

Документ подписан простой электронной подписью

Информация о владельце:

ФИО: Петровская Анна Викторовна

Должность: Директор

Дата подписания: 29.08.2025 14:38:26

Уникальный программный ключ:

798bda6555fbdebe827768f6f1710bd17a9070c31fd1b6a6ac5a1f10c8c5199

Приложение 6

к основной профессиональной образовательной программе
по направлению подготовки 09.03.03 Прикладная информатика
направленность (профиль) программы
Прикладная информатика в экономике

Министерство науки и высшего образования Российской Федерации
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Российский экономический университет имени Г.В. Плеханова»
Краснодарский филиал РЭУ им. Г. В. Плеханова

Факультет экономики, менеджмента и торговли

Кафедра экономики и цифровых технологий

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

по дисциплине Основы объектно-ориентированного
программирования

Направление подготовки 09.03.03 Прикладная информатика

Направленность (профиль) программы Прикладная информатика в экономике

Уровень высшего образования Бакалавриат

Год начала подготовки 2024

Краснодар – 2024 г.

Составитель:

к.к., доцент кафедры экономики и цифровых технологий Л.В. Кухаренко

Оценочные материалы одобрены на заседании кафедры экономики и цифровых технологий Краснодарского филиала РЭУ им. Г.В. Плеханова протокол № 9 от 14 марта 2024 г.

Оценочные материалы составлены на основе рабочей программы по дисциплине «Основы объектно-ориентированного программирования», утвержденной на заседании базовой кафедры цифровой экономики института развития информационного общества федерального государственного бюджетного образовательного учреждения высшего образования «Российский экономический университета имени Г.В. Плеханова» протокол № 11 от 17 мая 2021 г., разработанной авторами:

Иванов Е.А., ст. преподаватель, базовой кафедры Прикладной информатики и информационной безопасности

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

по дисциплине **Основы объектно-ориентированного программирования**

ПЕРЕЧЕНЬ КОМПЕТЕНЦИЙ С УКАЗАНИЕМ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ И ЭТАПОВ ИХ ФОРМИРОВАНИЯ ПО ДИСЦИПЛИНЕ

Формируемые компетенции (код и наименование компетенции)	Индикаторы достижения компетенций (код и наименование индикатора)	Результаты обучения (знания, умения)	Наименование контролируемых разделов и тем
ОПК-7. Способен разрабатывать алгоритмы и программы, пригодные для практического применения	ОПК-7.1. Разрабатывает и реализует эффективные алгоритмы и программы с использованием современных технологий и языков программирования	ОПК-7.1. 3-2. Знает основные языки программирования и современные программные среды разработки информационных систем и технологий ОПК-7.1. 3-3. Знает современные методы разработки и реализации алгоритмов решения прикладных задач с использованием различных языков программирования ОПК-7.1. 3-4. Знает технологию разработки и отладки программ ОПК-7.1. 3-5. Знает синтаксис и семантику языков программирования, виды вычислительных процессов, типы данных	Тема 1. Класс как абстрактный тип. Конструкторы и деструкторы Тема 2. Члены класса, статический полиморфизм Тема 3. Отношения между классами Тема 4. Динамический полиморфизм, механизм виртуальных функций. Средства обработки ошибок и исключений
		ОПК-7.1. У-1. Умеет выбирать и применять современные языки программирования для реализации алгоритмов и написания прикладных программ в зависимости от типа прикладной задачи ОПК-7.1. У-2. Умеет программировать приложения и создавать программные прототипы решения прикладных задач ОПК-7.1. У-3. Умеет разрабатывать программный код в современной среде программирования, анализировать и адаптировать под нужды заказчика	Тема 5. Множественное наследование, интерфейсы, динамическая информация о типе, параметрический полиморфизм Тема 6. Библиотека универсальных шаблонов

МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ, ХАРАКТЕРИЗУЮЩИЕ ЭТАПЫ ФОРМИРОВАНИЯ КОМПЕТЕНЦИЙ

Перечень учебных заданий на аудиторных занятиях

Вопросы для проведения опроса во время аудиторных занятий

Тема 1. Класс как абстрактный тип. Конструкторы и деструкторы

- Является ли язык Си++ чисто объектно-ориентированным языком программирования?
- Перечислите основные механизмы объектно-ориентированного подхода.
- Для чего нужны пространства имен?
- Какие виды доступа к членам классов возможны в Си++?
- В чем состоит отличие использования структуры вместо класса?
- Чем являются конструктор и деструктор по отношению к классу?
- Какие виды конструкторов возможны в Си++?
- Для чего применяется деструктор?
- Возможно ли использование конструктора с параметрами в качестве конструктора по умолчанию?
- Для чего используется спецификатор explicit?

