Department of Computer Science

CSC 326H1F — Fall 2006

Test 1

Aids allowed: one 2-sided sheet

Time: 50 minutes

Total marks: 45

name: ______________________________

student number: ____________________

1	
2	
3	
Total	

1. [20 marks]

The secret police in the repressive but imaginary country of Elbonia have arrested a number of people suspected of democratic tendencies. Every suspect has a list of telephone contacts, and the police want to add this information to their collection of suspicious phone numbers. Cleverly, the suspects have used computer files for their phone lists, hoping this will be too much for the police.

(a) [5 marks]

The police are surprisingly adept with Python, however, and want to transform the suspects' phone-list files into Python dictionaries — one dictionary per suspect. The files contain one line per phone number, something like this:

```
Jim      0847
Mary     3298
```

The desired dictionaries are to have names as keys and phone numbers as values. Write a Python function `ptd` (for "phones-to-dictionary") that takes a file name as its single parameter and returns a dictionary containing all the phone-number information from the file.

For example, if Bill's phone list is in the file "`Bill`" — as you'll see, we want the files to be named after the suspects — and it contains a phone number for Mary, then we should be able to use these Python commands to extract Mary's phone number:

```
bills_list = ptd("Bill")
print bills_list['Mary']
```

Both names and phone numbers can be kept as strings. You can assume that all names are single tokens ("words"). The Python string method `split` can be used to break up a line. Assume there are no errors in the data.

1. (continued)

(b) [5 marks]

There are many suspects, each with their own phone list, every one of them in a separate file — one file per suspect.

Write a Python function `ftd` (for "files-to-dictionary") that takes a list of files as its parameter and returns another dictionary, in which the keys are file names and the value for a particular key is the phone-list dictionary taken from that file. Use `ptd` from part (a) to read the phone-list files.

For example, if Bill, Jim and Mary are the suspects, then we might call `ftd` like this:

```
    combined_phones = ftd(['Bill', 'Jim', 'Mary'])
```

And then we could extract Bill's phonebook like this:

```
    bills_list = combined_phones['Bill']
```

(c) [2 marks]

Suppose `combined_phones` has been constructed as in the example at the end of part (b). Write a one-line Python command that will extract the phone number that Bill has listed for Mary from `combined_phones` and print it. Assume that there is such a phone number.

1. (continued)

(d) [8 marks]

Suppose again that `combined_phones` has been constructed as in the example at the end of part (b). Write a Python function `rpl` ("reverse phone list") that takes one parameter, a dictionary such as `combined_phones`, and returns a list of all the phone numbers found in `combined_phones`, with each phone number appearing only once.

Or, for a 2-mark bonus:

The basic part(d) question requires `rpl` to return a list of phone numbers. For the bonus, `rpl` should instead return a dictionary in which the keys are the phone numbers and the value for a particular phone number is a list of the names of all suspects' files that contain the key (the phone number).

If you answer both the basic part (d) and the bonus part, only the bonus part will be marked.

2. [20 marks = 5 + 3 × 5]

This question is about machine parts — physical objects that may wear out. Each part has a specified lifetime after which it can no longer be counted on.

(a) [5 marks]

Because the wearing-out process is related to the age of the parts, we need a `date` module to represent the date of manufacture and the current date. Python has similar standard modules, but this one is special to this question. Here is the complete module:

```
class Date(object):
    date = 0.0
    def __init__(self):
        self.date = Date.date
    def __sub__(self, other):
        return self.date - other.date
    def __repr__(self):
        return `self.date`
    def set(cls, date):
        cls.date = date
    set = classmethod(set)
```

Objects (instances) of the class `date.Date` are used to represent dates.

i. What is the purpose of the body of the constructor, the "`self.date = Date.date`" statement?

ii. What is the purpose of the `__sub__` method?

iii. What is the purpose of the `set` method?

iv. Parts of this question refer to `date.Date`, and other parts to `Date.date`. What's the difference?

In the rest of the question, you will create `date.Date` objects and subtract them, but you will not need the `__repr__` and `set` methods. You can assume the units of time are whatever is used in the `date.Date` class.

2. (continued)

(b) [15 marks = 5 for each class]

The machine parts in this question may be simple parts with specified lifetimes, or they may be compound parts made of other parts. For a compound part, the lifetime is the smallest of the lifetimes of its component parts.

We represent these kinds of parts with Python classes, `SimplePart` and `CompoundPart`. Both are children of a parent class, `BasicPart`. We never create a `BasicPart` on its own, but it provides features that are needed by both child classes.

Here's an example of how these classes might be used:

```
wire = SimplePart()
nail = SimplePart(12.0)
screw = SimplePart(18.0)
board = SimplePart(5.0)
moulding = SimplePart(10.0)
switch = CompoundPart([wire, screw, moulding])
bulb = SimplePart(2.0)
lamp = CompoundPart([switch, wire, moulding, bulb])
```

The lifetime of a lamp is 2.0, because that's the lifetime of a bulb, the shortest-lived of the parts of a lamp.

Write all three classes — `BasicPart`, `SimplePart` and `CompoundPart`. Every class must have a constructor and a method `worn_out` that returns true if the object has passed its expiry date, based on the time between the date when it was created and the current time.

A good solution will use inheritance to avoid re-writing methods where possible.

Hints etc.:

- The `date` module gives you all that you need to handle part lifetimes, and you must use it rather than any standard time-related Python module.
- Assume that a compound part is made of other parts that are new. That is, we do not put a year-old bulb in a new lamp (presumably reducing the lifetime of the lamp by a year).
- Python has a `min` function that finds the minimum value in a list.

2. (continued)

(You shouldn’t need the entire page for your answer.)

3. [5 marks]

Write a Python function `length` that takes a list `ls` as parameter and returns the number of elements in `ls`. That is, it returns the same value as the built-in function `len(ls)`.

Rules:

- You may not use the built-in `len` or the list method `__len__`.
- You may not use recursion.
- You may not iterate over `ls`, as in "`for item in ls:`".
- However, you may extract single elements of `ls`, as in "`ls[i]`".