

Name:

Advanced Programming, II Semester, 2011–2012
Quiz 2, 9 February 2012

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

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

```
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):
```

Note: Question 2 on reverse.

2. Assume we have defined a Python class `Node` to implement lists where each `Node` object stores data with names `self.value` and `self.next`. We use the representation where each list is terminated by a `Node` object with `self.value` and `self.next` set to `None`. Assume that in the class `Node` we have defined functions `append()` and `insert()` so that `l.append(x)` adds a node with value `x` to the end of the list pointed to by `l` and `l.insert(x)` adds a node with value `x` at the beginning of the list pointed to by `l`. Write a function `reverse()` so that `l.reverse()` reverses the list pointed to by `l`.

(6 marks)

Solution:

```
def reverse(self):

    # typical Haskell solution
    #
    # reverse Nil = Nil
    # reverse (x:xs) = (reverse xs) ++ [x]

    # if list is empty, do nothing
    if self.value == None:
        return

    # at this point list is nonempty, so
    # reverse tail and append head

    # reverse tail
    self.next.reverse()

    # append head
    self.append(self.value)

    # "remove" the head by copying self.next to self
    self.value = self.next.value
    self.next = self.next.next
    return
```