Indian Institute of Technology Gandhinagar

ES 112: Introduction to Computing

End-Semester Examination — Total Marks: **100**

Date: 22nd Nov 2014 — Time: **2 hr 15 minutes**.

Your info

Roll number:

Name:

Your Score

Q1	Q2	Q3	Total

Instructions :

- Write your roll number and name clearly, failing which you will lose points.
- You have to answer in the question paper itself in the given area. You will be given extra papers for rough work, which should be attached with the main sheet.
- Grading will be done based only on the answers on the main question paper.
- Do not overwrite. Unclear/Ambiguous answers may be discarded.
- For floating point numbers x, you can assume that $|x| < 10^{-16}$ essentially implies that x is zero.

---

1. **Multiple Choice Questions**. Circle the answers, note that there might be multiple correct answers to the same question. No need to give the reason.

$10 \times 2 = 20$.

(a) What are tuples good for?

i. Looking up values for a given key.
ii. Storing data that can be changed.
iii. Grouping data.
iv. Sorting.

(b) Which of the following code pieces will result in the comparison `a == [2,4,6]` being `True`.

i. `a = [1,2,3]*2`

ii.
```
a = []
for s in ‘246’:
  a.append(s)
```

iii. `a = [2**1] + [2**2] + [2**3]`

iv. `a = int(‘‘[2,4,6]’’)`

(c) Suppose dictionary `d = ‘‘john’’:40, ‘‘peter’’:45`, what happens when retrieving a value using `d[‘‘susan’’]`?

i. Since `‘‘susan’’` is not a value in the set, Python raises a ValueError exception.
ii. It is executed fine and no exception is raised, and it returns None.
iii. Since `‘‘susan’’` is not a key in the set, Python raises a KeyError exception.
iv. Since `‘‘susan’’` is not a key in the set, Python raises a syntax error.

(d) Which of the following are wrong class definitions?

i.
```
class Ex:
  pass
```

ii.
```
class Ex(object):
  pass
```

iii.
```
class Ex:
  def __init__(self,a):
    pass
```

iv.
```
class Ex:
  pass
def Ex.__init__(self,b,c):
    self.b = c
```

(e) Which are correct commands working with files?

i. `for line in read(filename):`

ii. `lines = open(filename,‘w’).read()`

iii. `open(filename).writelines(out)`

iv. `f.close()`

(f) Which commands result in an Exception?

i. `f = open(‘:::’)`

ii. `char = ‘abc’[7]`

iii. `num = [1,2,3][0]`

iv. `num = {1:‘a’}[‘a’]`

(g) Which of the `while` loops will not terminate?

i.
```
a = 1
while True:
   a = a % 2
   if(a > 0):
      break
```

ii.
```
a = 10
while True:
   a = a  % 2
   if(a > 0):
      break
```

iii.
```
a = 0
while True:
   a = 2**a
   if(a % 2 == 0):
      break
```

iv.
```
a = 1
while True:
   a = 2**a
   if(a % 2 != 0):
      break
```

(h) Which of these are not valid dictionary commands:

i.
```
d = {}
d[1,2] = 3
```

ii.
```
d = dict()
d[[1,2]] = 3
```

iii.
```
d = {‘[1,2]’: 3}
```

iv.
```
d = {}
d[‘dict()’] = dict()
```

(i) Which of the statements would yield `True`

i. `list(‘this is’) == list(‘is this’)`

ii. `list(‘t’) == tuple(‘t’)`

iii. `‘t t’ == ‘’.join(list(‘t’))`

iv. `list((1, + 2, + 3, + 4,)) == [1,2,3,4]`

(j) For which of these pieces of code does the `print` statement print something other than `None`.

i. `print list(‘2352’).sort()`

ii.
```
x = 1
def foo():
  return
x = foo()
print x
```

iii.
```
x = None
def foo(x):
  x = 2
foo(x)
print x
```

iv.
```
x = None
def foo(x):
  x = 2
  return x
x = foo(x)
print x
```

2. **For each program below, write the outputs that you will get when you run them. If it produces an error at some point in the execution write that down, and say why. If the program produces some output and then gives an error, write down the output and then write error and say why. You do not have to reproduce the exact error message that python gives out, but you need to point out why the error is occurring.** $6 \times 5 = 30$.

