МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт комплексной безопасности и специального приборостроения
Кафедра КБ-2 «Прикладные информационные технологии»

А.А. МЕРСОВ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКОЙ РАБОТЫ № 7

Изучение механизма простого и множественного наследования

по дисциплине: **«Языки программирования»**
(наименование дисциплины)

Москва – 2021

УДК
ББК

Печатается по решению редакционно-издательского совета «МИРЭА – Российский технологический университет»

Мерсов А.А.

Методические указания по выполнению практической работы № 7 по языкам программирования / А.А. Мерсов– М.: МИРЭА – Российский технологический университет, 2021.

Методические указания предназначены для выполнения практической работы по дисциплине «Языки программирования» и содержит перечень вариантов практической работы, а также краткое изложение теоретического материала в форме пояснений к заданию на работу. Для студентов, обучающихся по направлениям 10.03.01, 10.05.05, 10.05.03, 10.05.04.

Материалы рассмотрены на заседании учебно-методической комиссии
КБ-2 Протокол №1 от «28» августа 2021 г.
и одобрены на заседании кафедры КБ-2.

зав. кафедрой КБ-2

к.т.н. _____ / О.В.Трубиенко /

УДК ББК

Содержание

Общие указания к выполнению практической работы

Практические работы выполняются с использованием персональных компьютеров. Указания по технике безопасности совпадают с требованиями, предъявляемыми к пользователю ЭВМ. Другие опасные факторы отсутствуют.

Цель практической работы

Цель работы: ознакомление с механизмом простого и множественного наследования.

Практическая работа предполагает выполнение задания разработке и тестированию программного обеспечения.

Наследование- создание нового класса на базе уже имеющегося, или базового класса. Принцип наследования состоит в том, что элементы данных и методы базового класса автоматически становятся элементами данных нового класса.

Существуют два типа наследования:

Простое наследование- когда каждый потомок имеет только один родительский или базовый класс.

Множественное наследование– когда несколько базовых классов используются для создания нового класса - потомка, наследующих свойства всех своих родительских классов.

Формат описания класса при наследовании

Class *имя_класса_наследника*: *спецификатор_доступа имя_базового_класса*

{

собственные данные;

собственные методы;

};

Спецификатор доступа(*private* |*protected* |*public*) при объявлении производного класса определяет, какие элементы базового класса наследуются производным классом.

Ограничение доступа для производных классов

Если производный класс наследует базовый с ключом:

private– все элементы унаследованные от базового класса скрываются для всех следующих потомков и рассматриваются как *private* текущего класса;

protected – разрешает последующим потомкам доступ к элементам *protected*, но закрывает для потомков *public*, превращая их в *protected.*

Public – права доступа к элементам базового класса определяются самим базовым классом, как для текущего, так и для всех других потомков;

Конструкторы и деструкторы при наследовании

При создании конструктора производного класса:

- сначала вызывается конструктор базового класса, а затем вызывается конструктор текущего класса
- если базовый класс имеет несколько конструкторов, то необходимый указывается при определении конструктора производного класса;
- если конструктор базового класса имеет параметры, то они включаются в список параметров производного класса.

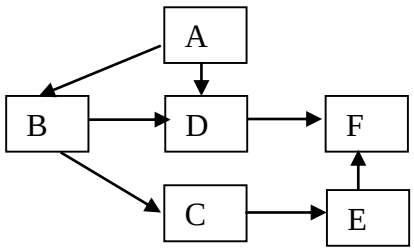
Деструкторы производного класса:

- не имеют параметров
- сначала вызывается деструктор производного класса, а затем базового
- деструкторы вызываются в порядке, обратном вызовам конструкторов.

Методический пример

Требуется построить схему классов с реализацией ее на языке программирования С++. Необходимо построить результирующий экземпляр класса и из него запустить функцию. Определить из какого класса она будет запускаться.

	A	B	C	D	E	F
A						
B	1					
C		1				
D	1	1			1	
E			1			
F				1	1	



```
class A{
public: f(){cout<<”A”;}}
class B:public virtual A{
public: f(){cout<<”B”;}}
class D:public virtual A,public virtual B{
public: f(){cout<<”D”;}}
class C:public virtual B{
public: f(){cout<<”C”;}}
class E:public C{
public: f(){cout<<”E”;}}
class F:public D,public E{};
void main(){
F object;
object.f();
}
```

В соответствии с правилом доминирования и принципами наследования и переопределения функций будет запускаться функция из класса E

Варианты задания

Схемы наследования:

1. если в вертикальном столбце нет ни одной 1, следовательно, данный класс является выходным.
2. если в строке нет ни одной 1, следовательно, данный класс не является производным ни от кого другого.

Таблица 1

	A	B	C	D	E	F
A						
B	1					
C	1					
D		1			1	
E			1			
F				1	1	

Таблица 2

	A	B	C	D	E	F
A						
B	1					
C	1					
D	1	1			1	
E			1	1		
F				1	1	

Таблица 3

	A	B	C	D	E	F
A						
B	1					
C		1				
D		1	1		1	
E				1		
F				1	1	

Таблица 4

	A	B	C	D	E	F
A						
B	1		1			
C	1					
D		1			1	
E			1	1		
F				1	1	

Таблица 5

	A	B	C	D	E	F
A						
B	1					
C	1					
D		1	1		1	
E			1	1		
F		1		1	1	

Таблица 6

	A	B	C	D	E	F
A						
B	1					
C	1					
D		1	1		1	
E	1		1			
F				1	1	

Таблица 7

	A	B	C	D	E	F
A						
B	1					
C	1					
D			1		1	
E			1			
F	1		1	1	1	

Таблица 8

	A	B	C	D	E	F
A						
B	1					
C	1					
D		1			1	
E	1	1	1			
F				1	1	

Таблица 9

	A	B	C	D	E	F
A						
B	1					
C		1				
D			1		1	
E	1	1	1			
F			1	1		

Таблица 0

	A	B	C	D	E	F
A						
B	1					
C	1	1				
D	1	1			1	
E		1	1			
F			1	1	1	