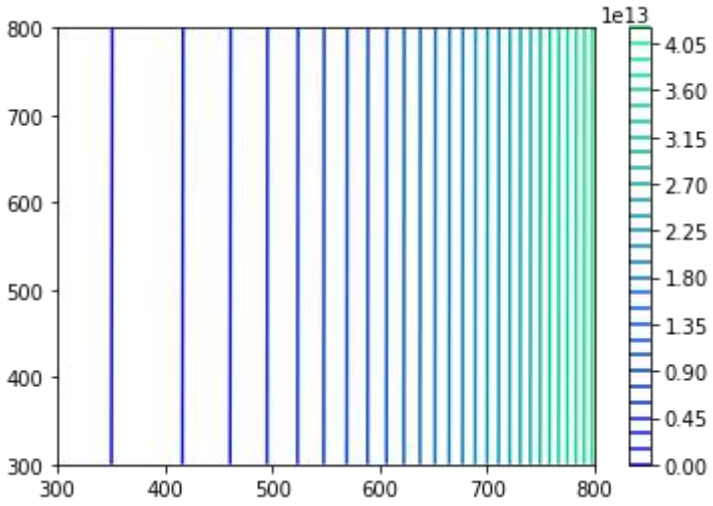
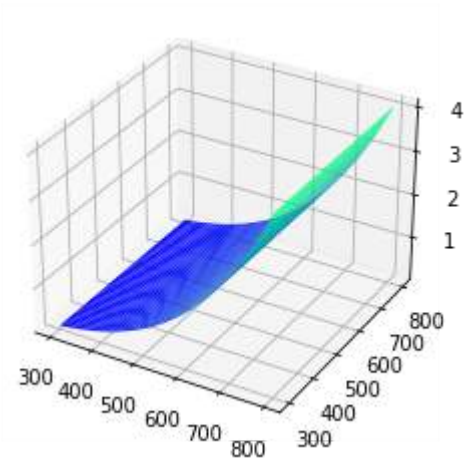
In [1]:
```
# # countour plot
import numpy as np
import matplotlib.pyplot as plt
def g(x,y):
    return 100*((x**2 - y)**2) + (1-x)**2
x = np.linspace(300,800,50)
y = np.linspace(300,800,50)
x,y = np.meshgrid(x,y)
z= g(x,y)
plt.contour(x,y,z,30,cmap='winter')
plt.colorbar()
plt.show()
```



In [2]:
```
# # mesh plot
from mpl_toolkits import mplot3d
fig = plt.figure()
ax = plt.axes(projection="3d")
ax.plot_surface(x,y,z, rstride=1, cstride=1,cmap='winter', edgecolor='none')
plt.show()
```

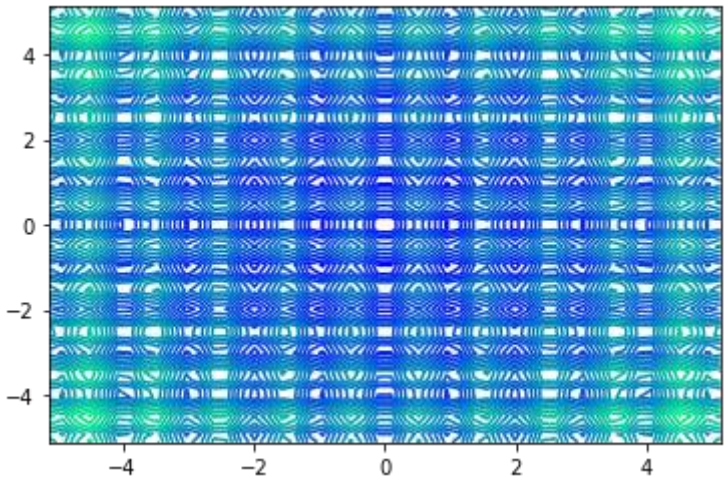


In [3]:
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
from mpl_toolkits.mplot3d import Axes3D
def g(x,y):
    return 20 +x**2-10*np.cos(2*3.14*x)+y**2-10*np.cos(2*3.14*y)
x= np.linspace(-5.12,5.12,50)
y= np.linspace(-5.12,5.12,50)
t,v= np.meshgrid(x,y)
```

