# Student Grouping System
## Question Class Documentation

# Overview

This document provides interface and implementation documentation for the Question class hierarchy in `survey.py`. The documented classes are `Question`, `MultipleChoiceQuestion`, `NumericQuestion`, `YesNoQuestion`, and `CheckboxQuestion`.

# 1 Interface Documentation

## 1.1 Abstract Class: `Question`

### Purpose

`Question` represents a generic survey question. It defines the public interface that all concrete question types must provide. This class is abstract and is not intended to be instantiated directly.

### Public Attributes

- `id`: A unique integer identifier for the question.
- `text`: The prompt displayed to respondents.

### Representation Invariant

- `len(self.text) > 0`.

### Public Methods

- `__str__() -> str`: Returns a string representation of the question that includes the question text and a description of the possible answers. The exact format is determined by subclasses.
- `validate_answer(answer: Answer) -> bool`: Returns `True` if and only if the provided `Answer` is valid for this question. Validity rules are defined by subclasses.
- `get_similarity(answer1: Answer, answer2: Answer) -> float`: Returns a similarity score between two valid answers. The result is a float between 0.0 and 1.0 inclusive. The meaning of similarity is defined by subclasses.

### Subclass Responsibilities

Any concrete subclass must implement:

- `__str__`
- `validate_answer`
- `get_similarity`

## 1.2 MultipleChoiceQuestion

### Purpose

`MultipleChoiceQuestion` represents a question whose answer must be exactly one element from a predefined list of options.

### Public Attributes

- `id`: A unique integer identifier for the question.
- `text`: The prompt displayed to respondents.

### Public Methods

- `__str__() -> str`: Returns the question text and a description of the possible options.
- `validate_answer(answer: Answer) -> bool`: Returns `True` if and only if `answer.content` is one of the allowed options for this question.
- `get_similarity(answer1: Answer, answer2: Answer) -> float`: Returns 1.0 if the answer contents are equal. Otherwise returns 0.0.

## 1.3 NumericQuestion

### Purpose

`NumericQuestion` represents a question whose answer must be an integer within a specified inclusive range.

### Public Attributes

- `id`: A unique integer identifier for the question.
- `text`: The prompt displayed to respondents.

### Public Methods

- `__str__() -> str`: Returns the question text and a description of the valid numeric range.
- `validate_answer(answer: Answer) -> bool`: Returns `True` if and only if `answer.content` is an integer between the minimum and maximum inclusive.
- `get_similarity(answer1: Answer, answer2: Answer) -> float`: Returns a similarity score based on the numerical distance between two valid answers over the full range of possible answers.

## 1.4 YesNoQuestion

### Purpose

`YesNoQuestion` represents a binary question whose valid answers correspond to `True` (yes) and `False` (no).

### Public Attributes

- `id`: A unique integer identifier for the question.
- `text`: The prompt displayed to respondents.

### Public Methods

`YesNoQuestion` inherits the public behavior of `MultipleChoiceQuestion`. The following public methods are available and behave according to multiple-choice logic over Boolean options:

- `__str__() -> str`
- `validate_answer(answer: Answer) -> bool`
- `get_similarity(answer1: Answer, answer2: Answer) -> float`

## 1.5 CheckboxQuestion

### Purpose

`CheckboxQuestion` represents a question whose answer can be a selection of one or more options from a predefined list of strings.

### Public Attributes

- `id`: A unique integer identifier for the question.
- `text`: The prompt displayed to respondents.

### Public Methods

- `__str__() -> str`: Returns the question text and a description of the available checkbox options.
- `validate_answer(answer: Answer) -> bool`: Returns `True` if and only if `answer.content` is a non-empty list, contains no duplicates, and each element is a valid option string for this question.
- `get_similarity(answer1: Answer, answer2: Answer) -> float`: Returns a similarity score defined as the size of the intersection of selected options divided by the size of the union of selected options. If the union is empty, the similarity is 1.0.

# 2 Implementation Documentation

## 2.1 Question

`__init__(id_, text)`

The initializer stores the provided identifier and text in the public attributes `id` and `text`. The precondition ensures the text is non-empty.

### Abstract Methods

`__str__`, `validate_answer`, and `get_similarity` raise `NotImplementedError` to enforce subclass implementations. This ensures that each concrete question type provides appropriate answer de-

scription, validation, and similarity logic.

## 2.2 MultipleChoiceQuestion

`__init__(id_, text, options)`

The initializer calls the superclass initializer to set `id` and `text`. It then stores the provided list of options in a private attribute used for validation and description of possible answers.

`__str__`

The string representation is formed by combining the question text with a formatted representation of the option list. This provides a readable description of possible answers.

`validate_answer`

The method checks membership of `answer.content` in the stored option list. The result of this membership check is returned.

`get_similarity`

The method compares `answer1.content` and `answer2.content`. If they are equal, it returns 1.0. Otherwise, it returns 0.0.

## 2.3 NumericQuestion

`__init__(id_, text, min_, max_)`

The initializer calls the superclass initializer to set `id` and `text`. It then stores the minimum and maximum bounds in private attributes. These bounds define the set of valid integer answers.

`__str__`

The string representation is formed by combining the question text with a formatted description of the valid inclusive range.

`validate_answer`

The method checks two conditions. First, it checks that the answer content has type `int`. Second, it checks that the integer lies within the inclusive range defined by the stored bounds. The method returns `True` only when both conditions are satisfied.

`get_similarity`

The method computes similarity using normalized distance. It computes the absolute difference between the two integer contents. It then divides this difference by the size of the numeric range, computed as `max - min`. Finally, it subtracts the resulting fraction from 1.0. This yields 1.0 when the answers match and 0.0 when the answers are as far apart as possible within the range.

## 2.4 YesNoQuestion

`__init__(id_, text)`

The initializer delegates to `MultipleChoiceQuestion` by passing a fixed option list containing `True` and `False`. This ensures that validation, similarity, and string formatting follow the multiple-choice logic while restricting valid answers to the Boolean domain.

### Inherited Behavior

`YesNoQuestion` does not override `__str__`, `validate_answer`, or `get_similarity`. The inherited implementations operate over the two Boolean options.

## 2.5 CheckboxQuestion

`__init__(id_, text, options)`

The initializer calls the superclass initializer to set `id` and `text`. It then stores the list of allowed option strings in a private attribute used for validation and output formatting.

`__str__`

The string representation is formed by combining the question text with a formatted representation of the allowed option list.

`validate_answer`

The method validates `answer.content` using a sequence of checks. It checks that the content is a list and that it is non-empty. It checks that there are no duplicates by comparing the list length to the size of the corresponding set. It then iterates through each element and verifies that it is a string and that it appears in the allowed option list. The method returns `True` only if all checks pass.

`get_similarity`

The method converts both answer lists into sets. It computes the union of the two sets to determine the total number of unique selected options. If the union is empty, it returns 1.0. Otherwise, it computes the intersection and returns the ratio of the intersection size to the union size.