Техническая документация к Mask

Обзор:
Класс Mask<N> реализует индексную маску фиксированного размера N, содержащую значения 0 и 1. Класс предоставляет операции для фильтрации и преобразования элементов контейнеров на основе маски. Маска применяется циклически при обработке контейнеров произвольного размера.

Файл Mask.h

Шаблон класса Mask

Приватное поле:
maskData: std::array<bool, N> - массив для хранения значений маски

Публичные методы:
1. Конструктор с std::initializer_list<bool>
2. size() - возвращает размер маски
3. at(size_t index) - доступ к элементу маски с проверкой границ
4. slice() - фильтрация контейнера
5. transform() - преобразование элементов контейнера по маске
6. slice_and_transform() - комбинированная операция фильтрации и преобразования

Файл Mask.cpp

Конструктор:
Mask(std::initializer_list<bool> maskList)

Как работает:
1. Проверяет, что размер переданного списка инициализации равен N
    - Если нет, выбрасывает std::invalid_argument
2. Копирует данные из списка инициализации в массив maskData
3. Проверяет, что все элементы равны 0 или 1
    - Если обнаружено другое значение, выбрасывает исключение

Как сделано:
1. Использует std::copy для копирования данных
2. Выполняет явную проверку каждого элемента через цикл for

Метод size() const
Как работает:
1. Возвращает константу N - размер маски

Как сделано:

1. Просто возвращает N - значение параметра шаблона

Метод at(size_t index) const

Как работает:
1. Проверяет, что индекс находится в диапазоне [0, N-1]
   - Если нет, выбрасывает std::out_of_range
2. Возвращает значение элемента маски по указанному индексу

Как сделано:
1. Использует стандартную проверку границ с выбрасыванием исключения

Файл Mask.tpp

Метод slice(Container& container) const
Удаляет из контейнера все элементы, для которых в маске стоит 0

Как работает:
Создает временный вектор result для хранения отфильтрованных элементов
1. Проходит по всем элементам контейнера:
   - Использует оператор % для циклического применения маски
   - Если maskData[i % N] == true (1), копирует элемент в result
2. Очищает исходный контейнер
3. Копирует все элементы из result обратно в контейнер
4. Изменяет исходный контейнер "на месте"
5. Сохраняет порядок элементов, удовлетворяющих маске
6. Работает с любым контейнером, имеющим методы at(), size(), clear(), push_back()

Метод transform(const Container& container, Func func) const
Создает новый контейнер, применяя функцию преобразования только к элементам, отмеченным маской

Как работает:
Создает пустой контейнер result того же типа
1. Проходит по всем элементам исходного контейнера:
   - Если элемент отмечен маской (1), добавляет func(element) в результат
   - Иначе добавляет исходный элемент без изменений
2. Возвращает новый контейнер
3. Возвращает новый контейнер
4. Не изменяет исходный контейнера
5. Возвращает контейнер того же размера, что и исходный
6. Сохраняет все элементы, но преобразует только отмеченные маской

Метод slice_and_transform(const Container& container, Func func) const

Фильтрует контейнер по маске и применяет функцию преобразования к оставшимся элементам

Как работает:

1. Определяет типы:
   - ValueType - тип элементов исходного контейнера
   - ResultType - тип результата функции преобразования, определяется через decltype
2. Создает std::vector<ResultType> для результатов
3. Проходит по всем элементам контейнера:
   - Если элемент отмечан маской (1), добавляет func(element) в результат
4. Возвращает вектор с преобразованными элементами
5. Возвращает только отфильтрованные и преобразованные элементы
6. Размер результата равен количеству единиц в маске с учетом циклического применения
7. Всегда возвращает std::vector, а не исходный тип контейнера