

PRACTICE EXAM *for* MIDTERM 2

PRACTICE MIDTERM 2

Computer Science 61A . October 13, 2015 . alvinwan.com/cs61a

- You have 1.5 hours to complete the exam.
- The exam is closed book, closed notes, closed computer, closed calculator, except one hand-written 8.5" × 11" crib sheet of your own creation and the official 61A midterm 1 study guide attached to the back of this exam.
- Mark your answers ON THE EXAM ITSELF. If you are not sure of your answer you may wish to provide a brief explanation.

First Name	
Last Name	
SID	
Email (...@berkeley.edu)	
Login (e.g., cs61a-ta)	
TA & section time	
Name of person to your left	
Name of person to your right	
<i>All the work on this exam is my own.</i> (please sign)	

0. (1 Point) On a scale of Brian to Sumukh, how do you feel? Brian _____ Sumukh

PRACTICE EXAM *for* MIDTERM 2**01. (4 Points) PUMPKIN FUN**

For each of the expressions in the table below, write the output displayed by the interactive Python interpreter when the expression is evaluated. **The output may have multiple lines.** Expressions are evaluated in order, and **expressions may affect later expressions.**

Whenever the interpreter would report an error, write Error. If execution would take forever, write Forever. Assume that you have started Python 3 and executed the following statements:

```
class Patch:
    pop = [0, 1, 2, 3, 4]
    best = 3

    def harvest(self, i):
        if self.pop.pop(i):
            print('Goodbye', i, ':(')
            self.steal(i-1)

    def steal(self, i):
        print(self.pop)
        if i % 2 == 0:
            self.pop.append(self.pop[:i])
        self.harvest(i)
```

Expression	Interactive Output
<pre>p = Patch() q = Patch() p.harvest(2)</pre>	
<pre>q.harvest(1) p.pop = [0, 1, 2, 3]</pre>	
<pre>Patch.steal(q)</pre>	
<pre>p.harvest(len(q.pop))</pre>	

PRACTICE EXAM *for* MIDTERM 2**02. (10 Points) TO BRIAN OR TO SUMUKH**

Fill in the environment diagram that results from executing the code below until the entire program is finished, an error occurs, or all frames are filled. You may not need to use all of the spaces or frames.

A complete answer will:

- Add all missing names and parent annotations to all local frames.
- Add all missing values created or referenced during execution.
- Show the return value for each local frame.

Remember: Do not add a new frame when calling a built-in function (such as `abs`). The built-in `abs` function is always written as `func abs(...) [parent=Global]`.

```
def dilemma(denero):
    def to(sumukh):
        nonlocal denero
        def orto(brian):
            denero = sumukh[:2]
            brian[2] = sumukh[1:]
            return sumukh
        sumukh[0] = [1, 2]
        denero = orto(sumukh)
        return denero
    return to([denero]*3)
```

```
dilemma(2)
```

PRACTICE EXAM *for* MIDTERM 2**03. (15 Points) TREES ARE FOR KIDS**

(a) **(8 points)** Implement `max_function`, which returns the maximum value of $f(x, y)$, where x is an entry and y is one of its children. Both the `Tree` class and the tree data abstraction appear on the midterm 2 study guide.

Warning: Do not violate the tree data abstraction! (Exams are flammable.)

```
def max_function(t, f):
    """Returns the maximum value of f(x, y) where x is the value of an
       entry and y is the value of that entry's immediate child. 0 is the
       smallest possible value.

    >>> dist = lambda x, y: pow(x**2 + y**2, 1/2)
    >>> t = tree(3, [tree(4, [tree(5), tree(6)]),
    ... tree(2, [tree(1), tree(3), tree(7, [tree(8)])])]
    >>> max_function(t, dist) # sqrt( 7 ^ 2 + 8 ^ 2 )
    10.63014581273465
    >>> diff = lambda x, y: y-x
    >>> max_function(t, diff) # 7 - 5
    5
    """
    if _____:
        return _____
    children = _____
    return max(_____)
```

(b) **(5 points)** Using only parentheses, square brackets, “tree,” and “lambda”, complete the following so that `max_function(hug, denero)` yields 2015. Only one of the four items above may go in each blank.

```
denero = _____ x, y: x_ _ _0_ _ _ y
hug = tree(_____: _2015_, _ _____(1)_ )
max_function(hug, denero)
```

(c) **(2 points)** Compute the runtime of `max_function` with respect to the number of nodes in the tree, assuming the definition of f below is passed in as `max_function(t, f)`:

```
def f(x, y):
    for i in range(x):
        print(i)
```

Runtime:

PRACTICE EXAM *for* MIDTERM 2**04. (10 Points) TREE TRIMMING**

Implement `is_uniform`, which takes in a tree and determines whether or not the tree is uniformly wide at each level. The width of a tree at level i , is the sum of all entries at depth i . Note that this solution implements *breadth-first search*, where the tree is traversed and consequently processed a level at a time.

```
def is_uniform(t):
    """
    >>> t = tree(3, [tree(4, [tree(5), tree(6)]), tree(2,
    ... [tree(1), tree(3), tree(7, [tree(8)])])]
    >>> is_uniform(t)
    False
    >>> t = tree(8, [tree(6, [tree(3), tree(5)]), tree(2), tree(0)])
    >>> is_uniform(t)
    True
    >>> t = tree(8, [tree(6, [tree(2), tree(1)]), tree(2,
    ... [tree(2), tree(3), tree(0, [tree(8)])])]
    >>> is_uniform(t)
    True
    """
    queue, curr, vals, width = ____, ____, ____, _____
    while _____:
        t = _____
        curr = _____
        _____
        if not _____:
            if _____:
                return False
            queue, curr, vals = _____, ____, _____
    return _____
```

PRACTICE EXAM *for* MIDTERM 2**BONUS.** (0 Points) CINDY'S SOCKS AND SANDALS

Cindy is starting a new trend in footwear, but even divas need breaks. Using the following rules, determine if Cindy can and should wear sandals based on the weather:

- Cindy does not wear socks and sandals within 2 days of a rainy day.
- Nathan steals Cindy's socks and sandals every other day after a rainy day.
- Paul performs a rain dance after six days without rain, to invoke another rainstorm.
- Assume the first day is a rainy day.

```
def siri_generator():
    """
    >>> siri = siri_generator()
    >>> siri()
    'Just rained.'
    >>> siri()
    'The Nathan attacked!'
    >>> siri(4)
    'Paul danced.'
    >>> siri(3)
    'Cindy wore socks and sandals.'
    """
    data = _____
    def siri(days=1):
        nonlocal data
        msg = ''
        data += _____
        rain, nathan, paul = _____, _____, _____
        if _____:
            msg, paul, nathan, rain = _____, _____, _____, _____
        elif _____:
            msg, nathan = _____, _____
        elif _____:
            msg = _____
        else:
            msg = _____
        data = _____
        return msg
    return siri
```