**Question 1**

```python
boolExprs = [3 > 2, 0 == 0, True, False, True or False, not False]
trueCount = 0
for expr in boolExprs:
    if expr:
        trueCount += 1
print(trueCount)
```

Hint: the line
```
trueCount += 1
```
does exactly the same thing as the line
```
trueCount = trueCount + 1
```

a. TypeError: unordered types: int, bool()
b. 3
c. 4
d. 5
e. none of the above

**Question 2**

```python
aSequence = [-1, 0, 'str', '0']
sum = aSequence[0] + aSequence[-1]
print(sum)
```

a. -1  
b. '-1'  
c. Index Error: list index out of range  
d. TypeError: unsupported operand type(s) for +: 'int' and 'str'  
e. none of the above

**Question 3**

```python
nested = [['0'], '0', 0]
print(nested[:1])
```

a. ['0']  
b. [['0']]  
c. ['0'], '0'  
d. [['0'], '0']  
e. none of the above

**Question 4**

```python
types = ['chocolate', 'vanilla', 'birthday', 'retirement']
ocasions = types[-2:]
print(types - occasions)
```

a. TypeError: unsupported operand type(s) for -: 'list' and 'list'  
b. 'chocolate', 'vanilla'  
c. ['chocolate', 'vanilla']  
d. [['chocolate', 'vanilla']]  
e. none of the above
**Question 5**
import turtle
s = turtle.Screen()
t = turtle.Turtle()
for i in range(6):
    t.forward(100)
    t.left(120)
a. an equilateral triangle with sides of length 100
b. two noncongruent equilateral triangles
c. an equilateral triangle with sides of length 200
d. TurtleError: module turtle has no method Turtle
e. none of the above

**Question 6**
def testReturn(int1, thing1):
    for i in range(int1):
        return thing1
returned = testReturn(2, 'a')
print(returned)
a. returned
b. thing1
c. thing1 thing1
d. 'a' 'a'
e. none of the above

**Question 7**
wishesWereFishes = True
wishesWereHorses = True
if wishesWereFishes:
    print('beggars would eat')
elif wishesWereHorses:
    print('beggars would ride')
else:
    print('sad rhyme but true')
a. beggars would eat
b. beggars would eat
beggars would ride
c. sad rhyme but true
d. SyntaxError: invalid syntax
e. none of the above
Question 8
bogie = ["Here's", 'looking', 'at', 'you', 'kid']
short = []
for word in bogie:
    if len(word) <= 3:
        short.append(word)
print(short)
a. [word, word, word]
b. ['word', 'word', 'word']
c) ['at', 'you', 'kid']
d) SyntaxError: 'if' inside for loop
e) none of the above

Question 9
print(zero())
def zero():
    return 0
a. 0
b. zero()
c. 'zero()'
d. NameError: name 'zero' is not defined
e. none of the above

Question 10
def echo(param):
    return param
print('here')
print(echo('there'))
a) there
    here
b) here
    there
c) here
    there
    here
d) here
    echo('there')
e) none of the above
Programming problems 11-13 (20 pts each).

Problem 11 Part A: (12 points)

Write a function named parallelLines(). The function parallelLines() takes three parameters:
1. a turtle, t
2. an integer, numLines, that is the number of parallel lines to draw
3. an integer, lineLength, that is the length of each line

The function parallelLines() should use the turtle t to draw the lines specified by the parameters numLines and lineLength. The space between parallel lines should be 25. You should not make any assumption about the state of the pen that is passed as a parameter, and there is no specification concerning the state of the pen when the function terminates.

For example, if numLines == 5 and lineLength == 100, the following is correct output

```
   ________
   ________
   ________
   ________
   ________
```

Problem 11 Part B: (8 points)

Write code that calls parallelLines(), using the parameters in the example in part A. (Hint: before calling the function, you must import the turtle module and create a turtle.)

Problem 12 Part A (12 points)

Write a function named containsLetter() that identifies all of the strings in a list that contain a specified letter and returns a list of those strings.

The function containsLetter() takes two parameters:
1. a string of length 1, aLetter
2. a list of strings, strList

For example, the following would be correct output:

```python
>>> hulkLine = ['you', 'wouldn\'t', 'like', 'me', 'when', 'i\'m', 'angry']
>>> searchLetter = 'i'
>>> print(containsLetter(searchLetter, hulkLine))
>>> ['like', 'i\'m']
```
Problem 12 Part B (8 points)

Write code that calls containsLetter() for each letter in 'hulk'. Each function call should pass one of the letters in 'hulk' and the string hulkLine as parameters. After each function call your code should print the list of words returned by containsLetter(). For full credit, you must put the call to containsLetter() inside of a for loop.

The following would be correct output.

[]
[]
["you", "wouldn't"]
["wouldn't", "like"]
["like"]

Question 13 Part A. (10 points)

Write a function named getInt(). The function getInt() takes two parameters:

i an int, minInt
ii an int, maxInt

The function getInt() should prompt the user for an integer that is no less than minInt and no greater than maxInt. You may assume that the user responds with an integer in this range. (Hint: user input is always a string.)

getInt() should return the number provided by the user.

For example, if getInt() is the called with parameters -5 and 7, the following would be correct input and output:

>>> Please give me a number no less than -5 and no greater than 7
>>> 4

Question 13 Part B (10 points)

Write code that calls getNumber(), using a negative value for minInt and a positive value for maxInt. Your code should then print out the returned integer and whether it is positive, negative or zero.