**Пример №1 *(построение дерева разбиений для программы поиска максимального элемента в массиве)***

В таблице 1 представлены функциональные требования к программе.

Необходимо для этой программы выполнить следующие шаги тестирования способами **разбиения по эквивалентности и анализа граничных значений**:

а) определить предусловия, постусловия, специальные требования из практики разработчика, ребра классов эквивалентности (опираясь, в том числе, на требования к разбиениям в соответствии с таблицей 1);

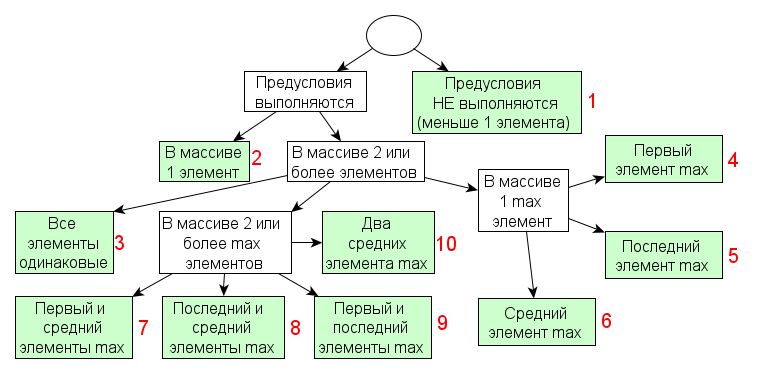
б) построить дерево разбиений;

в) описать входные данные и ожидаемые результаты тест-кейсов (во входных данных и ожидаемых результатах привести примеры конкретных массивов).

**Таблица 1.** Функциональные требования к программе и требования к разбиениям

|  |  |
| --- | --- |
| **Функциональные требования к программе** | **Требования к разбиениям** |
| 1) Ввод пользователем количества элементов одномерного массива целых чисел и значений элементов.  *Ввод числа элементов, равного 0, считается нарушением предусловий и должен приводить к аварийному сообщению.*  2) Поиск в массиве ***максимального*** элемента и вывод на экран значения этого элемента и его индекса (индексы начинаются с нуля). Если таких элементов несколько, то вывод значения и индекса ***первого*** из этих элементов. | Рассмотреть случаи, когда:  1) массив состоит из одного элемента;  2) массив состоит из одинаковых элементов;  3) в массиве один максимальный элемент;  4) в массиве несколько максимальных элементов.  Также рассмотреть случаи, когда в исходном массиве:  1) первый элемент максимальный (но при этом последний не максимальный);  2) последний элемент максимальный (но при этом первый не максимальный);  3) и первый, и последний элементы максимальные;  4) ни первый, ни последний элемент не максимальные. |

Построим дерево разбиений:



Это дерево имеет 10 листьев. Каждый лист задает отдельный тест-кейс. В таблице 2 представлены тест-кейсы (ТК), основанные на проведенных разбиениях (нумерация тест-кейсов соответствует нумерации листьев).

**Таблица 2.** Входные данные и ожидаемые результаты тест-кейсов

|  |  |  |
| --- | --- | --- |
| **№ ТК** | **Входные данные** | **Ожидаемые результаты** |
| **ТК1** | Введено *N*≤0, где *N* - число элементов массива. | Сообщение, что массив не содержит элементов. |
| **ТК2** | Пример массива из одного элемента:  15 | i\_max = 0 *(индекс первого максимального элемента)*  max = 15 *(значение первого максимального элемента)* |
| **ТК3** | Пример массива:  25, 25, 25 | i\_max = 0  max = 25 |
| **ТК4** | Пример массива:  17, 12, 10 | i\_max = 0  max = 17 |
| **ТК5** | Пример массива:  12, 10, 17 | i\_max = 2  max = 17 |
| **ТК6** | Пример массива:  12, 19, 17 | i\_max = 1  max = 19 |
| **ТК7** | Пример массива:  18, 11, 18, 15 | i\_max = 0  max = 18 |
| **ТК8** | Пример массива:  18, 23, 18, 23 | i\_max = 1  max = 23 |
| **ТК9** | Пример массива:  35, 23, 18, 35 | i\_max = 0  max = 35 |
| **ТК10** | Пример массива:  13, 27, 27, 9 | i\_max = 1  max = 27 |

**Пример №2 *(построение дерева разбиений для программы сортировки массива по возрастанию)***

В таблице 3 представлены функциональные требования к программе.

