What’s on the exam?

Your name (a few points)
“What does this print”
• Sequential execution – trace variable values
• Conditional execution (if) and boolean values
• Loops (for and while)
  – May require finding a pattern
• Function calls
  – Variable scope
  – Parameters and results
• Lists, Strings, ... (basics)

Write functions (s) to ...

Sequential execution, conditions

2. [5 points] What does this Python program print?

```python
a = 5
b = 3
if b > a :
    b = b + 17
print( b )
```

Reasoning about patterns in loops

3. [5 points] What does this Python program print?

```python
count = 0
for i in [ 1, 2, 3, 4 ] :
    for j in [ 1, 2, 3, 4, 5, 6 ]:
        if i == j :
            count = count + 1
print (count)
```

*You could solve this by tracing through every step ... but to solve it quickly, reason about it as a whole.*
Reasoning about patterns in loops

3. [5 points] What does this Python program print?
```python
count = 0
for i in [1, 2, 3, 4]:
    for j in [1, 2, 3, 4, 5, 6]:
        if i == j:
            count = count + 1
print(count)
```

You could solve this by tracing through every step ... but to solve it quickly, reason about it as a whole.

---

Reasoning about patterns in loops

3. [5 points] What does this Python program print?
```python
count = 0
for i in [1, 2, 3, 4]:
    for j in [1, 2, 3, 4, 5, 6]:
        if i == j:
            count = count + 1
print(count)
```

Count the pairs that are equal ... how many is that?

<table>
<thead>
<tr>
<th></th>
<th>1,1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2,2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>3,3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>4,4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Just the diagonal ... count 4 matches

---

4. [5 points] What does this Python program print?
```python
def sqsum(p, q):
    r = p*p + q*q
    return r

r = 3
s = 4
s = sqsum(r,s)
s = s - r
print(s)
```

Not the same r!

This is a question to check whether you understand the mechanism of function calls, including parameter passing, local variables, and value return.

---

```python
def sumpos(a):
    """Input a is an array of integers. Output sumpos(a) is the sum of the positive elements of a."
    For example:
    sumpos([1, -1, 2, -2, 3, -3]) = 1 + 2 + 3 = 6
    sumpos([-1, -2, -3]) = 0
    sumpos([1, 2, 3, 4, 5]) = 15
    """
```
def sumpos(a):
    """Input a is an array of integers. Output sumpos(a) is the sum of the positive elements of a.
    For example:
    sumpos([1, -1, 2, -2, 3, -3]) = 1 + 2 + 3 = 6
    sumpos([-1, -2, -3]) = 0
    sumpos([1, 2, 3, 4, 5]) = 15
    """
    Must be a loop ... summing something ... but just the positive elements

    Check your understanding with the examples ... and try your solution on them

    def longrun(a):
        """Returns the length of the longest sequence of identical elements in a. For example,
        longrun([0, 1, 1, 2, 3]) returns 2, because [1, 1] is the longest sequence of identical elements (all 1s).
        longrun("abcaabxxx") returns 3, because "aaa" and "xxx" have length 3. longrun([]) and longrun("") are 0.
        """
        There is generally one challenging question to test your algorithm design skills. Applying standard patterns is not enough for the challenge question.
        Strategy ... then pseudocode ... then code. Second draft of code on the exam.

How much Python do you need to know?

Basic control structures (if, while, for): Yes
Basic syntax: Yes (mostly)
Lists, strings, indexing, length: Yes

""".join(["abra", "cadabra"]) : No
    If I think you need a library function outside the basics, I’ll provide a reminder.
    Also, you can bring your book and notes!

Ground rules

Open book, open notes: Paper materials
No computers, cell phones, other electronics

Bring scratch paper. Use it.
Think before you code. Draft and revise.
Think clearly, write clearly.

50 minutes. Designed to be doable in 30, if you understand the material.
Example problem: W12, #2

```python
x = 5
y = 0
while x > 0 :
    y = y + x
    x = x // 2
print(y)
```

Example problem: W12, #3

```python
def isB(ch) :
    return ch == " "

def nb( s ) :
    n = 0
    for ch in s :
        if isB(ch) :
            n += 1
    return n

def q3() :
    phrase = "mystery wrapped in an enigma"
    print( nb(phrase) )
```

Design problems, W12

**Problem 4, W12**

```python
def magnify( ar ) :
    for i in range( len( ar ) ) :
        if ar[i] < 0 :
            ar[i] = 0 - ar[i]
    return

def q4() :
    weights = [-5, 5, -4, 6, -7, 3 ]
magnify(weights)
magsum = 0
    for w in weights :
        magsum += w
    print( magsum )
```

mostly_positive( ar ):

does list ar have more positive elements than negative?

strictlyAscending( ar ):

is each element (after the first) greater than the element before?

(Two different approaches can work.)

Could you combine them: Are (just) the positive elements strictly ascending?