```
z= g(t,v)
plt.contour(x,y,z,20,cmap='winter')
```

Out[3]:

```
<matplotlib.contour.QuadContourSet at 0x294563ce3d0>
```



In [4]:

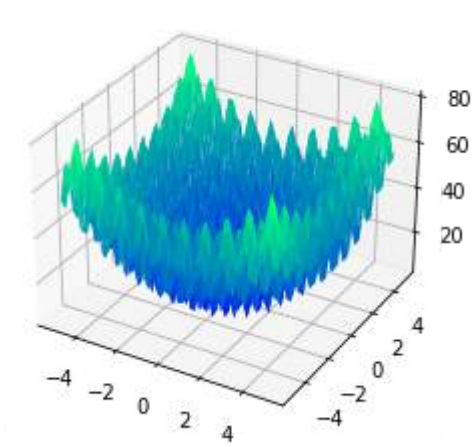
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
from mpl_toolkits.mplot3d import Axes3D

X = np.linspace(-5.12, 5.12, 50)
Y = np.linspace(-5.12, 5.12, 50)
X,Y = np.meshgrid(X, Y)

Z = (X**2 - 10 * np.cos(2 * np.pi * X)) + (Y**2 - 10 * np.cos(2 * np.pi * Y)) + 20

fig = plt.figure()
ax = fig.gca(projection='3d')
ax.plot_surface(X, Y, Z, rstride=1, cstride=1,cmap='winter',
   linewidth=0.08,
  antialiased=True,)
plt.show()
```

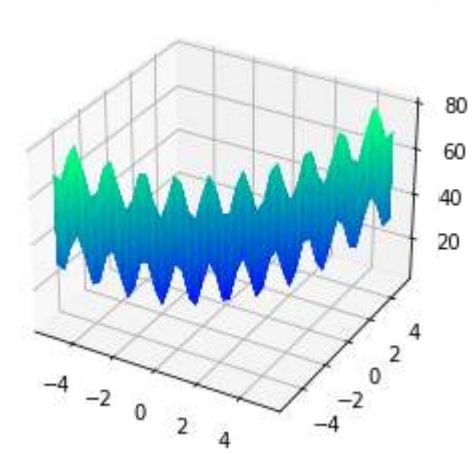
```
C:\Users\91953\AppData\Local\Temp\ipykernel_5996\3024530405.py:13: MatplotlibDepre
cationWarning: Calling gca() with keyword arguments was deprecated in Matplotlib
3.4. Starting two minor releases later, gca() will take no keyword arguments. The
gca() function should only be used to get the current axes, or if no axes exist, c
reate new axes with default keyword arguments. To create a new axes with non-defau
lt arguments, use plt.axes() or plt.subplot().
  ax = fig.gca(projection='3d')
```



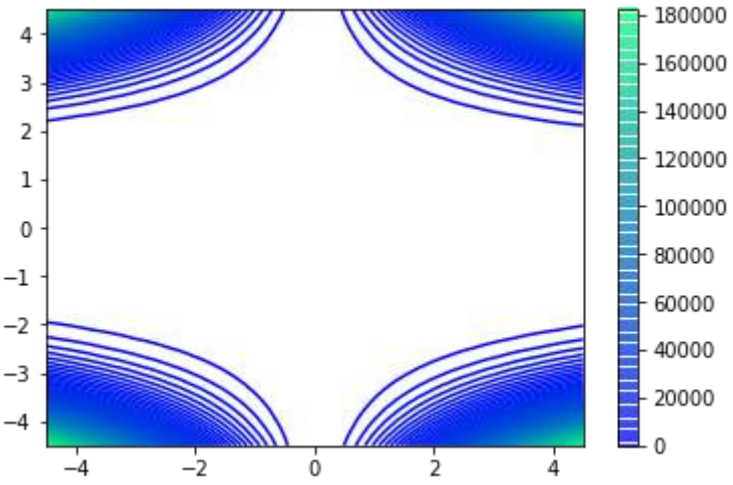
In [5]:

```
# mesh
from mpl_toolkits import mplot3d
```

