```
import torch
import numpy as np

data = [[1,2],[3,4]]
x_data = torch.tensor(data)

np_array = np.array(data)
x_np = torch.from_numpy(np_array)

x_ones = torch.ones_like(x_data)
print(f"One tensor:\n {x_ones} \n")

x_rand = torch.rand_like(x_data, dtype=torch.float)
print(f"Random Tensor:\n {x_rand} \n")

shape = (2,3,)
rand_tensor = torch.rand(shape)
ones_tensor = torch.ones(shape)
zeros_tensor = torch.zeros(shape)

print(f"Random Tensor: \n {rand_tensor} \n")
print(f"Ones Tensor: \n {ones_tensor} \n")
print(f"Zeros Tensor: \n {zeros_tensor} \n")

tensor = torch.rand(3,4)

print(f"Shape of tensor: {tensor.shape}")
print(f"Datatype of tensor: {tensor.dtype}")
print(f"Device tensor is stored on: {tensor.device}")

if torch.cuda.is_available():
    tensor = tensor.to("cuda")

tensor = torch.ones(4,4)
print("First row: ",tensor[0])
print("First colum: ", tensor[:, 0])
print("Last column:", tensor[..., -1])
```

```
tensor[:,1]=0
print(tensor)

t1 = torch.cat([tensor, tensor, tensor], dim=1)
print(t1)

y1 = tensor @ tensor.T
print(f"y1: {y1} \n")
y2 = tensor.matmul(tensor.T)
print(f"y2: {y2} \n")

y3 = torch.rand_like(tensor)
torch.matmul(tensor, tensor.T, out=y3)
print(f"y3: {y3} \n")

z1 = tensor*tensor
print(f"z1: {z1} \n")
z2 = tensor.mul(tensor)
print(f"z2: {z2} \n")

z3 = torch.rand_like(tensor)
torch.mul(tensor, tensor, out=z3)
print(f"z3: {z3} \n")

agg = tensor.sum()
agg_item = agg.item()
print(agg_item, type(agg_item))

print(tensor,"\n")
tensor.add_(5)
print(tensor)

t = torch.ones(5)
print(f"t: {t}")
n = t.numpy()
print(f"n: {n}")
```

```
t.add_(1)
print(f"t: {t}")
print(f"n: {n}")

n = np.ones(5)
t = torch.from_numpy(n)

np.add(n, 1, out=n)
print(f"t: {t}")
print(f"n: {n}")
```