```
class ArrayList:

    def __init__(self, size=4, initial_elements=None):
        if size <= 0:
            raise ValueError("Initial capacity must be greater than 0")

        self._capacity = size
        self._count = 0
        self._data = [None] * self._capacity

        if initial_elements:
            for element in initial_elements:
                self.append(element)

    # ------------------------
    # String representation
    # ------------------------
    def __str__(self):
        return "[" + ", ".join(str(self._data[i]) for i in range(self._count)) + "]"

    # ------------------------
    # Size
    # ------------------------
    def __len__(self):
        return self._count

    # ------------------------
    # Empty check
    # ------------------------
```

```
def isEmpty(self):
    return self._count == 0


# ------------------------
# Get element (positive & negative index)
# ------------------------
def getitem(self, index):
    if index < 0:
        index = self._count + index


    if index < 0 or index >= self._count:
        raise IndexError("Index out of range")


    return self._data[index]


# ------------------------
# Contains
# ------------------------
def __contains__(self, element):
    for i in range(self._count):
        if self._data[i] == element:
            return True
    return False


# ------------------------
# Iterator
# ------------------------
def __iter__(self):
    for i in range(self._count):
        yield self._data[i]

# -------------------------
# Resize (~1.125x growth similar idea)
# -------------------------
def _resize(self):
    new_capacity = int(self._capacity * 1.125) + 4
    if new_capacity <= self._capacity:
        new_capacity = self._capacity + 4

    new_data = [None] * new_capacity

    for i in range(self._count):
        new_data[i] = self._data[i]

    self._data = new_data
    self._capacity = new_capacity

# -------------------------
# Append
# -------------------------
def append(self, element):
    if self._count == self._capacity:
        self._resize()

    self._data[self._count] = element
    self._count += 1

# -------------------------
# Insert
# ------------------------
def insert(self, index, element):
    if index < 0 or index > self._count:
        raise IndexError("Index out of range")

    if self._count == self._capacity:
        self._resize()

    for i in range(self._count, index, -1):
        self._data[i] = self._data[i - 1]

    self._data[index] = element
    self._count += 1

# ------------------------
# Delete by element
# ------------------------
def remove(self, element):
    for i in range(self._count):
        if self._data[i] == element:
            for j in range(i, self._count - 1):
                self._data[j] = self._data[j + 1]

            self._data[self._count - 1] = None
            self._count -= 1
            return

    raise ValueError("Element not found")
```

```
# ------------------------
# Delete by index
# ------------------------
def pop(self, index):
    if index < 0 or index >= self._count:
        raise IndexError("Index out of range")

    removed = self._data[index]

    for i in range(index, self._count - 1):
        self._data[i] = self._data[i + 1]

    self._data[self._count - 1] = None
    self._count -= 1

    return removed

# ------------------------
# Clear
# ------------------------
def clear(self):
    self._data = [None] * self._capacity
    self._count = 0
```