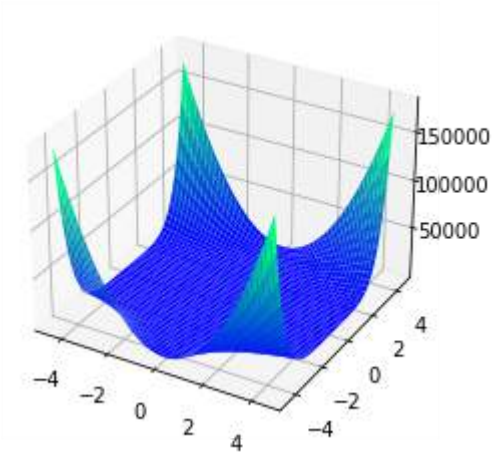
```
fig = plt.figure()
ax=plt.axes(projection='3d')
ax.plot_surface(x,y,z, rstride=1, cstride=1,cmap='winter', edgecolor='none')
plt.show()
```



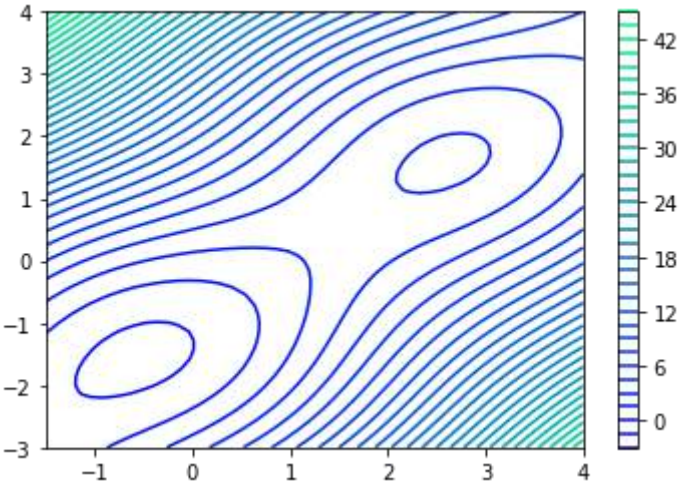
In [6]:
```
def g(x,y):
    return (1.5 -x +x*y)**2 + (2.25 - x + x*y**2)**2 +(2.625-x+x*y**3)**2
x= np.linspace(-4.5,4.5,50)
y= np.linspace(-4.5,4.5,50)
x,y = np.meshgrid(x,y)
z = g(x,y)
plt.contour(x,y,z,100,cmap='winter')
plt.colorbar()
plt.show()
```



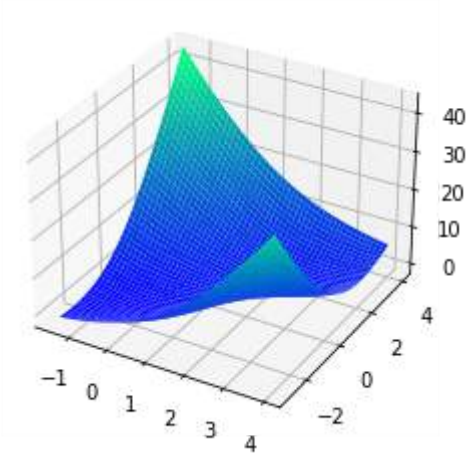
In [7]:
```
# mesh
fig = plt.figure()
ax= plt.axes(projection="3d")
ax.plot_surface(x,y,z, rstride=1, cstride=1,cmap='winter', edgecolor='none')
plt.show()
```



In [8]:
```
def g(x,y):
    return np.sin(x+y) + (x-y)**2 -1.5*x +2.5*y + 1
x= np.linspace(-1.5,4,50)
y= np.linspace(-3,4,50)
x,y= np.meshgrid(x,y)
w= g(x,y)
plt.contour(x,y,w,40,cmap='winter')
plt.colorbar()
plt.show()
```

