


Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий, математики и механики

Учебный курс

Методы программирования - 2

Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий, математики и механики

Учебный курс:

Методы программирования - 2

Практическая работа 7:

Разработка общего представления линейного списка

Гергель В.П., профессор ,
директор института ИТММ

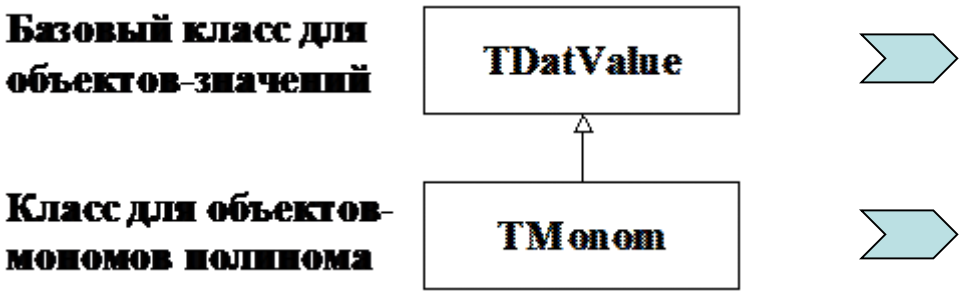
Содержание

Разработка общего представления линейного списка

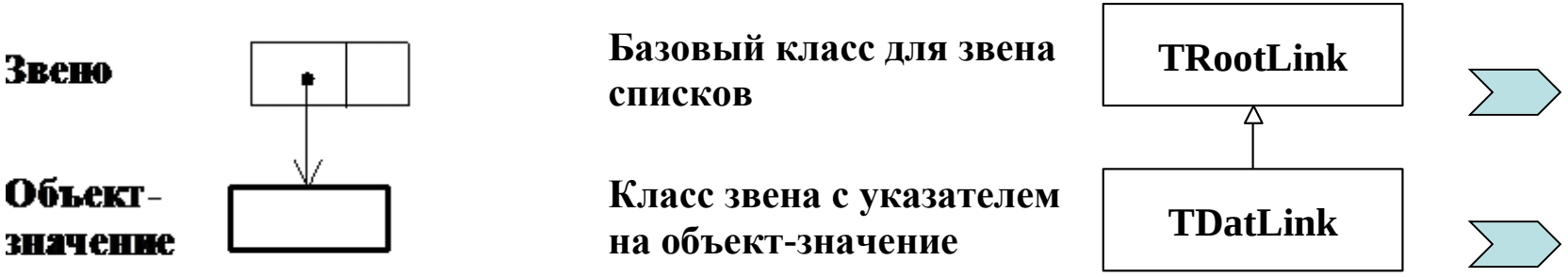
- Организация хранения значений разного типа
- Проектирование структуры списка
- Операции для работы со списком
  - Информационные методы
  - Методы доступа
  - Навигация по списку (итератор)
  - Вставка и удаление звеньев
- Обеспечение удобного интерфейса
  - Надежность преобразования типов
  - Управление временем жизни значений-объектов
  - Обеспечение удобного API

1. Организация хранения в списках значений разного типа

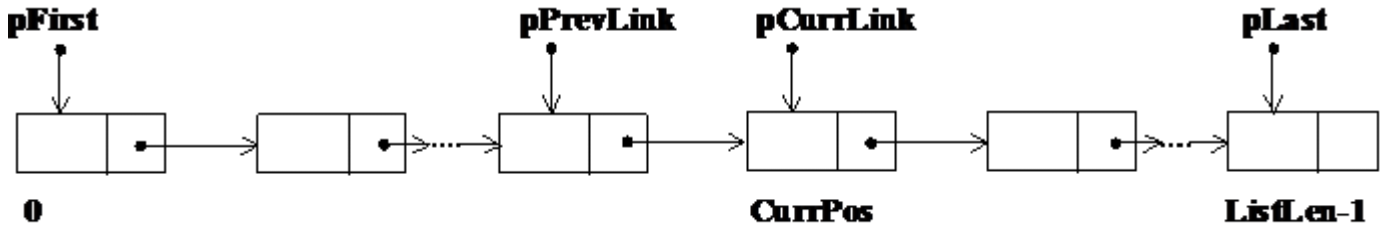
- Представим значений в виде объектов классов, являющиеся производными из одного общего базового класса



- В поле значения звена списка будем размещать указатель на объект значение



2. Проектирование структуры списка



- **pFirst** – указатель на первое звено списка
- **pLast** – указатель на последнее звено списка
- **pCurrLink** – указатель на текущее звено списка
- **pPrevLink** – указатель на звено, предшествующее текущему
- **CurrPos** – номер текущего звена
- **ListLen** – количество звеньев в списке

