Лабораторная работа №30

Тема. Объявление класса, создание экземпляров класса.

Цель: Освоить на практике создание пользовательских типов; выяснить назначение и принципы работы конструкторов и деструкторов, а также экспериментально определить ситуации когда они вызываются; выяснить влияние спецификатора доступа на доступность полей и методов внутри и извне класса.

Индивидуальное задание

1. Создать класс "Студент" содержащую следующие поля:
   - имя студента
   - отчество студента
   - фамилию студента
   - год рождения
   - группа
2. Определить конструктор для инициализации полей по умолчанию. Определить конструктор с параметрами и деструктор. Написать тестовый пример.
3. Добавить в начале класса спецификатор доступа private. Запустить программу. Какие возникли проблемы? В каких именно местах программы и к чему запрещен доступ? Почему? Как исправить возникшее положение?
4. Написать интерфейсные функции доступа к полям класса (получить/задать значение поля), вынеся поля в секцию private, разрешив доступ к конструкторам, деструкторам и интерфейсным функциям.
5. Провести те же действия, что и в задании 4, но изменив private на protected. В чем разница между ними?
6. Внести в конструкторы и деструктор выдачу сообщений на экран о том, какая функция была вызвана. В начале и конце основной программы поставить выдачу сообщений о начале и конце программы. Выяснить когда происходит вызовов конструкторов и деструкторов.
7. Добавить в класс свойства для инкапсуляции полей.

Контрольные вопросы

1. Назначение конструкторов и деструкторов.
2. Когда вызываются конструкторы и деструкторы. Почему сделано таким образом?
3. В чем отличия между спецификаторами доступа private, protected, public если класс не наследуется?
4. Зачем нужны интерфейсные функции?
5. Назначения и принцип функционирования свойств.

Лабораторная работа №31

Тема. Создание наследованного класса.

Цель: Создание простой иерархии классов и изучение методов инициализации объектов производных классов; исследование конструкции объектов производных классов; исследование уровней защищенности унаследованных компонент базового класса в объектах производного класса; исследование методов доступа к одноименным функциям базового и производных классов.

Индивидуальное задание

1.Создать базовый класс Base, в котором описать в разделе public поле a типа integer, в разделе protected поле b типа char, в разделе private поле c типа string. Написать конструктор, инициализирующий поля a, b и c тремя задаваемыми значениями и конструктор по умолчанию.

2. Создать в классе Base два разных открытых метода Test с различными параметрами (изменяющий поля класса). Проверить работу обоих методов. Какое ключевое слово пришлось указать после прототипов методов?

3. Создать класс Derived, производный от класса Base, в котором в разделе private описано поле d типа double. В классе Derived создать конструктор без параметров и конструктор с четырьмя параметрами для инициализации всех полей объекта.

4. В основной программе описать объект класса Derived с инициализацией конструктором по умолчанию и откомпилировать программу. Если есть проблемы, то устранить их. Вывести размеры типов Base и Derived и объяснить результаты.

5. Попробовать вызвать методы Test для объекта класса Derived. Прокомментировать результат.

6. Описать объект класса Derived, создаваемый через конструктор с параметрами. Продемонстрировать, инициализацию каких полей, унаследованных от класса Base, можно выполнять с помощью присваивания непосредственно в конструкторе класса Derived. Какие поля класса Derived обязательно нужно инициализировать с помощью конструктора класса Base? Для исследования можно вносить необходимые изменения в конструкторы классов Base и Derived.

7. Дополнить конструкторы всех классов выдачей сообщений о том для какого именно класса вызывается конструктор. Добавить деструкторы, выводящие подобную информацию. Посмотреть в какой последовательности вызываются конструкторы и деструкторы для объектов класса Derived.

8. Создать метод Show в классе Derived таким образом, чтобы выводились адреса и значения всех полей объекта. К каким полям, унаследованным от класса Base, нет доступа? Для снятия проблемы добавить в классе Base необходимые интерфейсные функции.

9. Создав объект класса Derived, исследовать размещение полей в памяти. В отчете привести схематическую структуру объекта.

10.Описать класс VeryDerived, производный от класса Derived и не имеющий новых полей. В классе описать конструктор со всеми необходимыми параметрами (сколько их нужно?).

11.В класс VeryDerived добавить общедоступный метод Modify, который модифицирует значения полей, унаследованных от базового класса по какому-либо

математическому закону (возводит в квадрат, увеличивает на заданное число, переставляет буквы в строке или еще каким образом).

12.На глобальном уровне и в классах Base и Derived описать подпрограммы (методы) Id, которые сообщают о своей принадлежности к классу или глобальному уровню. В метод Modify класса VeryDerived добавить вызовы всех трех функций . В каких разделах классов Baseи Derivedнужно описать методы Id, чтобы они были доступны в классе VeryDerived? Проверить работу программы, вызвав Modify для какого-либо объекта класса VeryDerived.

13.Оставить в методе Modify класса VeryDerived только один вызов в виде Id; и проверить работу программы в следующих вариантах:

- В начале метод Id определен в классах Derived, Baseи на глобальном уровне;
- Убираем описание Id из класса Derived;
- Убираем его из классов Derived и Base.

Как в каждом случае это отражается на работе программы?

Контрольные вопросы

1. Как действует наследование?
2. Каким образом вызываются конструкторы (деструкторы) для унаследованного объекта?
3. Как спецификаторы доступа действуют при наследовании?
4. Как происходит поиск вызываемой подпрограммы (метода) в случае наследования?
5. Как располагаются поля объекта в памяти при наследовании?
6. Каким образом перегружаются методы в производных классах?
7. Каким образом перегружаются методы в одном классе?

Лабораторная работа №32

Тема. Перегрузка методов

Цель: Исследовать и закрепить на практике использование механизма виртуальных методов ; исследовать изменения в структуре объекта при наличии виртуальных методов; исследовать поведение объекта при вызове обычных и виртуальных методов; закрепление материала по наследованию и абстрактным классам.

Индивидуальное задание

1. Создать класс Figure и Rectangle, производный от Figure. Реализовать в них методы Draw, выводящие информацию для какого класса они вызываются.

2. Описать два указателя F:Figure, R:Rectangle. Инициализировать их объектами соответствующих классов. Вызывать методы Draw для каждого объекта.

3. Уничтожить объекты, попробовать создать объекты созданных классов, инициализировав F объектом класса Rectangle, а R – Figure. Вызывать методы Draw для каждого из объектов. Что произошло при компиляции и как от этого избавиться? Проанализировать результат вызова методов. Каким образом можно заставить правильно вызвать метод для объекта класса Rectangle, заданного через указатель на Figure?

4. Объявить методы Draw виртуальными в обоих классах, обеспечив корректное переопределение метода в классе Rectangle. Проделать действия из пунктов 2 и 3. Объявить результаты.

5. Сделать класс Figure абстрактным, объявив метод Draw чисто виртуальным (абстрактным), не удаляя его реализации из кода. Откомпилировать программу, прокомментировать результат.

6. Удалить реализацию метода Figure. Draw, откомпилировать программу. Прокомментировать результат.

7. Дополнить метод Draw, возложив на него обязанности вывода информации о типе объекта. Внести в класс Figure открытый абстрактный метод function Area:double;, вычисляющий площадь фигуры. Откомпилировать программу, прокомментировать результат. Исправить программу, чтобы компилировалась и корректно выполнялась.

8. Реализовать корректно методы Draw и Area в классе Rectangle. Добавить в иерархию класс Circle, производный от Figureи класс Square, производный от Rectangle. Переопределять методы только в том случае, если будут ошибки компиляции. В основной части программы создать экземпляры всех классов (от Figure до Square) и вызвать для них метод Draw. Прокомментировать результаты и если необходимо исправить ошибки.

9. Создать в основной программе произвольный набор конкретных фигур с помощью открытого массива.

10.Для каждого типа фигур вычислить и вывести на экран размер одного объекта, а также адрес этого объекта и адреса его полей данных. Проанализировать результаты и дать им объяснения. Составить схему размещения объекта в памяти.

11.Из адресов построенных фигур создать массив указателей на базовый класс в виде открытого массива. Организовать цикл, в котором выводится (с помощью виртуальных методов) информация о каждой фигуре из массива. Объяснить результат работы программы.

12.“Отобрать” у метода Draw класса Rectangle его “виртуальность” (перегрузить), откомпилировать программу, прокомментировать результат.

13.Изменить тип методов на dynamic. Откомпилировать, объяснить результат.

14.Изменить тип методов на static(по умолчанию). Откомпилировать, объяснить результат.

15.Вернуть все обратно (виртуальные методы). Разнести все классы по отдельным библиотекам, назвав их соответственно. Имена классов изменить на соответствующие иерархии Delphi (начинать с буквы T). Отладить программу.

16.Удалить из реализации Rectangle перегрузку Draw и создание экземпляров этого класса, запустить программу. Прокомментировать результат.

Контрольные вопросы

1. Чем отличаются статические, динамические и виртуальные методы?
2. Что такое абстрактные классы?
3. Почему нельзя создать экземпляр абстрактного класса?
4. Когда используются динамические методы, а когда виртуальные?
5. Что происходит, если не перегружаются виртуальные методы, не перегруженные в базовом классе?
6. Как оформляются классы? Для чего необходимо вынесение класса в отдельный модуль?