Необходимо для этой программы выполнить следующие шаги тестирования способами **разбиения по эквивалентности и анализа граничных значений**:

а) определить предусловия, постусловия, специальные требования из практики разработчика, ребра классов эквивалентности (опираясь, в том числе, на требования к разбиениям в соответствии с таблицей 3);

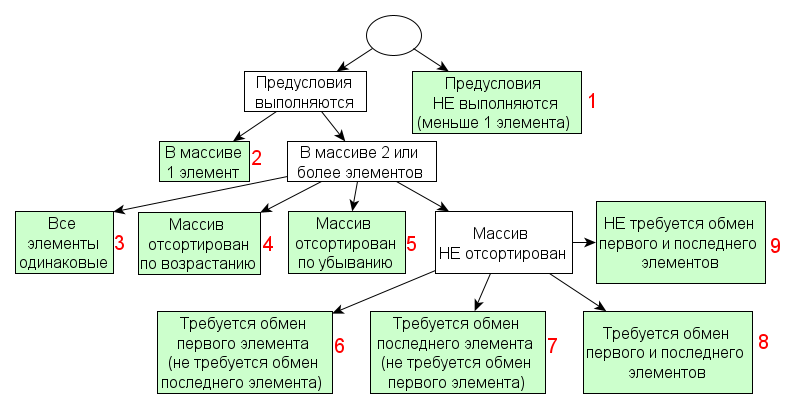
б) построить дерево разбиений;

в) описать входные данные и ожидаемые результаты тест-кейсов (во входных данных и ожидаемых результатах привести примеры конкретных массивов).

**Таблица 3.** Функциональные требования к программе и требования к разбиениям

|  |  |
| --- | --- |
| **Функциональные требования к программе** | **Требования к разбиениям** |
| 1) Ввод пользователем количества элементов одномерного массива целых чисел и значений элементов.  *Ввод числа элементов, равного 0, считается нарушением предусловий и должен приводить к аварийному сообщению.*  2) ***Сортировка массива по возрастанию*** и вывод на экран отсортированного массива. | Рассмотреть случаи, когда:  1) массив состоит из одного элемента;  2) массив состоит из одинаковых элементов;  3) не отсортирован;  4) отсортирован по возрастанию;  5) отсортирован по убыванию.  Также рассмотреть случаи, когда в исходном массиве:  1) требуется обмен первого элемента (но при этом не требуется обмен последнего);  2) требуется обмен последнего элемента (но при этом не требуется обмен первого);  3) требуется обмен и первого, и последнего элементов;  4) не требуется обмен первого и последнего элементов. |

Построим дерево разбиений:



Это дерево имеет 9 листьев. Каждый лист задает отдельный тест-кейс. В таблице 4 представлены тест-кейсы (ТК), основанные на проведенных разбиениях (нумерация тест-кейсов соответствует нумерации листьев).

**Таблица 4.** Входные данные и ожидаемые результаты тест-кейсов

|  |  |  |
| --- | --- | --- |
| **№ ТК** | **Входные данные** | **Ожидаемые результаты** |
| **ТК1** | Введено *N*≤0, где *N* - число элементов массива. | Сообщение, что массив не содержит элементов. |
| **ТК2** | Пример массива из одного элемента:  15 | Отсортированный по возрастанию массив:  15 |
| **ТК3** | Пример массива:  25, 25, 25 | Отсортированный по возрастанию массив:  25, 25, 25 |
| **ТК4** | Пример массива:  15, 25, 27, 35 | Отсортированный по возрастанию массив:  15, 25, 27, 35 |
| **ТК5** | Пример массива:  35, 27, 25, 15 | Отсортированный по возрастанию массив:  15, 25, 27, 35 |
| **ТК6** | Пример массива:  25, 15, 27, 35 | Отсортированный по возрастанию массив:  15, 25, 27, 35 |
| **ТК7** | Пример массива:  15, 25, 35, 27 | Отсортированный по возрастанию массив:  15, 25, 27, 35 |
| **ТК8** | Пример массива:  35, 25, 27, 15 | Отсортированный по возрастанию массив:  15, 25, 27, 35 |
| **ТК9** | Пример массива:  15, 27, 25, 35 | Отсортированный по возрастанию массив:  15, 25, 27, 35 |