1. The `print` function has optional *keyword arguments* which can be listed last that modify it behavior. The `print` function syntax: `print(value,...,sep=' ',end='\n', file=sys.stdout)

a) Predict the expected output of each of the following.

<table>
<thead>
<tr>
<th>Program</th>
<th>Expected Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>print('cat',5,'dog')</code></td>
<td></td>
</tr>
<tr>
<td><code>print()</code></td>
<td></td>
</tr>
<tr>
<td><code>print('cat',5,end='')</code></td>
<td></td>
</tr>
<tr>
<td><code>print('horse')</code></td>
<td></td>
</tr>
<tr>
<td><code>print('cow')</code></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Program</th>
<th>Expected Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>print ('cat',5,'dog',end='#',sep='23')</code></td>
<td></td>
</tr>
<tr>
<td><code>print ('cat',5,'dog',sep='23','horse')</code></td>
<td></td>
</tr>
<tr>
<td><code>print ('cat',5,'dog',sep='&gt;!*3')</code></td>
<td></td>
</tr>
</tbody>
</table>

2. Review of assignment statements. Predict the output of the following programs:

```python
da = 123
b = a
a += 1
print ('a is', a)
print ('b is', b)
c = ['cat', 'dog']
d = c
c.append('cow')
print('c is', c)
print('d is', d)
c = 'cat'
d = c
c += 'fish'
print('c is', c)
print('d is', d)
```

Most simple programs have a similar functional-decomposition design pattern (IPO - Input, Process, Output):

```python
""" Simple IPO program to sum a list of numbers. """
def main():
    label, values = getInput()
    total = sum(values)
    displayResults(label, total)

def getInput():
    """ Get label and list of values to sum."""
    label = input("What are we summing? ")
    numberOfValues = int(input("How many values are there? "))
    values = []
    for i in range(numberOfValues):
        values.append(eval(input("Enter the next number: ")))
    return label, values

def displayResults(label, total):
    """ Display sum of values. """
    print("The sum of", label, "values is", total)
main()  # starts the main function running
```

What are we summing? money
How many values are there? 4
Enter the next number: 10
Enter the next number: 20
Enter the next number: 30
Enter the next number: 50
The sum of money values is 110
When a function is called, a call-frame is pushed onto the run-time stack part of memory. A call-frame contains information about the function: (1) return address -- where function was called from, (2) formal parameters, and (3) local variables -- temporary values created inside the function.

def main():
    label, values = getInput()
    total = sum(values)
    displayResults(label, total)

def getInput():
    """ Get label and list of values to sum."""
    label = input("What are we summing? ")
    numberOfValues = int(input("How many values are there? "))
    values = []
    for i in range(numberOfValues):
        values.append(eval(input("Enter the next number: ")))
    return label, values

def displayResults(label, total):
    """ Display sum of values. """
    print("The sum of", label, "values is", total)

start here

Sequence of snapshots of the run-time stack during execution of program

### At start of main
- r.a.
- label
- numberOfValues
- values
- total

### At start of getInput
- r.a.
- **(**
- label
- numberOfValues
- values
- i

### At end of getInput
- r.a.
- **)**
- label
- numberOfValues
- values
- i

### After returning to (** from getInput
- "money"

### After calling and returning returning to (** from sum
- r.a.
- **(****
- label
- total

### At start of displayResults
- r.a.
- (****
- label
- values
- total

### After returning to (** from displayResults
- "money"

### After returning to (*) from main
- [10,20,30,50]

Lecture 1 Page 2
3. Design a program to roll two 6-sided dice 1,000 times to determine the percentage of each outcome (i.e., sum of both dice). Report the outcome(s) with the highest percentage.
   a) How would you solve this problem “by hand” without a computer?

b) What built-in Python module/function can be used to simulate a 6-sided die?

c) What built-in data structure(s) could be used to tally the 1,000 roll outcomes?

d) Customize the functional-decomposition diagram for the dice problem by briefly describing what each function does and what parameters are passed.