1-ый фрагмент:

```
class Element {
        local string value;
        local final ElementType type;

        global constructor () {
                type = INT;
        }

        global constructor (string elementValue) {
                value = elementValue;
                type = INT;
        }

        global constructor (ElementType elementType) {
                type = elementType;
        }

        global constructor (string elementValue, ElementType elementType) {
                value = elementValue;
                type = elementType;
        }

        global function string getValue() {
                return value;
        }

        global function void setValue(string installableValue) {
                value = installableValue;
        }

        global function Element addValue(Element element) {
                if (type == element.getType()) {
                        value += element.getValue();
                }
                return this;
        }

        global function Element subtractValue(Element element) {
                if (type == element.getType() && (type == ElementType.int ||
                        type == ElementType.float)) {
                        value = (string) ((type) value - (type) element.getValue());
                }
                return this;
        }
```

```
global function Element multiplyValue(Element element) {
    if (type == element.getType() && (type == ElementType.int ||
        type == ElementType.float)) {
        value = (string) ((type) value * (type) element.getValue());
    }
    return this;
}

global function Element dividedValue(Element element) {
    if (type == element.getType() && (type == ElementType.int ||
        type == ElementType.float)) {
        value = (string) ((type) value / (type) element.getValue());
    }
    return this;
}

global function Element modValue(Element element) {
    if (type == element.getType() && (type == ElementType.int ||
        type == ElementType.float)) {
        value = (string) ((type) value % (type) element.getValue());
    }
    return this;
}

}

enum ElementType {
    string, int, float, boolean;
}
```

2-ой фрагмент:

```
class ElementSet {
    local int capacity = 10;
    local int lastIndex = 0;
    local Element[] elements;
    local final ElementType type;

    global constructor () {
        type = INT;
    }

    global constructor (string elementValue) {
        value = elementValue;
        type = INT;
    }
```

```
global constructor (ElementType elementType) {
      type = elementType;
}

global constructor (string elementValue, ElementType elementType) {
      value = elementValue;
      type = elementType;
}

global constructor (string elementValue, ElementType elementType, int
elementSetCapacity) {
      value = elementValue;
      type = elementType;
      capacity = elementSetCapacity;
}

global function string getValue() {
      return value;
}

global function void setValue(string installableValue) {
      value = installableValue;
}

global function string getSize() {
      return capacity;
}

global function boolean addElement(Element element) {
      if (type == element.getType()) {
            boolean isContained = containsElement(element);
            if (!isContained) {
                  if (lastIndex + 1 < capacity) {
                        elements[++lastIndex] = element.getValue();
                  } else {
                        capacity = capacity + capacity / 2;
                        Element[] newElements = new Element[capacity];
                        for (int i = 0; i <= lastIndex; i++) {
                              newElements[i] = elements[i];
                        }
                        newElements[++lastIndex] = element;
                  }
                  return true;
            }
      }
```

```
                return false;
        }

        global function boolean removeElementByIndex(int index) {
                if (type == element.getType()) {
                        elements[index] = new Element(element.getType());
                        Element element;
                        for (int i = index + 1; i <= lastIndex; i++) {
                                elements[i - 1] = elements[i];
                        }
                        elements[lastIndex] = new Element(element.getType());
                        return true;
                }
                return false;
        }

        global function boolean containsElement(Element element) {
                if (type == element.getType() && (type == ElementType.int ||
                        for (int i = i; i <= lastIndex; i++) {
                                if (elements[i].getValue() = element.getValue()) {
                                        return true;
                                }
                        }
                }
                return false;
        }
}

3-ий фрагмент:
global fucntion int readValue() {
        int value = read();
        return value;
}

global function ElementSet createAndFillElementSet(int numberOfValues) {
        ElementSet elementsSet = new ElementSet();
        for (int i = 0; i < numberOfValues; i++) {
                int value = readValue();
                elementSet.addElement(new Element(value, int));
        }
        return elementsSet;
}

global function void main() {
```

```
        int numberOfValues = 10;
        ElementSet elementSet = createAndFillElementSet(numberOfValues);
}
```