Please fill in your Student Number and Name.

Student Number : _____

Name:

_____

_____

Student Number:

_____

University of Cape Town ~ Department of Computer Science

Computer Science 1018F ~ 2009

Test 1

Question	Max	Mark	Internal	External
1	10			
2	5			
3	15			
TOTAL	**30**			

Marks : 30

Time : 40 minutes

Instructions:

a) Answer all questions.

b) Write your answers in the space provided.

c) Show all calculations where applicable.

Question 1 [10]

a) What are the fundamental characteristics of the following native Python types: [6]

i. Dictionary

Mutable strucutre (1/2) which associates an immutable key with a mutable value. Values are only accessible through the key (1/2). Can contain mixed types in both keys and values (1/2). No notion of inherent ordering of keys & values (1/2) [any 3 out of 4 answers acceptable]

ii. List

Dynamic (mutable) structure (1/2) capable of storing mixed types (1/2). Allows array-style indexing (1/2). Can grow dynamically (1/2) [any 3 out of 4 answers acceptable]

iii. Tuple

Non-dynamic (immutable) structure (1/2) but capable of storing mixed types (1/2). Similar to list in that it allows array-style indexing (1/2). Fixed in size (1/2). [any 3 out of 4 answers acceptable]

iv. String

Effectively a tuple containing characters [1/2] All characteristics of tuples apply (same marks for these)

b) Mention three advantages of the Python programming language. [3]

Choose any three from (1 mark each): strong library support, highly portable, easy to learn (because of the emphasis on predictability and not requiring too much scaffolding), automatic memory management, emphasis on simplicity.

c) Write a Python statement that will print out *Hello World!* [1]

print 'Hello World!'

Question 2 [5]

Consider the following program and answer the questions that follow.

```
def dosomething(l1):

      l2 = []
      for e in l1:
         l2.append(e+e)
      print l2
      return l2
```

a) What does this function do? [2]

Doubles the elements of a list and places the results in a new list which is printed and returned (assuming a list of numbers is passsed in).

b) What is the output if the input is [1,2,3] [1]

[2,4,6]

c) What is the output if the input is [‘a’, ‘b’, ‘c’] [1]

[‘aa’, ‘bb’, ‘cc’]

d) What characteristics of Python explain your answers to (b) and (c) [1]

Function parameters are assigned through dynamic typing (which allows different types to be passed in to a function)

Question 3 [15]

The following is the declaration for a class that implements a sparse array of integers:

```
Class SparseArray():
      """An array that only actively stores entries that
      are different from a set value."""
```

From the user's perspective SparseArray behaves like an array structure (with elements accessible by indexing) but it is to be implemented using a Dictionary data type that only stores entries different from a set value (*setval*).

Logically, this kind of array might look like: [-1 -1 -1 5 4 -1 -1 30 -1 54] where *setval* = -1 and only 5, 4, 30 and 54 are explicitly stored.

a) What is the term for hiding the implementation details of a class like this? [1]

Encapsulation

b) Write an initialisation function that creates an empty dictionary that is not accessible from outside the class. The initialiser should take in an optional *setval* parameter for elements that are not stored explicitly (and *setval* should default to zero). [3]

```
def __init__(self, setval = 0):          [1]
    "initialise the empty dictionary"
    __d = {}                             [1]
    __v = setval                         [1]
```

c) Write methods to get and set particular entries in the sparse array. The users should be able to call these functions using standard array indexing (e.g., *sparse[5] = 10*, *x = sparse[0]*) but the values must be placed in and acquired from the private internal dictionary or the *setval* attribute. [6]

```
__getitem__(self, key):                  [1]
    "return the value in the sparse array at index key"
    if key in __d:                       [1]
        return __d[key]                  [1]
    else:
        return __v                       [1]

__setitem__(self, key, value):           [1]
    "set the value in the sparse array at index key"
    __d[key] = value                     [1]
```

d) Write a method for the SparseArray class, called *mult(x)*, that multiplies ***every*** element in the array by x. [5]

```
def mult(self, x):                  [1]
    "Multiply all elements of the sparse array by x" [1]
    for v in __d.keys():            [1]
        __d[v] = __d[v] * x         [1]
    __v *= x                        [1]
```