```python
# -*- coding: utf-8 -*-
"""
Created on Sat Nov 16 16:12:36 2019

@author: Alexm
"""
#PART 2: VARIATION OF THE MAGNETIC FIELD WITH CONSTANT J=1
from __future__ import division
import numpy as np
from numpy.random import rand
import matplotlib.pyplot as plt

print('Alexandra Mulholland 17336557')
J=-1
T=1
print('Part 2: varying the external magnetic field for constant J equal to %d'%J)
print('T=%d'%T)
nt=1
nbs=60
N=10
stepsequil=1000
stepsmoncar=1000
Bfield=np.linspace(0.0,25.0,nbs)

#we now define temperature as a constant value rather than a range of values
#we can change this value to test the effect it has on the magnetic field
#The magnetic field is expected to increase the value of the Curie or Neel temp
#as it keeps the system in equilibrium for longer
E= np.zeros(nt)
M= np.zeros(nt)
C= np.zeros(nt)
X = np.zeros(nt)
T=1

E,M,C,X = np.zeros(nbs), np.zeros(nbs), np.zeros(nbs), np.zeros(nbs)

n1= 1.0/(stepsmoncar*N*N)
n2=1.0/(stepsmoncar*stepsmoncar*N*N)
#again, used later to find X and C

def randomstate(N):
    state = 2*np.random.randint(2, size=(N,N))-1
    return state
#now the magf is not zero, but a range of values
J=-1
#same monte carlo code as explained in part 1 code
def moncar(config, beta, magf):
    for i in range(N):
        for j in range(N):
                a = np.random.randint(0, N)
                b = np.random.randint(0, N)
                s =  config[a, b]
                spinmag=s*magf
                naybor = config[(a+1)%N,b] + config[a,(b+1)%N] + config[(a-1)%N,b] + config[a,(b-1)%N]
                EC = 2*J*s*naybor+2*spinmag
                if EC < 0:
                    s *= -1
                elif rand() < np.exp(-EC*beta):
                    s *= -1
                config[a, b] = s
    return config

#now factoring in the effect of an external magnetic field into the energy
def TotEnergy(config, magf):
    energy = 0
    for i in range(len(config)):
        for j in range(len(config)):
            S = config[i,j]
            spinmag =S*magf
            naybor = config[(i+1)%N, j] + config[i,(j+1)%N] + config[(i-1)%N, j] + config[i,(j-1)%N]
            energy += -naybor*J*S/4 -spinmag
    return energy
```

```


def TotSpin(config, MAGFIELD):
    mag = np.sum(config)
    return mag

for bt in range(nbs):
    avE = avM = avE2 = avM2 = 0
    config = randomstate(N)
    BT=1.0/T
    BT2=BT*BT
    MAGFIELD=1.0*Bfield[bt] #mirroring the technique used for the temp

    for i in range(stepsequil):
        moncar(config, BT, bt)

    for i in range(stepsmoncar):
        moncar(config, BT, MAGFIELD)
        Ene = TotEnergy(config, MAGFIELD)
        Mag = TotSpin(config, MAGFIELD)

    avE = avE + Ene
    avM = avM + Mag
    avM2 = avM2 + Mag*Mag
    avE2 = avE2 + Ene*Ene

    E[bt] = n1*avE   #finding the average energy over the magnetic field range
    M[bt] = n1*avM   #average magnetisation over mf range
    C[bt] = (n1*avE2 - n2*avE*avE)*BT2 #heat capacity over mf range
    X[bt] = (n1*avM2 - n2*avM*avM)*BT  #susceptibility over mf range



plt.figure(1)
plt.plot(Bfield, E,'p', color='k',
    markersize=10,
    markerfacecolor='white',markeredgecolor='k',
    markeredgewidth=1)
plt.suptitle('Plot of the average energy versus magnetic field')
plt.title('T=1K and J=%d'%J, fontsize=10)
plt.xlabel("Magnetic Field, B / T", fontsize=12);
plt.ylabel("Energy/J ", fontsize=12); plt.axis('tight');



plt.figure(2)
#plotting the absolute value of the magnetisation as this can be negative (1 or -1)
plt.plot(Bfield, 1000*M,'-p', color='grey',
    markersize=10,
    markerfacecolor='white',markeredgecolor='grey',
    markeredgewidth=1)
plt.suptitle('Plot of the average magnetisation versus magnetic field')
plt.title('T=1K and J=%d'%J, fontsize=10)
plt.xlabel("Magnetic Field, B / T", fontsize=12);
plt.ylabel("Magnetisation / Am^-1", fontsize=12);


#specific heat
plt.figure(3)
plt.plot(Bfield, C,'p', color='c',
    markersize=10,
    markerfacecolor='white',markeredgecolor='c',
    markeredgewidth=1)
plt.suptitle('Plot of the specific heat versus magnetic field', fontsize=13)
plt.title('T=1K and J=%d'%J, fontsize=10)
plt.xlabel("Magnetic Field, B / T", fontsize=12);
plt.ylabel("Heat capacity C / JK^-1 ", fontsize=12); plt.axis('tight');


#suscpetibility- measure of the error in the magnetisation
plt.figure(4)
plt.plot(Bfield, X, 'p',color='blue',
```

```
    markersize=10,
    markerfacecolor='white',markeredgecolor='blue',
    markeredgewidth=1)
plt.suptitle('Plot of the susceptibility versus magnetic field', fontsize=13)
plt.title('T=1K and J=%d'%J, fontsize=10)
plt.xlabel("Magnetic Field, B / T", fontsize=12);
plt.ylabel("susceptibility ", fontsize=12); plt.axis('tight');
```