☑ Для повышения общности схемы реализации будем использовать вместо величины NULL константу **pStop** для фиксации ситуаций, в которых указатель не содержит адрес какого-либо звена списка

3. Операции для работы со списком…

- Информационные методы

  - IsEmpty – проверить, не является ли список пустым
  - GetListLength – получить количество звеньев списка

- Методы доступа к значениям в списке

  - GetDatValue – получить указатель на значение из звена списка (обращение возможно только в первому (FIRST), текущему (CURRENT) или последнему звеньям списка (LAST); желаемый вариант доступа задается через параметр метода)

3. Операции для работы со списком…

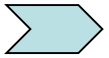
- Методы навигации по списку (***итератор***)

  - Reset – установить текущую позицию на первое звено
  - GoNext – переместить текущую позицию на звено вправо
  - IsListEnded – проверка завершения списка (под ситуацией завершения списка понимается состояние после применения GoNext для текущей позиции, установленной на последнем звене списка, т.е. когда pPrevLink=pLast, pCurrLink=pStop)
  - GetCurrentPos – получить номер текущего звена
  - SetCurrentPos – установить текущую позицию на звено с заданным номером (прямой доступ к звеньям !?)

3. Операции для работы со списком…

- Вставка звеньев 
  - InsFirst – вставить звено перед первым звеном списка
  - InsLast – вставить звено после последнего звена
  - InsCurrent – вставить звено перед текущим звеном списка
- ☑ При выполнении операций вставки звеньев следует учитывать, что список может быть пустым
- ☑ После выполнения вставки необходимо обеспечить корректность значений указателей первого, текущего и последнего звеньев списка
- ☑ При корректировке указателей следует учитывать возможность различного положения текущей позиции списка
  - текущая позиция является первым звеном (pCurrLink=pFirst),
  - текущая позиция является вторым звеном (pPrevLink=pFirst),
  - текущая позиция находится внутри списка,
  - текущая позиция является последним звеном (pCurrLink=pLast),
  - текущая позиция выходит за пределы списка (pPrevLink=pLast)

3. Операции для работы со списком

- Удаление звеньев 
  - DelFirst – удалить первое звено списка
  - DelCurrent – удалить текущее звено
  - DelList – удалить список

Пример: программа, приложение

4. Обеспечение удобного интерфейса…

- Надежность преобразование типа
  - ☑ Указатель производного типа может автоматически приводиться к указателю на базовый тип
  - ☑ Для обратного приведения – от базового типа к производному – необходимо явное преобразования типа

```
PTDatValue pValue;
PTMonom pMonom;
pMonom = PTMonom(pValue);
```

⇒ такое преобразование типа не является безопасным

  - ☑ Преобразование и использованием информацией о типе времени исполнения

```
pMonom = dynamic_cast<PTMonom>(pValue);
```

⇒ преобразование типа выполняется только если тип находится выше по иерархии наследования от объекта, указываемого pVal (иначе pMonom=NULL)

4. Обеспечение удобного интерфейса…

↳ Владение (создание и удаление) значениями

☑ Использование нескольких указателей на объект усложняет контроль за его временем жизни

☑ Возможное решение – передача объектов *по значению*

– при записи в список указателя на объект в списке запоминается указатель на копию объекта-значения,

– при получении указателя из списка создается еще одна копия объекта-значения и как результат передается указатель на новую созданную копию

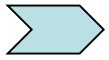
⇒ Возможность создания копий обеспечивает метод GetCopy

4. Обеспечение удобного интерфейса

Обеспечение удобного API

☑ Работа с объектами является более удобным по сравнению с использованием указателей

⇒ Все перечисленные моменты (безопасное преобразование типов, создание копий, использование объектов) могут быть учтены при помощи создания шаблона класса-переходника (*proxy*) TList к классу TDatList



Пример: программа

Заключение

- Организация хранения объектов-значений разных типов
- Проектирование структуры списка
- Операции для работы со списком
- Обеспечения удобного интерфейса (классы-переходники)

Вопросы для обсуждения

- Организация навигации по структуре данных (итераторы)
- Унификация работы с данными (классы-оболочки)

Темы занятий для самостоятельной работы

- Расширение набора операций со списком (поиск по значению или условию, совмещение итератора с операциями обработки,…)
- Разработка двунаправленного списка (наследование)

Следующая тема

- Общая характеристика стандартной библиотеки шаблонов)

Контакты

Нижегородский государственный университет им. Н.И. Лобачевского (www.unn.ru)

Институт информационных технологий, математики и механики (www.itmm.unn.ru)

603950, Нижний Новгород, пр. Гагарина, 23,
р.т.: (831) 462-33-56,

Гергель Виктор Павлович

(http://www.software.unn.ru/?dir=*17*)

E-mail: gergel@unn.ru