In [1]:

```
import numpy as np
```

In [2]:

```
import matplotlib.pyplot as plt
%matplotlib inline
```

In [11]:

```
points = np.arange(-5,5,0.01)
```

In [14]:

```
# グリッドを作る
dx,dy=np.meshgrid(points,points)
```

In [15]:

```
dx
```

Out[15]:

```
array([[-5.  , -4.99, -4.98, ...,  4.97,  4.98,  4.99],
       [-5.  , -4.99, -4.98, ...,  4.97,  4.98,  4.99],
       [-5.  , -4.99, -4.98, ...,  4.97,  4.98,  4.99],
       ...,
       [-5.  , -4.99, -4.98, ...,  4.97,  4.98,  4.99],
       [-5.  , -4.99, -4.98, ...,  4.97,  4.98,  4.99],
       [-5.  , -4.99, -4.98, ...,  4.97,  4.98,  4.99]])
```

In [16]:

```
dy
```

Out[16]:

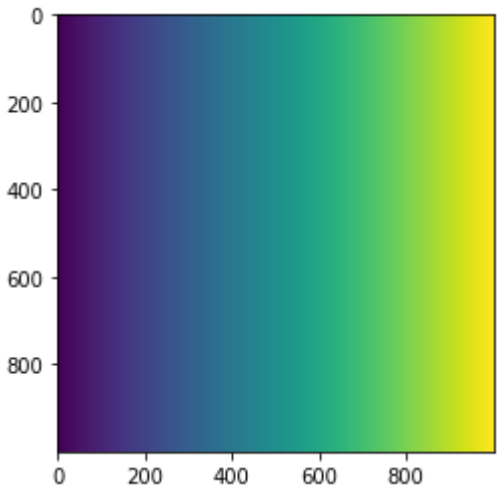
```
array([[-5.  , -5.  , -5.  , ..., -5.  , -5.  , -5.  ],
       [-4.99, -4.99, -4.99, ..., -4.99, -4.99, -4.99],
       [-4.98, -4.98, -4.98, ..., -4.98, -4.98, -4.98],
       ...,
       [ 4.97,  4.97,  4.97, ...,  4.97,  4.97,  4.97],
       [ 4.98,  4.98,  4.98, ...,  4.98,  4.98,  4.98],
       [ 4.99,  4.99,  4.99, ...,  4.99,  4.99,  4.99]])
```

In [17]:

```
plt.imshow(dx)
```

Out[17]:

<matplotlib.image.AxesImage at 0x2b21b54e888>

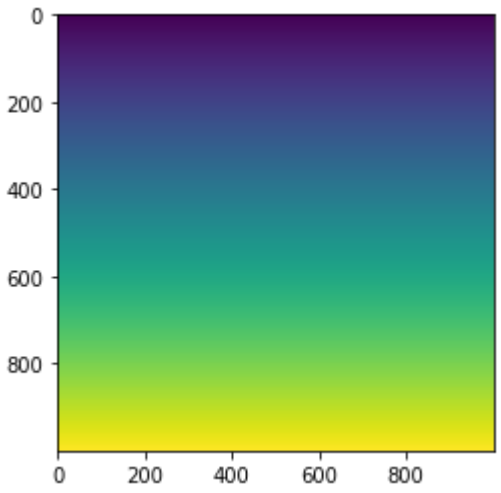


In [18]:

```
plt.imshow(dy)
```

Out[18]:

<matplotlib.image.AxesImage at 0x2b21d043608>



In [19]:

```
z=(np.sin(dx)+np.sin(dy))
```

In [20]:

```
z
```

Out[20]:

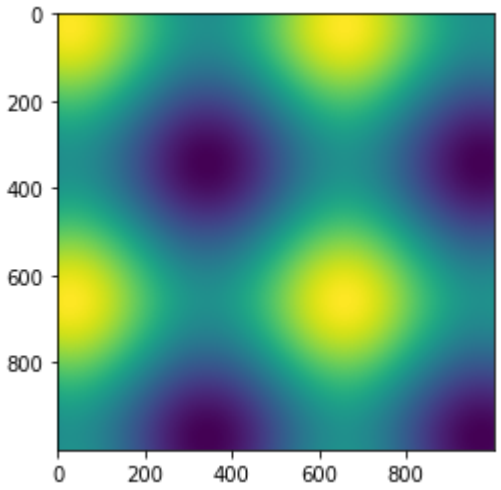
```
array([[ 1.91784855e+00,  1.92063718e+00,  1.92332964e+00, ...,
        -8.07710558e-03, -5.48108704e-03, -2.78862876e-03],
       [ 1.92063718e+00,  1.92342581e+00,  1.92611827e+00, ...,
        -5.28847682e-03, -2.69245827e-03, -5.85087534e-14],
       [ 1.92332964e+00,  1.92611827e+00,  1.92881072e+00, ...,
        -2.59601854e-03, -5.63993297e-14,  2.69245827e-03],
       ...,
       [-8.07710558e-03, -5.28847682e-03, -2.59601854e-03, ...,
        -1.93400276e+00, -1.93140674e+00, -1.92871428e+00],
       [-5.48108704e-03, -2.69245827e-03, -5.63993297e-14, ...,
        -1.93140674e+00, -1.92881072e+00, -1.92611827e+00],
       [-2.78862876e-03, -5.85087534e-14,  2.69245827e-03, ...,
        -1.92871428e+00, -1.92611827e+00, -1.92342581e+00]])
```

