Name: ________________________________________________

Directions: MAKE SURE TO COPY YOUR ANSWERS TO A SEPARATE SHEET FOR SENDING ME AN ELECTRONIC COPY LATER.

**1.** (25) Consider the grading program example in Section 1.9. Suppose an instructor perversely sets a rule that the *highest* few quiz grades would be dropped. State a single line to be changed in the existing code, and state what the new line would be. For example, if you think that Line 32 should be changed to **x = 88**, write your answer as

```
32 x = 88
```

**2.** Consider each part of this problem as a separate, independent but complete Python interactive session (minus output and error messages, if any). For each part, answer "error" or "no error", respectively, depending on whether the an error message occurs from execution of the code.

(a) (5)

```
>>> x = [[1,2,3],[5,12,13],[6,7]]
```

(b) (5)

```
>>> sqrt(9)
```

(c) (5)

```
>>> def sq(x):
...     return x*x
...
>>> y = sq
>>> y = y(3)
```

(d) (5)

```
>>> class x:
...     def __init__(self):
...         self.y = 8
...
>>> a = x()
>>> a.z = 88
```

(e) (5)

```
>>> x = (1,2,3)
>>> x.append(4)
```

**3.** (25) The function **places()** below returns a Python list of the instances of the character **c** in the string **s**. For example,

```
>>> y
'abcdefaba'
>>> places(y,'a')
[0, 6, 8]
```

Fill in the blanks.

```
def places(s,c):
   base = ______________________
   insts = ____________________
   while True:
      place = s[base:]_______________
      if place == -1: break
      insts.append(_________________)
      base = ___________________
   return insts
```

**4.** (25) The built-in Python function **map(f,sq)** (in its basic form) calls the function **f()** on each element of the sequence **sq**, returning a list consisting of the results.

The function **extractcol(j,m)** returns column **j** of the matrix **m**, where we are defining a matrix to be a two-dimensional array with the same number of elements in each row.

For instance,

```
>>> m
[[2, 3], [7, 8], [0, 28]]
>>> extractcol(1,m)
[3, 8, 28]
```

Fill in the blanks.

```
def extractcol(j,m):
   def getelt(r): return ___________________
   return ______________________
```

**Solutions:**

**1.**

```
49  tmp = tmp[:(len(tmp) - ndrop)]
```

**2.**

Parts (b) and (e) produce error messages.

**3.**

```
def places(s,c):
   base = 0
   insts = []
   while True:
      place = s[base:].find(c)
      if place == -1: break
      insts.append(base + place)
      base = base + place + 1
   return insts
```

**4.**

```
def extractcol(j,m):
        def getelt(r): return r[j]
        return map(getelt,m)
```