ECS 10
Concepts of Computation
Example Final Problems

1. Here is a little program, not necessarily correct.

```
ages= {}
ages["cat"]=4
if 4 in ages:
    print ages[4]
```

This program will...

a) print `cat`
b) print `4`
c) run without error, but not print anything.
d) cause an error, because we cannot use the string `cat` as a list index.
e) cause an error, because we cannot use the integer 4 as a dictionary key.

Answer: c. `ages` *is a dictionary. Anything can be used as a dictionary key. But there is only one thing in this dictionary, stored under the key* `"cat"`*. There is nothing stored under the key* `4`*.*

2. The following code will crash. Why?

```
bird = 'awk'
letters = []
for char in bird:
    letters = letters + char
```

a. The for loop does not include a `range` function.
b. The variable `letters` is not correctly initialized.
c. The variable `char` was not initialized at all.
d. You cannot concatenate a list and a string.
e. It should be `letters == letters+char`

Answer: d. The variable `char` *contains a character, but* `letters` *is a list.*

3. After executing this line,

```
x = 5
```

which of the following expressions have the value True?

```
1. (not (x > 9)) and (x > 6)
2. not ((x > 9) and (x > 6))
3. (not (x > 9)) or (x > 6)
4. not ((x > 9) or (x > 6))
```

a) 1, 2 and 3
b) 2, 3 and 4
c) 2 and 4
d) 2 and 3
e) 1 and 4

Answer: b. 1 is True and False == False, 2 is not (False and False) == True, 3 is True or False == True, and 4 is not (False or False), which is True.

4. The following program is not necessarily correct.

```
def minimum (L):
    min = L[0]
    for num in L:
        if num < min:
            min = num

numbers = [4,7,2,8,5,7]
minimum(numbers)
print min
```

a) This program prints `2`
b) This program prints `4`
c) This program causes an error because `num` is undefined.
d) This program causes an error because `L` is undefined.
e) This program causes an error because `min` is undefined.

Answer: e. `min` *is defined in the function, but not in the main program, so the print statement will cause an error.*

5. The following program is not necessarily correct.

```
def setUpList():
    L = []
    for i in range(4):
        L = L + [i]
    return L

numbers = setUpList()
print numbers
```

a) This program prints `[0, 1, 2, 3]`
b) This program prints `[[0], [1], [2], [3]]`
c) This program prints `[0]`
d) This program causes an error because the return value of the function is not passed correctly.
e) This program causes an error because the function has no argument.

Answer: a. The program concatenates lists correctly, and correctly returns a list of integers.

6. The following program prints....

```
for i in range(5):
    L = []
    L = L+[i]
print L
```

a) `[1,2,3,4,5]`
b) `[0,1,2,3,4]`
c) `[1]`
d) `[4]`
e) `[5]`

Answer: d. The list is initialized to the empty list every time the block in the for loop is executed, so the final value of L is the list containing only the last integer in range(5), which is 4.

7. Here is a program with a blank in it.

```
D = {}
L = ["Eric","Cecile","Walter"]
P = ["cat","rabbit","fish"]
for i in range(3):
     name = L[i]
     pet = P[i]
     -----------------
for x in L:
     print D[x]
```

It prints out:

```
cat
rabbit
fish
```

The missing line should be:

a) `D{i} = pet`
b) `D{name} = pet`
c) `D[i] = pet`
d) `D[name] = pet`
e) `D = D + {pet}`

Answer: d. You index a dictionary with square brackets, just like a list. Also, in the second for loop, the variable `x` *takes on each of the names in* `L` *in turn.*

8. The following program is not necessarily correct.

```
def setToTen():
    x = 10
    return x

x = x-1
print x
```

a) This program prints `10`
b) This program prints `9`
c) This program causes an error because `x` is not initialized.
d) This program causes an error because the return value of the function is not passed correctly.
e) This program causes an error because the function has no argument.

Answer: c. The variable `x` *is not initialized because the function* `setToTen()` *is never called. This question tests to see if you know that a function is not executed until it is called, and that local variable are invisible to the main program.*

9. Programming problem:

   Here is a program with the block inside a function definition missing. Write the complete definition of the function `lengths(L)` below. The function `lengths(L)` takes a list of strings `L` as its input argument, and returns a list containing the lengths of all the strings.

```
def lengths(L):
    --------------------
    --------------------
    --------------------

people = ["Anne","Katherine","James","Lu"]
print lengths(people)
```

   So this program should print

```
[4,9,5,2]
```

   Your function should work with any input list of strings, so that for instance if the program is changed by changing the list of strings to

```
people = ["Joe","Jose"]
```

   the output (without changing anything else in the function) should become

```
[3,4]
```

   Remember that if `x` is a string, the built-in function `len(x)` returns the number of characters in `x`, and if `x` is a list, `len(x)` returns the number of elements in `x`.

10. Programming problem:

    Your friend in animal science has been making observations on how many pounds of food the four pigs in the pig barn eat each day, over the course of a week. He has kept his observations for each day in file called `pigs.csv`, which is eight lines long, and looks like this (the middle four lines are not shown):

```
Date, Porky, Heather, Rose, Gus, Bob
12/1,5.5,4.8,5.3,5.2,4.6
12/2,5.8,5.3,4.9,5.2,4.7
...
12/7,6.2,5.5,5.5,3.6,4.8
```

    You offer to write a program for him that will read in the data and respond to queries about how much a particular pig ate over the course of the week. For instance, running your program might look like this:

```
Enter name of pig: Heather
Heather ate 4.8, 5.3, ..., 5.5
Enter name of pig: Bob
Bob ate 4.6, 4.7, ..., 4.8
Enter name of pig: Petunia
There is no pig named Petunia
Enter name of pig:
Press enter to exit
```

    (Again, the middle four numbers in each response of the computer are not shown). Your program should use a dictionary.

11. Programming Problem:

We will be given a list of prices, written as strings such as `"$12,350.34"` or `"$85.99"`. Write a function `dollars(s)` which takes such a string as its input argument, and returns an integer number of dollars, rounded down to the nearest dollar. So if you fill in your function in this program:

```
def dollars(s):
     ____________________
     ____________________
     ____________________
     ____________________
     ____________________

prices = ["$12,350.34","$85.99","$5,302.69"]
for i in range(0,3):
     print dollars(prices[i])
```

it should print

```
12350
85
5302
```