Question 1

```
def testString(aString):
    aDict = {}
    for letter in aString:
        num = aString.count(letter)
        if num not in aDict:
            aDict[num] = letter
        else:
            return num
    return -1

text = 'eager'
print(testString(text))
```

a. -1
b. 1
c. 2
d. 3
e. none of the above

Question 2

```
def container(s1, s2):
    rtnVal = ''
    for i in range(len(s2)):
        sub = s2[:i]
        if sub not in s1:
            rtnVal += sub
    return rtnVal

t1 = 'hoof'
t2 = 'hoot'
print(container(t1, t2))
```

a. '' (the empty string)
b. hhohoo
c. hhohoohoot
d. hoot
e. none of the above

Question 3

```
def reflections(t):
    tList = t.split()
    pos = 0
    anti = -1
    for word in tList:
        if tList[pos] == tList[anti]:
            return tList[pos]
        pos += 1
        anti -= 1
s = 'a thing worth doing is worth doing well'
print(reflections(s))
```

a. worth
b. '' (the empty string)
c. None
d. NameError: name 't' is not defined
e. none of the above

Question 4

```
word = 'the'
sub = ''
accum = ''
for let in word[1:]:
    for subLet in sub:
        if let < subLet:
            accum += let
            break
    sub += let
print(accum)
```

a. t
b. th
c. he
d. e
none of the above

Question 5

```
spell = {1:'one', 0:'zero', 2:'two'}
print((spell[0][1]))
```

a. o
b. n
c. z
d. e
e. none of the above

Question 6

```
ben = "Lost time is never found again"

def property(t):
    tList = t.split()
    d = {}
    for word in tList:
        length = len(word)
        if len(word) not in d:
            d[length] = 1
        else:
            d[length] += 1
    return d

print(len(property(ben)))
```

a. 5
b. 6
c. 11
d. 25
e. none of the above

Question 7

```
aDict = {1:'first int', 0:'zero'}
print(aDict[0])
```

a. 1
b. first int
c. 1:'first int'
d. zero
e. none of the above

Question 8

```
bools = [True and not True, True or not True, not not True, not False, not True
or False]
d = {True: 0, False: 0}
for expr in bools:
    if expr:
        d[True] += 1
        continue
    d[False] += 1
print(d)
```

a. {True: 1, False: 1}
b. {True: 2, False: 3}
c. {False: 2, True: 3}
d. {False: 5: True: 3}
e. none of the above

Question 9

```
import turtle
t = turtle.Turtle()
for i in range(2):
    if i%3 == 0:
        t.forward(100)
        t.left(120)
    if i%2 == 1:
        t.forward(100)
        t.left(120)
    elif i%3 == 0:
        t.forward(100)
        t.left(120)
```

a. a straight line
b. two sides of an equilateral triangle
c. an equilateral triangle
d. SyntaxError: invalid syntax
e. none of the above

Question 10

```
fishLine = ['one', 'fish', 'two', 'fish', 'red', 'fish', 'blue', 'fish']
indx = 0
while indx < len(fishLine):
    if len(fishLine[indx]) == 4:
        indx += 1
        continue
    indx += 5
print(indx)
```

a. 0
b. 1
c. 3
d. 8
e. none of the above

Question 11a (8 points)

Write a function named ***capT*** that uses turtle graphics to draw a capital letter 'T' of specified height. The function ***capT*** takes two parameters:

1. ***turt***, a turtle that is used for drawing
2. ***height***, the height of the letter T

The cross stroke of the 'T' should be half the length of the height stroke.

The function ***capT*** should draw a 'T' beginning at the initial position and orientation of ***turt***, and should leave ***turt*** with the same position and orientation on exit. Initially, ***turt*** may be in any position on the screen and have any orientation.

Question 11b (12 points)

Write a function named ***tRow*** that uses turtle graphics and the function ***capT*** from Question 11a to draw a series of capital T's of increasing (or decreasing) size. The T's should all rest on a common baseline (which is not drawn). The enclosing rectangle for each letter T should just touch but not overlap the enclosing rectangle of the next T. (Do not draw the enclosing rectangle.) Begin drawing at the initial location and direction of the turtle.

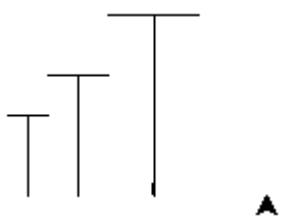
The function ***tRow*** should repeatedly call ***capT*** to draw the T's.

The function tRow takes 4 parameters:

1. ***t***, a turtle used for drawing
2. ***num***, the number of T's to draw
3. ***init***, the height of the first T
4. ***ratio***, the ratio of the heights of successive T's

For example if ***tRow*** is called by the following code, the drawing below would be correct output.

```
import turtle
s = turtle.Screen()
t = turtle.Turtle()
t.left(90)
tRow(t, 3, 40, 1.5)
```



Question 12 (20 points)
Write a function named ***fileStats***. The function ***fileStats*** takes two string parameters: ***inFile***, the name of an input file and ***outFile***, the name of an output file.

The function ***fileStats*** should read and analyze the contents of the input file and write the statistics it compiles to the output file. The statistics you should compute about the input file are:

- the number of characters;
- the number of words;
- the number of lines;
- the number of digits;
- the number of punctuation marks

Each statistic should be written to a separate line of the output file. (Hint: the string class contains constants named punctuation and digits.)

For example, if the input file contains the following lines:

Lord I'm one, Lord I'm two, Lord I'm three, Lord I'm four,
Lord I'm 500 miles from my home.
500 miles, 500 miles, 500 miles, 500 miles
Lord I'm five hundred miles from my home.

Then an output file with the following lines would be correct:

characters 181
words 35
lines 4
digits 15
punctuation 15

Question 13 (20 points)
Write a function named ***symmetry*** that takes a string, ***text***, as a parameter and returns a dictionary ***d***. Every letter that is both the first and last letter of some word in ***text*** should be a key in ***d***. For example, if ***text*** contains the word 'shucks', 's' should be a key in ***d***. The value of 's' in ***d*** is the number of words that both begin and end with 's'. The parameter ***text*** contains only upper and lower case characters and white space.

The following is an example of correct output:

```
t = '''The sun did not shine
it was too wet to play
so we sat in the house
all that cold cold wet day

I sat there with Sally
we sat there we two
and I said how I wish
we had something to do'''

print(symmetry(t))
{'d': 1, 'i': 3, 't': 1}
```