  1.6000  2.1000  2.4000  2.5000  2.4000  2.1000  1.6000  0.9000
0.2000 |    0.4500  0.8000  1.0500  1.2000  1.2500  1.2000  1.0500  0.8000  0.4500
0.3000 |    0.3000  0.5333  0.7000  0.8000  0.8333  0.8000  0.7000  0.5333  0.3000
0.4000 |    0.2250  0.4000  0.5250  0.6000  0.6250  0.6000  0.5250  0.4000  0.2250
0.5000 |    0.1800  0.3200  0.4200  0.4800  0.5000  0.4800  0.4200  0.3200  0.1800
0.6000 |    0.1500  0.2667  0.3500  0.4000  0.4167  0.4000  0.3500  0.2667  0.1500
0.7000 |    0.1286  0.2286  0.3000  0.3429  0.3571  0.3429  0.3000  0.2286  0.1286
0.8000 |    0.1125  0.2000  0.2625  0.3000  0.3125  0.3000  0.2625  0.2000  0.1125
0.9000 |    0.1000  0.1778  0.2333  0.2667  0.2778  0.2667  0.2333  0.1778  0.1000
```

```
#include <iostream>
#include <iomanip>

void table (){
  float var[9][9] =
{{0,0,0,0,0,0,0,0,0},{0,0,0,0,0,0,0,0,0,0},{0,0,0,0,0,0,0,0,0,0},{0,0,0,0,0,0,0,0,0,0},{0,0,0,0,0,0,0,0,0,0},{0,
0,0,0,0,0,0,0,0,0},{0,0,0,0,0,0,0,0,0,0},{0,0,0,0,0,0,0,0,0,0},{0,0,0,0,0,0,0,0,0,0}};
  std::cout << std::fixed << std::setprecision(4) << "mean\n
q:\t0.1\t0.2\t0.3\t0.4\t0.5\t0.6\t0.7\t0.8\t0.9\np
-----------------------------------------------------------------------\n";
  for (int c = 0; c<9; c++) {
    std::cout << static_cast<float>(c+1)/10.0 << " |\t";
    for (int b = 0; b<9; b++){
      var[c][b] =
(static_cast<float>(b+1)/10.0)*(1.0-(static_cast<float>(b+1)/10.0))/(static_cast<float>(c+1)/10.0);
      std::cout << (static_cast<float>(b+1)/10.0)/(static_cast<float>(c+1)/10.0) << "\t";
```

```
    }
    std::cout << "\n";
  }
  std::cout <<"\nvariance\n q:\t0.1\t0.2\t0.3\t0.4\t0.5\t0.6\t0.7\t0.8\t0.9\np
-----------------------------------------------------------------------\n";
  for (int i = 0; i<9; i++){
    std::cout << static_cast<float>(i+1)/10.0 << " |\t";
    for (int a = 0; a<9; a++){
      std::cout << var[i][a] << "\t";
    }
    std::cout << "\n";
  }
}
int main() {
  table();
  return 0;
}
```