In [1]:
```
import Pkg
Pkg.add("DataStructures")
Pkg.add("Shuffle")
Pkg.add("Plots")
Pkg.add("Distributions")
Pkg.add("Random")
using DataStructures
using Shuffle
using Plots
using Random
using Distributions


#takes in an integer n
#returns array of length n of integers +/- 1 (randomly chosen) that represent s
function initial_config(n::Int)
  config = zeros(n)
  if n%2 != 0
    println("Size n must be even.")
    return
  else
    for i=1:n
      config[i] = rand([-1, 1])
    end
  end
  #print(config)
  return config
  #essentially from N, we get a randomized array of spin ups and downs
  #e.g., N=2 may equal [1,-1]; N=4 may equal [1,-1,-1,1] as our initial configu
end

function gaussian_rf(N)
  nd = Normal(0, 1)
  return rand(nd, N)
end

function unit_rf(N)
  field = zeros(N)
  for i=1:N
    field[i] = rand([-.5, .5])
  end
  return field
end


#standard function for getting the hamilitonian of 1/r^2
#Ising Model
function get_energy(s::AbstractArray, h::AbstractArray, J)
  E0 = 0.0
  E1 = 0.0
  E2 = 0.0
  for i=1:length(s)
    #if i != length(s)
      #E0 += J*s[i]*s[i+1]
    #else
      #E0 += J*s[i]*s[1]
    #end
    for j=i:length(s)
      if j != i
        E1 += J*(s[i]-s[j])^2/(i-j)^2
      end
    end
    E2 += h[i]*s[i]
  end
  E = -E1/2 - E2
  return E
end
```