

Sample Quiz Questions, Quiz 2

Sample Quiz Questions

Here some sample quiz questions. There are not intended to provide comprehensive coverage of the material covered thus far in 6.00. However, they should give you a sense of the kinds of questions that will be on the quiz.

Please note that while this set of sample questions emphasizes the material covered since the first quiz, you are responsible for material that was covered by that first quiz.

This quiz will be **open book and open notes**.

1) Are each of the following True or False?

In Python, one **cannot** assign a type to a variable.

All trees are binary.

In Python, a subclass can override a function definition contained in a super class.

In Python, one **cannot** override the `__init__` method of a superclass.

Depth-first search has **lower** complexity than breadth-first search.

Black box tests should be designed **without** looking at the code to be tested.

Regression testing refers to the process used to test programs that have random variables.

Sample Quiz Questions, Quiz 2

2) What does the following code print?

```
class Shape(object):
    def __eq__(s1, s2):
        return s1.area() == s2.area()
    def __ge__(s1, s2):
        return s1.area() >= s2.area()

class Square(Shape):
    def __init__(self, h):
        self.side = float(h)
    def area(self):
        return self.side**2
    def __str__(self):
        return 'Square with side ' + str(self.side)

class Circle(Shape):
    def __init__(self, radius):
        self.radius = radius
    def area(self):
        return 3.14159*(self.radius**2)
    def __str__(self):
        return 'Circle with radius ' + str(self.radius)

def f(L):
    if len(L) == 0: return None
    x = L[0]
    for s in L:
        if s >= x:
            x = s
    return x

s = Square(4)
print s.area()
L = []
shapes = {0:Circle, 1: Square}
for i in range(10):
    L.append(shapes[i%2](i))
print L[4]
print f(L)
```

Sample Quiz Questions, Quiz 2

3) Consider the two functions specified below that are used to play a “guess a number game.”

```
def cmpGuess(guess, maxVal):
    """Assumes that guess is an integer in range(maxVal).
    returns -1 if guess is < than the magic number, 0 if it is equal
    to the magic number and 1 if it is greater than the magic number.
    The magic number is in range(maxVal)."""

def findNumber(maxVal):
    """Assumes that maxVal is a positive integer. Returns a
    number, num, such that cmpGuess(num, maxVal) == 0."""
```

Write a Python implementation of `findNumber` that guesses the magic number defined by `cmpGuess`. Your program should have the lowest time complexity possible.

4) Provide a Python implementation for the function `checkParity` specified below. (18 points)

```
def checkParity(s):
    """If s is not a str, halts with an assertion exception.
    Otherwise returns True if each character in s occurs an
    even number of times and False otherwise. E.g.,
    checkParity('abab') returns True and checkParity('ababb')
    returns False"""
```

Hint: Consider using a dictionary with chars as keys.

4) What is the algorithmic (time) complexity of the function `comp`, defined below? Be explicit about what the variables in your answer stand for.

```
def comp(s1, s2):
    res = 0
    for c1 in s1:
        res += 1
        for c2 in s2:
            res -= 1
    return res
```

5) A university offers 500 subjects. Associated with each subject is a number of units (u), an expected work load (w), and an expected value (v). A student wants to maximize his or her value, but is willing to work no more than a total of H hours and allowed to take no more than D units. Formulate this as an optimization problem. I.e., define what is to be maximized and the set of constraints. Be sure to define any variables you use.

Sample Quiz Questions, Quiz 2

6) Define each of the following

- 6.1. Directed graph
- 6.2. Cycle (in a graph)
- 6.3. Breadth-first-search

7) Characterize the efficiency, using “big O” notation, of the following code:

```
a = []
for i in range(0, n):
    a.append(i)
for i in range(0, n):
    for j in range(0, n):
        print a[i]*a[j]
```

8) Consider a class named *Queue* that looks something like

```
class Queue (object):
    def __init__(self, discipline = 'FIFO'):
        self.discipline = discipline
```

What does the following code print?

```
q1 = Queue('LIFO')
q2 = Queue('LIFO')
q3 = Queue( )
q3 = q1
print q2 == q3
print q1 == q3
```

9) Write a specification and implementation of a function that discovers whether or not an element is a member of a list in $O(\log(n))$ time, where n is the length of the list. Use the least restrictive assumption that you need to make it possible to write an implementation that works in $O(\log(n))$ time.