Name:

Advanced Programming, II Semester, 2014–2015

Quiz 2, 2 March 2015

Answer all questions in the space provided. Use the designated space for rough work, if any.
Don't forget to fill your name!

1. Complete the following function definition so that it behaves as described below—that is, fill in the parameters for `f()` in the correct order, with default values, as appropriate.

```
def f(.........):
  print("a",a,"b",b,"c",c,"d",d)
```

Expected behaviour:

```
>>> f(b=4,a=3)
a 3 b 4 c 10 d 15

>>> f(3,5,7)
a 3 b 5 c 7 d 15

>>> f(3,c=7)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: f() takes at least 2 arguments (2 given)
```

(4 marks)

Solution:

```
def f(a,b,c=10,d=15):
```

- The first line gives us the default values for `c` and `d`.
- The second line tells us the positions for `a` and `b`.
- The third line tells us that neither `a` nor `b` has default values.

Rough work

2. The function `minout(inl)` takes a list `inl` of distinct natural numbers (i.e., integers from the set $\{0,1,2,\ldots\}$) and returns the smallest natural number not present in `inl`. In other words, if 0 is not in `inl`, then `minout(inl)` returns 0, else if 0 is in `inl` but 1 is not in `inl`, them `minout(inl)` returns 1, etc.

   For example, `minout([1,3,2,4,17])` is `0` and `minout([1,3,0,2,4])` is `5`.

   Here is a recursive algorithm to compute `minout`. (Note that `inl` is *not* assumed to be sorted, nor do we make any assumptions about the range of values in `inl`.)

   - Suppose the length of `inl` is n. Construct two lists `lower` and `upper`, where `lower` contains elements smaller than $\lfloor \frac{n}{2} \rfloor$ and `upper` contains the rest.
   - If the size of `lower` is strictly smaller than $\frac{n}{2}$, the missing number is smaller than $\frac{n}{2}$, so recursively invoke `minout` on `lower`.
   - Otherwise, invoke `minout` on `upper`. (All numbers in `upper` are bigger than $\frac{n}{2}$, so some offset is required.)

   Analyze the running time of this algorithm. (Write a recurrence and solve it.) *(6 marks)*

   Solution:

   The recurrence is:

   - $T(1) = 1$
   - $T(n) = T\left(\dfrac{n}{2}\right) + n$

   Unwinding this we get $n + \dfrac{n}{2} + \dfrac{n}{4} + \ldots 1 = O(n)$.

Rough work