In [21]:

```
plt.imshow(z)
```

Out[21]:

```
<matplotlib.image.AxesImage at 0x2b21b5541c8>
```

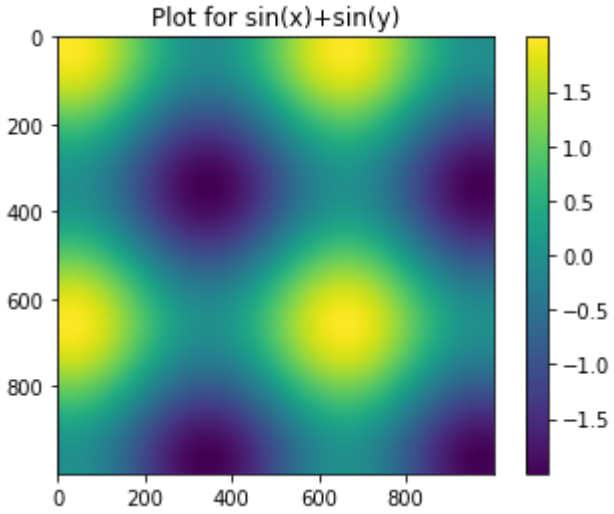


In [23]:

```
plt.imshow(z)
plt.colorbar()
plt.title('Plot for sin(x)+sin(y)')
```

Out[23]:

```
Text(0.5, 1.0, 'Plot for sin(x)+sin(y)')
```



In [24]:

```
A = np.array([1,2,3,4])
```

In [25]:

```
B=np.array([1000,2000,3000,4000])
```

In [28]:

```
# numpy 条件に合った値をとってくる

# 早くないやり方
A = np.array([1,2,3,4])
B= np.array([100,200,300,400])

# 真偽値のアレイ
condition = np.array([True,True,False,False])

# リスト内包表記を使った例
answer = [(a if cond else b) for a,b,cond in zip(A,B,condition)]
```

In [29]:

```
answer
```

Out[29]:

```
[1, 2, 300, 400]
```

In [30]:

```
# numpy.whereを使う
answer2 = np.where(condition,A,B)
answer2
```

Out[30]:

```
array([  1,   2, 300, 400])
```

In [31]:

```
#np.whereは2次元のアレイにも使える
from numpy.random import randn
arr = randn(5,5)
arr
```

Out[31]:

```
array([[-1.69700489, -0.94527922, -0.71986686,  0.5868675 , -0.36649892],
       [ 0.52880149, -1.84349713, -0.23781421,  0.04946025,  1.30279397],
       [-1.07642694,  0.25456671,  1.6381167 ,  1.17315222, -1.99543922],
       [ 2.69026045, -0.29680055,  0.24988415, -0.08749234,  0.72009259],
       [ 0.18515058,  0.14867018,  1.68159461,  1.10379028, -1.28224215]])
```

In [32]:

```
# 0より小さければ0を。そうでなければ、元の値を。
np.where(arr < 0,0,arr)
```

Out[32]:

```
array([[0.        , 0.        , 0.        , 0.5868675 , 0.        ],
       [0.52880149, 0.        , 0.        , 0.04946025, 1.30279397],
       [0.        , 0.25456671, 1.6381167 , 1.17315222, 0.        ],
       [2.69026045, 0.        , 0.24988415, 0.        , 0.72009259],
       [0.18515058, 0.14867018, 1.68159461, 1.10379028, 0.        ]])
```

In [33]:

```
# その他の統計的な計算
arr = np.array([[1,2,3],[4,5,6],[7,8,9]])
arr
```

Out[33]:

```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

In [34]:

```
# 合計
arr.sum()
```

Out[34]:

45

In [35]:

```
#計算を進める軸を指定出来る
arr.sum(0)
```

Out[35]:

array([12, 15, 18])

In [36]:

```
arr.sum(1)
```

Out[36]:

array([6, 15, 24])

In [37]:

```
#平均
arr.mean()
```

Out[37]:

5.0

In [38]:

```
#平均
arr.mean(0)
```

Out[38]:

array([4., 5., 6.])

In [39]:

```
#標準偏差
arr.std()
```

Out[39]:

2.581988897471611

In [40]:

```
#分散
arr.var()
```

Out[40]:

6.666666666666667

In [41]:

```
# any と all
bool_arr = np.array([True,False,True])
# １つでもTrueがあるか
bool_arr.any()
```

Out[41]:

```
True
```

In [42]:

```
# 全部Trueか？
bool_arr.all()
```

Out[42]:

```
False
```

In [43]:

```
# アレイをソートする
# ランダムなアレイを作って、
arr = randn(5)
arr
```

Out[43]:

```
array([ 0.73515776, -0.63924318, -1.66239876,  0.12497615,  0.27204485])
```

In [44]:

```
# ソートする
arr.sort()
arr
```

Out[44]:

```
array([-1.66239876, -0.63924318,  0.12497615,  0.27204485,  0.73515776])
```

In [45]:

```
# uniqueも便利
countries = np.array(['France', 'Japan', 'USA', 'Russia','USA','Mexico','Japan'])
np.unique(countries)
```

Out[45]:

```
array(['France', 'Japan', 'Mexico', 'Russia', 'USA'], dtype='<U6')
```

In [46]:

```
# in1d test values in one array
np.in1d(['France','USA','Sweden'],countries)
```

Out[46]:

```
array([ True,  True, False])
```

In [ ]:

```
```