```python
import numpy as np
import numpy.random as npr
from test_util import *
from funkyyak import grad
npr.seed(1)

def test_grad_fanout():
    fun = lambda x : np.sin(np.sin(x) + np.sin(x))
    df = grad(fun)
    check_grads(fun, npr.randn())
    check_grads(df, npr.rand())

def test_grad_const():
    fun = lambda x : 1
    df = grad(fun)
    assert np.allclose(df(2.0), 0.0)

def test_grad_identity():
    fun = lambda x : x
    df = grad(fun)
    ddf = grad(df)
    assert np.allclose(df(2.0), 1.0)
    assert np.allclose(ddf(2.0), 0.0)

def test_hess_vector_prod():
    npr.seed(1)
    randv = npr.randn(10)
    def fun(x):
        return np.sin(np.dot(x, randv))
    df = grad(fun)
    def vector_product(x, v):
        return np.sin(np.dot(v, df(x)))
    ddf = grad(vector_product)
```

```
34          A = npr.randn(10)
35          B = npr.randn(10)
36          check_grads(fun, A)
37          check_grads(vector_product, A, B)
38
39      # TODO:
40      # Grad three or more, wrt different args
41      # Diamond patterns
42      # Taking grad again after returning const
43      # Empty functions
44      # 2nd derivatives with fanout, thinking about the outgrad adder
```