   (a)
   ```
   for i in [1,3,5,7,9]:
      print i, ":", i**2
   ```

   (b)
   ```
   x=2
   y=10
   for j in range(0,y,x):
      print j
      print x + y
   print "done"
   ```

   (c)
   ```
   msg = ""
   for ch in "secret".split("e"):
      msg = msg + "e" + ch + "e"
   print msg
   ```

   (d)
   ```
   def foobar(arg):
     if arg == []:
       return arg
     else:
       return foobar(arg[1:]) + [ arg[0] ]
   print foobar(range(100))
   ```

   (e)
   ```
   def fe(n):
     if n==0:
       return True
     else:
       return fo(n-1)
   def fo(n):
     if n==0:
       return False
     else:
       return fe(n-1)

   print fo(100)
   print fe(120)
   ```

   (f)
   ```
   def foo(n):
     if n == 1:
   ```

```
        return True
    if n%3 != 0:
        return False
    return foo(n//3)

print foo(81)
print foo(28)
```

3. **Write down a program for each of the following questions.**: 10 + 10 + 5 + 15 + 10.

   (a) Pig Latin is a constructed language game in which words in English are altered according to a simple set of rules. Pig Latin takes the first consonant (if it exists) of an English word, moves it to the end of the word and suffixes an `ay`. If the first letter of the word is a vowel, then we do not move it, and just add `way` at the end. For example, `pig` yields `igpay`, `banana` yields `ananabay`, and `trash` yields `rashtay`, but `egg` gives `eggway` and `inbox` gives `inboxway`. Write down a python function `translate(sent)` that takes a sentence and outputs the word by word Pig Latin conversion of that sentence. Do not worry about the capitalization.

   For example:

   `Good Luck to You` → `oodgay ucklay otay youway`.

(b) Write a function `flatten(mylist)` that takes any list of lists (of arbitrary depth) and flattens it into a single list. For example, your function should have the following behaviour.
`flatten([1,2,3])` → `[1,2,3]`.
`flatten([[1],2,[1,2,3]])` → `[1,2,1,2,3]`.
`flatten([[1],2,[1,2,[1,5,2]]])` → `[1,2,1,2,1,5,2]`.

(c) Without using a loop, using only map/reduce/filter, write a program to do the following: given a number n, return 2^m such that $2^m \leq n < 2^{m+1}$. Feel free to use the `math` library if needed.

(d) A polynomial of degree n in a single variable x is written as $f(x) = a_0 + a_1x + \ldots + a_nx^n$ where the a_i are numbers. Hence it can be represented by storing at most $n+1$ coefficients $a_0, a_1, \ldots, a_n$ (which can be negative, positive or zero). Create a class `Poly` that has the following properties:

- Will be initialized by the function `__init__(a)` which sets the value of a_i to be the i^{th} element in `a`.
- Will have a function `evaluate(x)` which will evaluate the value of the polynomial at `x`.
- Will override the addition operator to add two polynomials meaningfully. Also use type based dispatching to meaningfully define addition with a number (integer or float).
- Will have a function `degree()` which returns the degree of the polynomial.
- A function `testzero()` to test whether the polynomial gives zero on all inputs.

...

(e) In doing text analysis, it is important to see which set of words follow a given word. Write down a function `findFollow(mystr)` that does the following: for all the (space separated) words in string `mystr`, it outputs the list of unique words that immediately follows it. Assume string is in lower case only. For instance, if the input string is

```
the sound they make
the sound i make
autumn leaves
```

The output will be :

```
sound: they,i
the: sound
make: the,autumn
leaves:
they: make
I: make
autumn: leaves
```