## 2.4 Особенности реализации контролирующего режима

Основным назначением ОТП «Линейные списки» является обучение пошаговым алгоритмам работы со списковыми структурами. Возможность пошагового выполнения алгоритмов достигается путем использования потоков. Выполняется это следующим образом. Пользователь нажимает на кнопку «Добавить после», обработчик этой функции запускает поток на выполнение. После того как выполнится первое действие алгоритма, поток будет приостановлен. Возобновится поток нажатием пользователя на кнопку «Next».

Опишем краткий набор функций, используемых во время работы с потоками в приложении.

Создание потока выполняется функцией:

function BeginThread(SecurityAttributes: Pointer; StackSize: LongWord; ThreadFunc: TThreadFunc; Parameter: Pointer; CreationFlags: LongWord; var ThreadId: TThreadID): THandle;

Основной параметр для данной функции это «ThreadFunc» - это функция которая будет выполняться в отдельном потоке. «ThreadId» возвращает уникальный идентификатор для нового потока. Параметр «Parameter» в виде указателя передает в функцию-поток входные агрументы. После инициализации потока функцию возвращает дескриптор потока Windows.

Приостановка потока выполняется функцией:

SuspendThread(hTread: Integer), где hTread – дескриптор потока – результат функции BeginThread().

Возобновляется поток аналогично приостановке, только выполнением функции ResumeThread(hTread: Integer).

Таким образом, все алгоритмы для работы со списковой структурой разделены функциями остановки потока, после выполнения одного шага.

Помимо возможности управления алгоритмом, также необходимо предусмотреть возможность формирование вопросов в зависимости от используемого алгоритма, а также знать непосредственно правильный ответ на поставленный вопрос.

Для решения этой задачи использовались специальные структуры TDictionary. TDictionary представляет собой структуру, в которой информация хранится в наборе пар ключ – значение.

Этот класс предоставляет сопоставление из набора ключей в набор значений. Когда создается экземпляр класса TDictionary, необходимо указать типы данных в которых будут храниться ключи и значения. Такая особенность структуры, когда заранее не известны типы данных которые будут в ней использоваться, реализуется с использованием дженериков (Generics) – универсальных типов данных.

Generics, или как их иногда называют обобщенными параметрами –это набор инструментов абстракции, которые позволяют развязать алгоритм (например, процедуру или функцию) или структуру данных (например, класс, интерфейс или запись) из одного или нескольких конкретных типов, которые использует алгоритм или структура данных.

Метод или тип данных, который использует другие типы в его определении, можно сделать более общим, заменив один или несколько конкретных типов параметрами типа. Это позволяет сделать процедуру более общей, заменив экземпляры литеральную константы в своем теле процедуры именем параметра и добавив параметр в список параметров процедуры.

Таким образом в структуре TDictionary можно хранить ключи и значение различных типов данных. В модуле «UQuestions» созданы 2 экземпляра структуры TDictionary<integer, string> с названиями \_hashAdd и \_hashDelete. В данных структурах по ключам хранятся различные аврианты ответов, в зависимости от нужной структуры.