Exam 1

Q1

According to the Unicode standard, what is the upper case version of the character with code point `U+00E3`?

- a: `U+00C3`
- b: `U+0040`
- c: `U+13F0`
- d: `U+2102`

Q2

To perform a case-insensitive comparison between two strings (`s1` and `s2`), which of the following would be the best approach?

- a: `s1.upper() == s2.upper()`
- b: `s1.title() == s2.title()`
- c: `s1.lower() == s2.lower()`
- d: `s1.casefold() == s2.casefold()`

Q3

Given the following string:

```
s = '3.14/4.15/6.7/8.9'
```

Which of the following will evaluate to the *number* `4.15`?

- a: `s[s.index('/'): s.index('/')][1]`
- b: `s.split('/')[1]`
- c: `float(s.split('/')[-3])`
- d: `float(s).split('/')[1]`

Q4

Given a string `data`, what code will correctly, and most efficiently, determine if some given substring `s` is present in `data`?

- a: `True if data.index(s) > 0 else False`

- b: `True if data.find(s) > 0 else False`
- c: `s in data`
- d: `[s[i] in data for i in range(len(s))]`

Q5

The following function is intended to find the index of the first negative number in a given list `numbers`:

```
def find_first_negative(numbers):
    i = 0
    while numbers[i] >= 0:
        i += 1
    return i
```

This function will return the correct result:

- a: if `numbers` contains at least one negative number
- b: if `numbers` contains only positive numbers
- c: always
- d: never

Q6

The following functions are all meant to validate an input for the following conditions:

1. the input is an integer
2. the integer is positive
3. the integer is less than `100`

and needs to raise a `ValueError` if the input does not satisfy these conditions.

I:

```
def validate(num):
    return isinstance(num, int) and num > 0 and
num < 100
```

II:

```
def validate(num):
    if not(isinstance(num, int) and num > 0 and
num < 100):
        return ValueError('Invalid input')
```

III:

```
def validate(num):
    if not(isinstance(num, int) and num > 0 and
num < 100):
        raise ValueError('Invalid input')
```

Which functions will work correctly?

- a: all of them
- b: none of them
- c: `II` and `III` only
- d: `III` only

Q7

You want to create a function that takes two positional arguments, and one optional keyword-only argument (with a default of `True`).

What should the function header look like?

- a: `def func(a, b, kwarg=True)`
- b: `def func(a, b, **kwargs, kwarg=True)`
- c: `def func(a, b, *, kwarg=True)`
- d: `def func(kwarg=True, *, a, b)`

Q8

One of the solutions to a quadratic equation:

$$ax^2 + bx + c = 0$$

is given by this formula:

$$\frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

Write a function that calculates this solution of a quadratic equations given specific values for `a`, `b`, and `c`. The result should be rounded to `2` digits after the decimal point.

For example, given the equation:

$$3x^2 + 4x - 5 = 0$$

your function should return `0.79`.

What is the result for this equation?

$$x^2 - 5x - 8 = 0$$

Q9

The following function performs a (very) simplistic encryption of a given string:

In [1]:
```python
def encrypt(s):
    return ''.join(chr(ord(c) + 10) for c in s)
```

Write a function that reverses the encryption, and decrypt the following string:

```
'S}kkm*Xo\x81~yx'
```

Q10

Given the following strings:

In [2]:
```python
currencies = 'USD, CAD, USD, JPY,  AUD'
values = [100, 200, 300, 400, 500]
```

Which of these functions will produce the following result when called with `currencies` and `values` passed as the first and second positional arguments respectively:

`func(currencies, values) --> "100 USD, 200 CAD, 300 USD, 400 JPY, 500 AUD"`

I.

```python
def func(currencies, values):
    currencies = currencies.split(',')
    result = ''
    for i in range(min(len(currencies),
len(values))):
        currency = currencies[i].strip()
        value = str(values[i])
        result = result + value + ' ' + currency +
', '
    return result.strip(', ')
```

II.

```
def func(currencies, values):
    currencies = [s.strip() for s in
currencies.split(',')]
    result = []
    for currency, value in zip(currencies, s2):
        result.append(str(value) + ' ' + currency)
    return ', '.join(result)
```

III.

```
def func(currencies, values):
    return ', '.join(
        [
            str(v1) + ' ' + v2
            for v1, v2 in zip(
                values,
                [s.strip() for s in
currencies.split(',')]
            )
        ]
    )
```

IV.

```
def func(currencies, values):
    currencies = [s.strip() for s in
currencies.split(',')]
    result = [
        ' '.join([str(v1), v2])
        for v1, v2 in zip(values, currencies)
    ]
    return ', '.join(result)
```

- a. I and II only
- b. I, III, IV only
- c. none of them
- d. all of them