In [ ]:
```
Python Function to Check Password Validity
```

In [*]:
```
import re

def check_password(password):
    # Check if the password has at least two uppercase and two lowercase le
    if len(re.findall(r'[A-Z]', password)) >= 2 and len(re.findall(r'[a-z]'
        # Check if the password has at least a number and three special cha
        if any(char.isdigit() for char in password) and len(re.findall(r'[!
            # Check if the length of the password is 10 characters
            if len(password) == 10:
                return "Valid Password"

    return "Invalid Password"

# Example usage:
password_input = input("Enter the password: ")
result = check_password(password_input)
print(result)
```

```
Enter the password:
```

In [ ]:
```
Solve Questions using Lambda, Filter, Map, List Comprehension
```

In [*]:
```
# B Check if the string starts with a particular letter
starts_with_letter = lambda string, letter: string.startswith(letter)
print(starts_with_letter("Python", "P"))  # True

# B Check if the string is numeric
is_numeric = lambda string: string.isnumeric()
print(is_numeric("123"))  # True

# B Sort a list of tuples having fruit names and their quantity
fruits = [("mango", 99), ("orange", 80), ("grapes", 1000)]
sorted_fruits = sorted(fruits, key=lambda x: x[1])
print(sorted_fruits)

# B Find the squares of numbers from 1 to 10
squares = [x**2 for x in range(1, 11)]
print(squares)

# B Find the cube root of numbers from 1 to 10
cube_roots = list(map(lambda x: x**(1/3), range(1, 11)))
print(cube_roots)

# B Check if a given number is even
is_even = lambda num: num % 2 == 0
print(is_even(8))  # True

# B Filter odd numbers from the given list
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
odd_numbers = list(filter(lambda x: x % 2 != 0, numbers))
print(odd_numbers)

# B Sort a list of integers into positive and negative integers lists
integers = [1, 2, 3, 4, 5, 6, -1, -2, -3, -4, -5, 0]
positive_numbers = [num for num in integers if num > 0]
negative_numbers = [num for num in integers if num < 0]
print(positive_numbers, negative_numbers)
```

In [ ]:
```
```