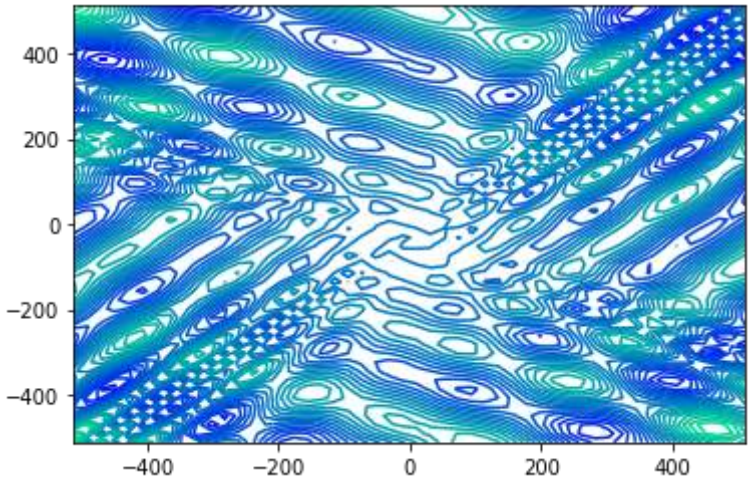


In [9]:
```
fig = plt.figure()
ax= plt.axes(projection="3d")
ax.plot_surface(x,y,w, rstride=1, cstride=1,cmap='winter', edgecolor='none')
plt.show()
```

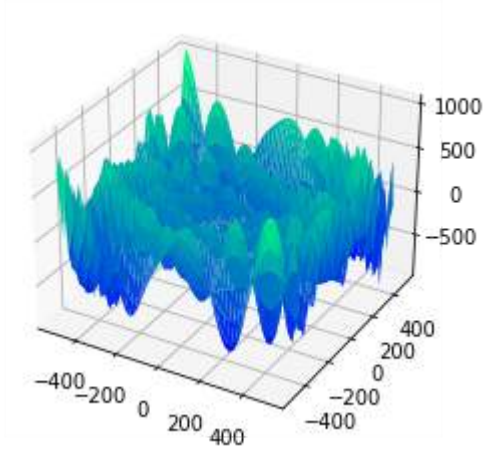


In [10]:
```
# # contour
```

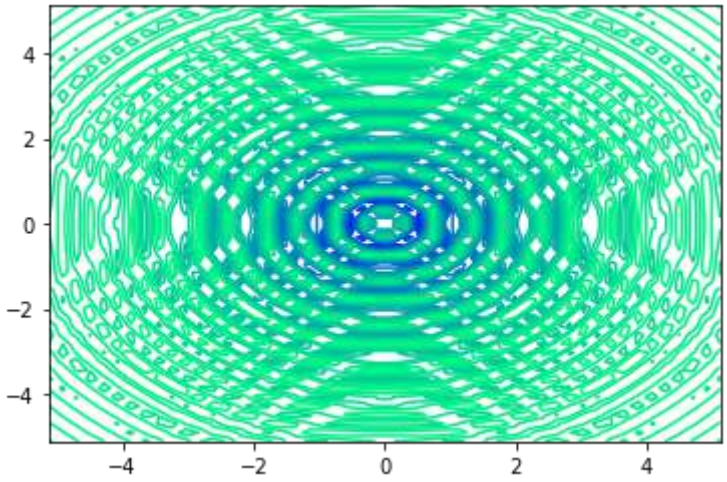
```
def g(x,y):
    return -1*(y+47)*np.sin(np.sqrt(np.abs(y+x/2 +47))) -1*x*np.sin(np.sqrt(np.abs
x= np.linspace(-512,512,50)
y= np.linspace(-512,512,50)
x,y= np.meshgrid(x,y)
z= g(x,y)
plt.contour(x,y,z,30,cmap='winter')
plt.show()
```



