Question 1:

```
list=["Apple","Mango","Banana","Orange","Grapes"]
print(list[0])
print(list[-1])
print(list[1:-1])
```

Question 2:

```
dictionary = {'Alice': 21, 'Bob': 22, 'Charlie': 20, 'David': 23, 'Eve':
19, 'Frank': 24}
print(dictionary['Charlie'])
# Add new entry
dictionary["Jack"] = 67
print(dictionary)
```

Question 3:

```
def cal(lst):
    freq = {}
    for item in lst:
        freq[item] = freq.get(item, 0) + 1

    print("Repetitive elements are:")
    for key, value in freq.items():
        if value > 1:
            print(key)

# Example list
lst = [1, 2, 3, 3, 4, 4, 5, 7, 5, 5]
cal(lst)
```

Question 4:

```python
def group(lst, size):
    result = []
    for i in range(0, len(lst), size):
        result.append(lst[i:i + size])
    return result
numbers = [1, 2, 3, 4, 5, 6, 7]
print(group(numbers, 3))
```

Question 5:

```python
def lensort(strings):
    n = len(strings)
    for i in range(n):
        for j in range(0, n - i - 1):
            if len(strings[j]) > len(strings[j + 1]):

                strings[j], strings[j + 1] = strings[j + 1], strings[j]
    return strings


words = ["apple", "pie", "banana", "kiwi", "orange"]
print(lensort(words))
```

Question 6:

```python
def extsort(files):

    for i in range(len(files)):
        for j in range(i + 1, len(files)):
            ext_i = files[i].split('.')[-1] if '.' in files[i] else ''
            ext_j = files[j].split('.')[-1] if '.' in files[j] else ''
            if ext_i > ext_j:
                files[i], files[j] = files[j], files[i]
    return files


filenames = ["report.doc", "data.csv", "image.png", "notes.txt",
"archive.zip", "script.py"]
print(extsort(filenames))
```

Question 7:

```python
def write_table(file_name, num):
    file = open(file_name, 'w')
    for i in range(1, 11):
        line = str(num) + " x " + str(i) + " = " + str(num * i) + "\n"
        file.write(line)
    file.close()

# Example usage
write_table("table.txt", 556)
```

Question 8:

```python
file = open("text.txt", 'r', encoding='utf-8')
contents = file.read()
contents = ''.join(reversed(contents))
print(contents)
```

Question 9:

```python
with open('text.txt', 'r',encoding='utf-8') as file:
    for line in file:
        print(line.rstrip()[::-1])
```

Question 10:

```python
import sys
import textwrap

def wrap_lines(filename, width):
    try:
        with open(filename, 'r') as file:
            for line in file:
                # Remove trailing newline and wrap line
                wrapped = textwrap.wrap(line.rstrip(), width)
                for wrapped_line in wrapped:
                    print(wrapped_line)
    except FileNotFoundError:
        print(f"Error: File '{filename}' not found.")
    except ValueError:
        print("Error: Width must be a valid integer.")

if __name__ == "__main__":
    if len(sys.argv) != 3:
        print("Usage: python wrap.py <filename> <width>")
    else:
        filename = sys.argv[1]
        try:
            width = int(sys.argv[2])
            wrap_lines(filename, width)
        except ValueError:
            print("Error: Width must be an integer.")
```

Question 11:

```python
def square(n):
    return n * n

numbers = [1, 2, 3, 4]
squared = custom_map(square, numbers)
print(squared)  # Output: [1, 4, 9, 16]
```

Question 12:

```python
def custom_filter(func, iterable):
    return [item for item in iterable if func(item)]

def is_even(n):
    return n % 2 == 0

numbers = [1, 2, 3, 4, 5, 6]
evens = custom_filter(is_even, numbers)
print(evens)
```

Question 13:

```python
def triplet(n):
    result = []
    for a in range(n):
        for b in range(a, n):
            c = a + b
            if c < n:
                result.append((a, b, c))
    return result
```

Question 14:

```python
def parse_csv(filename):
    rows = []
    with open(filename, 'r') as file:
        for line in file:
            # Strip newline and split by comma
            rows.append(line.strip().split(','))
    return rows
```

Question 15:

```python
import string

def mutate(word):
    mutations = set()
    letters = string.ascii_lowercase

    # Insert a character
    for i in range(len(word) + 1):
        for c in letters:
            mutations.add(word[:i] + c + word[i:])

    # Delete a character
    for i in range(len(word)):
        mutations.add(word[:i] + word[i+1:])

    # Replace a character
    for i in range(len(word)):
        for c in letters:
            if c != word[i]:
                mutations.add(word[:i] + c + word[i+1:])

    # Swap two consecutive characters
    for i in range(len(word) - 1):
        swapped = list(word)
        swapped[i], swapped[i+1] = swapped[i+1], swapped[i]
        mutations.add(''.join(swapped))
    return mutations

def nearly_equal(a, b):
    return b in mutate(a)
print(nearly_equal("cat", "cta"))  # True (swap)
print(nearly_equal("cat", "cut"))  # True (replace 'a' with 'u')
print(nearly_equal("cat", "cats")) # True (insert 's')
print(nearly_equal("cat", "at"))   # True (delete 'c')
print(nearly_equal("cat", "dog"))  # False
```

Question 16:

```
from collections import Counter
import os

def count_character_frequency(filename):
    with open(filename, 'r', encoding='utf-8') as file:
        content = file.read()
        frequency = Counter(content)
    return frequency
file_path = "C:/Users/23103212/pYHTON/sixteen.py"
print("Current working directory:", os.getcwd())

# Count and print character frequencies
freq = count_character_frequency(file_path)
for char, count in freq.items():
    print(f"{repr(char)}: {count}")
```

Question 17:

```
from collections import defaultdict

def group_anagrams(words):
    anagram_dict = defaultdict(list)

    for word in words:

        key = ''.join(sorted(word))
        anagram_dict[key].append(word)


    return [group for group in anagram_dict.values() if len(group) > 1]

words = ['eat', 'ate', 'tea', 'hello', 'silent', 'listen', 'enlist',
'world']


anagram_groups = group_anagrams(words)
for group in anagram_groups:
    print(group)
```