```
1: Test Preview: Summary for jsh114 of c4
2: ---------------------------------------
3:
4:   Public Tests:
5:     Marks for coding exercises::  10 / 10
6:
7: Git Repo: git@gitlab.doc.ic.ac.uk:lab1819_autumn/496differentiation_jsh114.git
8: Commit ID: aba0f
```

```
# -*- coding: utf-8 -*-

"""
Use this file for your answers.

This file should been in the root of the repository
(do not move it or change the file name)

"""

# NB this is tested on python 2.7. Watch out for integer division

import numpy as np

def grad_f1(x):
    """
    4 marks

    :param x: input array with shape (2, )
    :return: the gradient of f1, with shape (2, )
    """
    x1, x2 = x
    dx1 = 8*x1 - 2*x2 -1
    dx2 = 8*x2 - 2*x1 -1
    return np.array([dx1, dx2])

def grad_f2(x):
    """
    6 marks

    :param x: input array with shape (2, )
    :return: the gradient of f2, with shape (2, )
    """
    x1, x2 = x
    dx1 = (2*x1-2)*np.cos(x1**2+x2**2-2*x1+1) + 6*x1 - 2*x2 - 2
    dx2 = 2*x2*np.cos(x1**2+x2**2-2*x1+1) + 6*x2 - 2*x1 + 6
    return np.array([dx1, dx2])

def grad_f3(x):
    """
    This question is optional. The test will still run (so you can see if you are correct by
    looking at the testResults.txt file), but the marks are for grad_f1 and grad_f2 only.

    Do not delete this function.

    :param x: input array with shape (2, )
    :return: the gradient of f3, with shape (2, )
    """
    x1, x2 = x

    dx1 = (
        (2*x1-2)*np.exp(-x1**2-x2**2+2*x1-1) +
        (6*x1-2*x2-2)*np.exp(-3*x1**2-3*x2**2+2*x1*x2+2*x1-6*x2-3) +
        float(20*x1)/(100*x1**2+100*x2**2+1)
    )

    dx2 = (
        (2*x2)*np.exp(-x1**2-x2**2+2*x1-1) +
        (6*x2-2*x1+6)*np.exp(-3*x1**2-3*x2**2+2*x1*x2+2*x1-6*x2-3) +
        float(20*x2)/(100*x1**2+100*x2**2+1)
    )

    return np.array([dx1, dx2])

```

```
1: -------- Test Output --------
2:
3:
4: -------- Test Errors --------
5: ...
6: ----------------------------------------------------------------------
7: Ran 3 tests in 0.001s
8:
9: OK
```