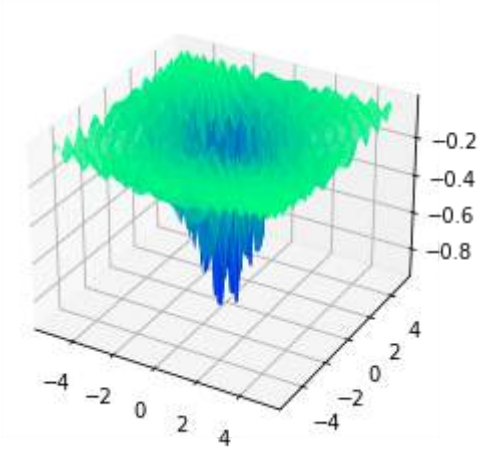
In [11]:
```
# # mesh
fig= plt.figure()
ax= plt.axes(projection="3d")
ax.plot_surface(x,y,z,rstride=1,cstride=1,cmap='winter')
plt.show()
```



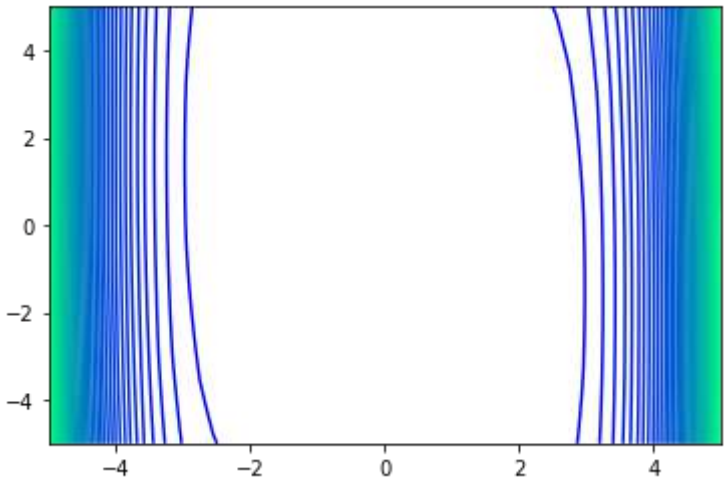
In [12]:
```
def g(x,y):
    return -1*((1+np.cos(12*np.sqrt(x**2+y**2)))/(0.5*(x**2+y**2)+2))
x= np.linspace(-5.12,5.12,50)
y=np.linspace(-5.12,5.12,50)
x,y = np.meshgrid(x,y)
z= g(x,y)
plt.contour(x,y,z,30,cmap='winter')
plt.show()
```



In [13]:
```
fig= plt.figure()
ax= plt.axes(projection='3d')
ax.plot_surface(x,y,z,rstride=1,cstride=1,cmap='winter')
plt.show()
```

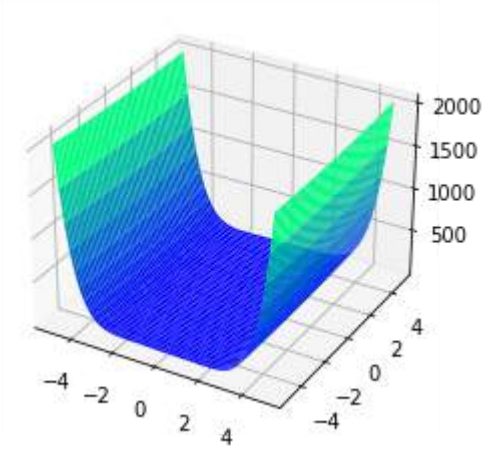


In [14]:
```
def g(x,y):
    return 2*(x**2) -1.05*(x**4) + (x**6)/6 +x*y + y**2
x= np.linspace(-5,5,50)
y=np.linspace(-5,5,50)
x,y= np.meshgrid(x,y)
z= g(x,y)
plt.contour(x,y,z,40,cmap='winter')
plt.show()
```



In [15]:
```
fig= plt.figure()
ax= plt.axes(projection='3d')
```

```
ax.plot_surface(x,y,z,rstride=1,cstride=1,cmap='winter')
plt.show()
```



In [ ]: