МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ
НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ УКРАИНЫ
«КИЕВСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ»

ПОЛИМОРФИЗМ С++

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к лабораторной работе № 3

по дисциплине «ООП»

Киев 2016

СОДЕРЖАНИЕ

1 ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ

Цель работы – изучить основные концепции объектно-ориентированного программирования. Изучить особенности виртуальных функций, абстрактных классов и полиморфизма.

2 ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Полиморфизм – свойство кода C++ вести себя по-разному в зависимости от ситуации, возникающей в момент выполнения; это возможность объектов разных классов, связанных наследованием, реагировать различным образом при обращении к одной и той же функции-элементу с одинаковым именем и разным определением. Возникает проблема распознавания функций, решаемая при помощи раннего (статического) связывания или позднего (динамического).

Раннее связывание происходит на этапе компиляции, путём вызова фиксированных идентификаторов функций компилятором, замены идентификаторов данными (физическими адресами) компоновщиком задач.

Позднее связывание – при выполнении программы, выбор нужной функции – самой программой. Возможно для объектов связанных иерархией и реализуется через механизм виртуальных функций. Преимущество позднего связывания – гибкость кода, простота изменения программы; недостаток – уменьшение скорости обработки (построение виртуальных таблиц, определение виртуальных функций).

Рассмотрим следующую задачу: есть класс Shape и несколько его наследников – Circl (круг), Rectangle (прямоугольник), Square (квадрат) и т.п. Каждый из классов-наследников имеет свою собственную функцию Draw, рисующую соответствующую фигуру. Выдвигаем условие: работать с данными фигурами как с объектами базового класса Shape, чтобы программа сама понимала что это за объект и как его рисовать. Например:

```
Shape *ShapeArray[10];
ShapeArray[0] = new Circl;
ShapeArray[1] = new Rectangle;
ShapeArray[2] = new Square;
…
for (int i=0; i<10;i++)
  ShapeArray[i]->Draw();
```


Рассмотренный ранее механизм наследования такую задачу решить не может; поскольку ко всем объектам мы обращаемся через тип базового класса Shape, то только функция этого класса и будет вызываться.

Данная задача может быть решена при помощи механизма виртуальных функций.

2.1. Виртуальные функции и абстрактные классы

Виртуальная функция – функция, объявляемая в базовом классе и переопределяемая в производных. Производные классы могут иметь свою собственную реализацию виртуальной функции. При этом функция, переопределяемая в производном классе, должна иметь тот же список аргументов и тип возвращаемого значения, что и виртуальная функция базового класса. Виртуальная функция объявляется и определяется с помощью спецификатора virtual. Формат определения виртуальной функции:

```
virtual тип_возвращаемого значения имя_функции(параметры){тело}
```

Для рассмотренного ранее примера будем иметь:

```
virtual void draw();
```

Если функция однажды была объявлена виртуальной, она остаётся виртуальной и во всех наследниках. Дополним класс Shape функцией, определяющей площадь фигуры, тогда будем иметь:

```
virtual void Shape::area(){return 0;}
void Circl::area(){return 3.14*r*r;}
```

Применяют виртуальные функции:

- в базовых классах (на верхнем уровне иерархии), с последующим переопределением в классах-наследниках;
- как функции, описывающие атрибуты класса и зависящие от структуры и типа класса;
- функции ввода/вывода;
- функции, определяемые специфическими классами.

Для каждого класса с виртуальной функцией строится таблица, содержащая адреса виртуальной функции всех объектов, и определяется

указатель на эту функцию. Распределение оперативной памяти между элементами-данными и указателями на таблицу виртуальных функций осуществляется таким образом, что указатель находится в конце списка переменных первого базового класса. Инициализация указателя на таблицу виртуальных функций осуществляется с помощью специального кода, генерируемого компилятором и вставляемого в начало кода базового конструктора. Для конкретной иерархии объектов адрес виртуальной функции имеет одно и то же смещение в таблице. Адрес виртуальной функции определяется как значение указателя на виртуальную таблицу и смещение.

При вызове виртуальной функции компилятор генерирует код, определяющий указатель на таблицу виртуальных функций. По найденному указателю на виртуальную таблицу определяется указатель на функцию (с учётом смещения).

Ограничения при использовании виртуальных функций:

- конструктор не может быть виртуальным;
- переопределение виртуальной функции с иным перечнем параметров и изменением типа возвращаемого значения невозможно;
- виртуальный механизм можно выключить, используя при вызове функции оператор разрешения видимости ::.

Иногда в базовом классе определяют *чистую виртуальную функцию* – функция, для которой не указана реализация. Для того, чтобы определить такую функцию, достаточно указать, что её тело равно нулю:

```
virtual void area()=0;
virtual void Draw()=0;
```

В нашем примере именно так целесообразно объявить данные функции, поскольку не ясно, как можно нарисовать абстрактную фигуру и вычислить её площадь. Чистая виртуальная функция должна быть переопределена во всех производных классах.

Класс, в котором имеется хотя бы одна чистая виртуальная функция, называется абстрактным классом. Назначение абстрактного класса – обеспечить интерфейс с другими классами. Правила использования:

- абстрактным может быть только базовый класс. Чисто виртуальная функция, которая не переопределяется в производном классе, остаётся чистой виртуальной, поэтому такой производный класс (как и базовый) остаётся абстрактным;
- не существует объектов абстрактного класса;
- не допускается непосредственный вызов чистой виртуальной функции;
- нельзя использовать в качестве параметра и типа возвращаемого значения.

Если класс содержит виртуальные функции, то желательно создавать виртуальные деструкторы. Это приведёт к тому, что все деструкторы производных классов станут виртуальными. В этом случае, если объект в иерархии объектов удаляется явным вызовом оператора delete, который применён к указателю базового класса на объект производного класса, то будет вызван деструктор соответствующего класса.

В C++ абстрактные классы позволяют создавать так называемые интерфейсы.

Интерфейс – программная/синтаксическая структура, определяющая отношение между объектами, которые разделяют определенное поведенческое множество и не связаны никак иначе. При проектировании классов, разработка интерфейса тождественна разработке спецификации (множества методов, который каждый класс, использующий интерфейс *должен* реализовывать).

3 ЗАДАНИЯ

Разработать пошаговый алгоритм решения задачи. Разработать UML диаграмму классов. Выполнить программную реализацию задания согласно варианту. Прототипы классов должны содержаться в отдельном *.h-файле. В программе обязательно предусмотреть вывод информации об исполнителе работы (ФИО), номере варианта, выбранном уровне сложности и задании. Предусмотреть возможность повторного запуска программы (запрос о желании выйти из программы, или продолжить работу). Если программная реализация предполагает решение нескольких подзадач, разработать меню для демонстрации каждой из них. Ключевые моменты программы обязательно должны содержать комментарии.

Уровень А (1 балл)

1. Спроектировать иерархию классов: класс **геометрические фигуры** и его наследники: **квадрат, прямоугольник, треугольник**. Определить в базовом классе и переопределить в наследниках методы вычисления площади и периметра фигуры. Элементы-данные фигур объявляются в базовом классе, а инициализируются в наследниках (элементы данные: стороны фигур).

2. Спроектировать иерархию классов: класс **числа** и его наследники: **целое число, вещественное число, комплексное число**. Определить в базовом классе и переопределить в наследниках методы сложения, вычитания, умножения и деления чисел. Элементы-данные чисел объявляются в базовом классе, а инициализируются в наследниках (элементы данные: два операнда).

3. Спроектировать иерархию классов: класс **треугольник** и его наследники: **прямоугольный треугольник, равнобедренный треугольник, равносторонний треугольник**. Определить в базовом классе и переопределить в наследниках методы вычисления площади и периметра треугольника каждого типа. Элементы-данные фигур объявляются в базовом классе, а инициализируются в наследниках (элементы данные: две стороны и угол).

4. Спроектировать иерархию классов: класс **геометрические фигуры** и его наследники: **круг, эллипс, кольцо**. Определить в базовом классе и переопределить в наследниках методы вычисления площади и периметра фигуры. Элементы-данные фигур объявляются в базовом классе, а инициализируются в наследниках (элементы данные: радиусы).

5. Спроектировать иерархию классов: класс **фигуры в пространстве** и его наследники: **шар, тор, эллипсоид**. Определить в базовом классе и переопределить в наследниках методы вычисления полной площади поверхности и объема фигуры. Элементы-данные фигур объявляются в базовом классе, а инициализируются в наследниках (элементы данные: радиусы).

6. Спроектировать иерархию классов: класс **функции** и его наследники: **квадратный корень, конечный геометрический ряд**. Определить в базовом классе и переопределить в наследниках методы вычисления значения функции $\sqrt{x}$ и $\frac{1-x^{m+1}}{1-x}, x \neq 1, m \in \mathrm{N}$ для заданного значения переменной, используя разложение в ряд Маклорена. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

7. Спроектировать иерархию классов: класс **геометрические фигуры** и его наследники: **ромб, квадрат, параллелограмм**. Определить в базовом классе и переопределить в наследниках методы вычисления площади и периметра фигуры. Элементы-данные фигур объявляются в базовом классе, а инициализируются в наследниках (элементы данные: стороны фигур).

8. Спроектировать иерархию классов: класс **фигуры в пространстве** и его наследники: **тетраэдр, куб, параллелепипед**. Определить в базовом классе и переопределить в наследниках методы вычисления полной площади поверхности и объема фигуры. Элементы-данные фигур объявляются в базовом классе, а инициализируются в наследниках (элементы данные: стороны).

9. Спроектировать иерархию классов: класс **функция от одной переменной** и его наследники: **константа, линейная зависимость, парабола**. Определить в базовом классе и переопределить в наследниках методы вычисления значения функции для заданного значения переменной. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

10.Спроектировать иерархию классов: класс **функция от одной переменной** и его наследники: **синус и его производная, косинус и его производная**. Определить в базовом классе и переопределить в наследниках методы вычисления значения функции для заданного значения переменной. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

11.Спроектировать иерархию классов: класс **функция от нескольких переменных** и его наследники: **тип 1 (**$x^2+\sqrt{3y^3}$**), тип 2 (**$3x^2\cdot\sqrt[3]{z}+e^{\sqrt{4y}}$**)**. Определить в базовом классе и переопределить в наследниках методы вычисления значения функции для заданных значений переменных. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменных).

12.Спроектировать иерархию классов: класс **геометрические фигуры** и его наследники: **ромб, прямоугольник, треугольник**. Определить в базовом классе и переопределить в наследниках методы вычисления площади и периметра фигуры. Элементы-данные фигур объявляются в базовом классе, а инициализируются в наследниках (элементы данные: стороны фигур).

13.Спроектировать иерархию классов: класс **фигуры в пространстве** и его наследники: **цилиндр, конус, эллиптический цилиндр**. Определить в базовом классе и переопределить в наследниках методы вычисления полной площади поверхности и объема фигуры. Элементы-данные фигур объявляются

в базовом классе, а инициализируются в наследниках (элементы данные: радиусы и высота).

14.Спроектировать иерархию классов: класс **функция от нескольких переменных** и его наследники: **тип 1 (**$7x^4+\sqrt{3yx}$**), тип 2 (**$\sin(x)+3z^y-yz$**).** Определить в базовом классе и переопределить в наследниках методы вычисления значения функции для заданных значений переменных. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменных).

15.Спроектировать иерархию классов: класс **функции** и его наследники: **экспонента, натуральный логарифм**. Определить в базовом классе и переопределить в наследниках методы вычисления значения функции e^x и $\ln(x)$ для заданного значения переменной, используя разложение в ряд Маклорена. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

16.Спроектировать иерархию классов: класс **функция от одной переменной** и его наследники: **арксинус и его производная, арккосинус и его производная.** Определить в базовом классе и переопределить в наследниках методы вычисления значения функции для заданного значения переменной. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

17.Спроектировать иерархию классов: класс **многогранники** и его наследники: **тетраэдр, куб, октаэдр**. Определить в базовом классе и переопределить в наследниках методы вычисления полной площади поверхности и объема фигуры. Элементы-данные фигур объявляются в базовом классе, а инициализируются в наследниках (элементы данные: длина стороны).

18.Спроектировать иерархию классов: класс **геометрические фигуры** и его наследники: **ромб, квадрат, трапеция.** Определить в базовом классе и переопределить в наследниках методы вычисления площади и периметра

фигуры. Элементы-данные фигур объявляются в базовом классе, а инициализируются в наследниках (элементы данные: стороны фигур).

19.Спроектировать иерархию классов: класс **фигуры в пространстве** и его наследники: **цилиндр, конус, усеченный конус**. Определить в базовом классе и переопределить в наследниках методы вычисления полной площади поверхности и объема фигуры. Элементы-данные фигур объявляются в базовом классе, а инициализируются в наследниках (элементы данные: радиусы и высота).

20.Спроектировать иерархию классов: класс **функция от одной переменной** и его наследники: **экспоненциальная и ее производная, гиперболический синус и его производная, гиперболический косинус и его производная**. Определить в базовом классе и переопределить в наследниках методы вычисления значения функции для заданного значения переменной. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

21.Спроектировать иерархию классов: класс **многогранники** и его наследники: **тетраэдр, додекаэдр, икосаэдр.** Определить в базовом классе и переопределить в наследниках методы вычисления полной площади поверхности и объема фигуры. Элементы-данные фигур объявляются в базовом классе, а инициализируются в наследниках (элементы данные: длина стороны).

22.Спроектировать иерархию классов: класс **функции** и его наследники: **синус, косинус**. Определить в базовом классе и переопределить в наследниках методы вычисления значения функции $\sin(x)$ и $\cos(x)$ для заданного значения переменной, используя разложение в ряд Маклорена. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

23.Спроектировать иерархию классов: класс **фигуры в пространстве** и его наследники: **пирамида, призма, усеченная пирамида**. Определить в

базовом классе и переопределить в наследниках методы вычисления полной площади поверхности и объема фигуры. Элементы-данные фигур объявляются в базовом классе, а инициализируются в наследниках (элементы данные: стороны и высота).

24.Спроектировать иерархию классов: класс **функция от одной переменной** и его наследники: **синус и его производная, натуральный логарифм и его производная.** Определить в базовом классе и переопределить в наследниках методы вычисления значения функции для заданного значения переменной. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

25.Спроектировать иерархию классов: класс **функции** и его наследники: **тангенс, арктангенс**. Определить в базовом классе и переопределить в наследниках методы вычисления значения функции $tg(x)$ и $arctg(x)$ для заданного значения переменной, используя разложение в ряд Маклорена. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

26.Спроектировать иерархию классов: класс **функции** и его наследники: **арксинус, арккосинус**. Определить в базовом классе и переопределить в наследниках методы вычисления значения функции $\arcsin(x)$ и $\arccos(x)$ для заданного значения переменной, используя разложение в ряд Маклорена. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

27.Спроектировать иерархию классов: класс **функции** и его наследники: **синус гиперболический, косинус гиперболический.** Определить в базовом классе и переопределить в наследниках методы вычисления значения функции $sh(x)$ и $ch(x)$ для заданного значения переменной, используя разложение в ряд Маклорена. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

28.Спроектировать иерархию классов: класс **функции** и его наследники: **тангенс гиперболический, арктангенс гиперболический**. Определить в базовом классе и переопределить в наследниках методы вычисления значения функции $th(x)$ и $arcth(x)$ для заданного значения переменной, используя разложение в ряд Маклорена. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

29.Спроектировать иерархию классов: класс **функция от одной переменной** и его наследники: **синус, косинус, тангенс, котангенс.** Определить в базовом классе и переопределить в наследниках методы вычисления значения функции для заданного значения переменной. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменной).

30.Спроектировать иерархию классов: класс **функция от нескольких переменных** и его наследники: **тип 1 (**$2z^x + \sqrt{3y^3}$**), тип 2 (**$3x^2 \cdot \sqrt[3]{xy} + 3z^4 - \sqrt{2u}$**).** Определить в базовом классе и переопределить в наследниках методы вычисления значения функции для заданных значений переменных. Элементы-данные объявляются в базовом классе, а инициализируются в наследниках (элементы данные: значение переменных).

Уровень Б (+1 балл)

Выполнить задание уровня А, с условием, работы с наследниками через объект базового абстрактного класса. Определяемые в базовых классах методы должны быть чисто виртуальными.

4 ТРЕБОВАНИЯ К ОТЧЕТУ

Отчет подается после полной сдачи и защиты лабораторной работы в электронном виде (документ Word).

Отчет должен быть оформлен согласно ДСТУ 3008-95.

В отчет должен содержать следующие пункты:

- титульный лист;
- содержание;
- цель работы;
- постановка задачи;
- аналитические выкладки;
- пошаговый алгоритм решения задачи;
- UML-диаграмму классов;
- исходный код программы с комментариями;
- примеры работы программы;
- выводы, с обоснованием результата.

Отчет оценивается максимально в 0,5 балла.

5 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое полиморфизм?
2. Что такое раннее и позднее связывание?
3. В чем недостатки раннего и позднего связывания?
4. Что такое виртуальный метод (функция)?
5. В чём разница между виртуальным и чисто виртуальным методом (функцией)?
6. Что такое абстрактный класс?
7. Что такое интерфейс?
8. Как в языке C++ реализуются интерфейсы?
9. Можно ли создать объект абстрактного класса?
10. Можно ли вызвать метод абстрактного родительского класса через операцию расширения видимости?