

1) The following code is riddled with bugs: some are due to bad syntax and some are due to bad logic. There are 7 bugs, each bug is on its own line and each bug is worth one mark. Circle up to 7 lines; **one mark will be deducted for each additional line circled**. This question is scored out of 6 and a maximum of 7 will be awarded.

```
import random
MIN_ASCII_VALUE = ord('A')
MAX_ASCII_VALUE = ord('z')

def encode(strToEncode) :
    encodeList = list(strToEncode)
    randomList = []
    encodeString = ('', )

    i = 0
    while i < len(encodeList) :
        if (encodeList[i] == 'A') :
            # Switch As to Zs
            encodeList[i] = 'Z'
        elif (encodeList[i] == 'Z')
            # Switch Zs to As
            encodeList[i] = 'A'

        # Create a random character for each original character
        randomList.append(chr(random.randint(MIN_ASCII_VALUE, \
                                                MAX_ASCII_VALUE)))

        i = i - 1

    # Reverse the list of original characters
    encodeList.sort()

    # Create the encoded string
    for i, aChar in enumerate(randomList, encodeList) :
        # Add the encoded character followed by a random character
        encodeString = encodeString + aChar
        encodeString = randomList[i] + encodeString

    # Return the string
    return encodeList
```

2) Examine the following small Python programs. Determine the values of the variables at the end of each program's execution. There are 11 variables and each is worth one mark. This question is scored out of 10; a maximum of 11 will be awarded.

```
a = 5
b = 7
c = 11
```

```
if (b < c) :
    b = 17
    if (a > c) :
        a = 23
    else :
        c = 7
else :
    b = 3
```

a: 5

b: 17

c: 7

```
x = 10 <= 5 * 3 or 7 > 6 + 2
y = 4 != 9 - 4 and 8 * 2 >= 11
z = not "ABC" > "ABCD"
```

x: True

y: True

z: True

```
i = 10
j = 0
while (i > 0) :
    i = i - 1
    continue
else :
    j = 5
```

i: 0

j: 5

```
aList = [8, 4, 2, 6]
subList = aList[2 : ]
del aList[3]
test = 6 in aList
```

aList: [8, 4, 2]

subList: [2, 6]

test: False

3. a) Write a function `getLargerNumber (...)` that has two parameters, each representing a non-zero integer. The function must ask the user to enter a number that is larger than the two parameters. If the user does not enter a larger number then the function must ask again. Once the user has entered a larger number, the function must return the value entered. The following is sample output from calling `getLargerNumber(5, 7)`:

```
You must provide a number larger than 5 and 7.  
Enter it now: 3  
You must provide a number larger than 5 and 7.  
Enter it now: 5  
You must provide a number larger than 5 and 7.  
Enter it now: 7  
You must provide a number larger than 5 and 7.  
Enter it now: 8
```

There is no need to write a `main()` function or a call to `getLargerNumber(...)`.
This question is scored out of 4.

```
def getLargerNumber(num1, num2) :  
    larger = num1  
    while (larger <= num1) or (larger <= num2) :  
        print("You must provide a number larger than", num1, \  
              "and", str(num2) + ".")  
        larger = input("Enter it now: ")  
        larger = int(larger)  
  
    return larger
```

b) Write a function `printMixedString(...)` with two parameters, each representing strings. These strings must be the same size; if not, the function must print an error message. Each pair of characters in the parameter strings must be examined in order. One of them is selected for printing if:

- it is an uppercase 'L'
- its ASCII value is greater than the other

The following is sample output from calling `printMixedString("Lion", "alas")` followed by `printMixedString("Hello", "alas")`:

```
Llos  
You can't mix those!
```

There is no need to write a `main()` function or a call to `printMixedString(...)`. This question is scored out of 4.

```
def printMixedString(string1, string2) :  
    if len(string1) != len(string2) :  
        print("You can't mix those!")  
        return  
  
    string3 = ''  
    for char1, char2 in zip(string1, string2) :  
        if char1 == 'L' or char2 == 'L' :  
            string3 = string3 + 'L'  
        elif char1 > char2 :  
            string3 = string3 + char1  
        else :  
            string3 = string3 + char2  
  
    print(string3)
```