Тема 2. Члены класса, статический полиморфизм

- Каким образом инициализируются статические члены класса?
- Могут ли статические члены класса быть inline-функциями?
- В чем состоит основная особенность дружественных функций?
- Для чего используется перегрузка операций?
- Для каких целей осуществляется перегрузка функций и методов?

Тема 4. Динамический полиморфизм, механизм виртуальных функций. Средства обработки ошибок и исключений

- Что происходит при вызове виртуальной функции через полиморфный объект?
- В каких случаях деструктор следует объявлять виртуальным?
- Что такое таблица виртуальных методов?
- Как объявляется чистая виртуальная функция?
- Что такое абстрактный класс?

Тема 5. Множественное наследование, интерфейсы, динамическая информация о типе, параметрический полиморфизм

- Почему при множественном наследовании может возникать проблема неоднозначности?
- Какую проблему решает использование виртуального базового класса?
- Каким образом интерфейс может решить проблему неоднозначности?
- Что включает в себя механизм безопасного преобразования объектов?
- Доступен ли механизм RTTI по умолчанию?

Вопросы для проведения групповой дискуссии во время аудиторных занятий

Тема 3. Отношения между классами

- Доступность членов базового класса и модификаторы доступа при наследовании.
- Случаи, в которых может потребоваться наследование.
- Какие компоненты базового класса присутствуют в производном классе?
- Какие методы базового класса не наследуются?
- Каким образом можно использовать конструктор базового класса?
- Что происходит при наследовании, если в базовом классе есть метод такой же как и в производном?
- Какое приведение типа является безопасным?

Тема 6. Библиотека универсальных шаблонов

- Чем STL лучше или хуже других библиотек?
- Есть ли смысл писать свой класс для работы с деревьями, если он уже есть в STL?
- Какие основные компоненты входят в состав библиотеки STL?
- Перечислите основные виды контейнеров.
- Какие основные операции можно реализовать с помощью итераторов?
- Почему алгоритмы не включены непосредственно в контейнеры?
- В чем состоят достоинства и недостатки STL подхода?

Задания для текущего контроля

Комплект тестов/тестовых заданий

Индикаторы достижения: ОПК-7.1

Тестовые задания закрытого типа

1. Тестовое задание закрытого типа с выбором одного правильного ответа

1.1. Что называется конструктором?

- а) метод, имя которого совпадает с именем класса и который вызывается автоматически при создании объекта класса
- б) метод, имя которого совпадает с именем класса и который вызывается автоматически при объявлении класса (до создания объекта класса)
- в) метод, имя которого необязательно совпадает с именем класса и который вызывается при создании объекта класса
- г) метод, имя которого совпадает с именем класса и который необходимо явно вызывать из головной программы при объявлении объекта класса

1.2. Объект - это

- а) переменная, содержащая указатель на класс
- б) экземпляр класса
- в) класс, который содержит в себе данные и методы их обработки

1.3. Выберите правильное объявление производного класса

- а) `class MoreDetails::Details;`
- б) `class MoreDetails: public class Details;`
- в) `class MoreDetails: public Details;`
- г) `class MoreDetails: class(Details)`

1.4. Возможность и способ обращения производного класса к элементам базового определяется

- а) ключами доступа: private, public, protected в теле производного класса
- б) только ключом доступа protected в заголовке объявления производного класса
- в) ключами доступа: private, public, protected в заголовке объявления производного класса
- г) ключами доступа: private, public, protected в теле базового класса

1.5. Дружественная функция - это

- а) функция другого класса, среди аргументов которой есть элементы данного класса
- б) функция, объявленная в классе с атрибутом friend, но не являющаяся членом класса;
- в) функция, являющаяся членом класса и объявленная с атрибутом friend;
- г) функция, которая в другом классе объявлена как дружественная данному

1.6. Переопределение операций имеет вид:

- а) имя_класса, ключевое слово operation, символ операции
- б) имя_класса, ключевое слово operator, символ операции, в круглых скобках могут быть указаны аргументы
- в) имя_класса, ключевое слово operator, список аргументов
- г) имя_класса, два двоеточия, ключевое слово operator, символ операции

1.7. Существует ли в C++ готовый набор шаблонов:

- а) нет
- б) да, существует специальная библиотека STL
- в) зависит от версии компилятора

1.8. Выберите правильный вариант выделения динамической памяти под переменную X типа float:

- а) float *ptr = new float; X = *ptr;
- б) float &ptr = new float; X = &ptr;
- в) float * ptr = &X; X = new float;

1.9. Дано определение класса

```
class monstr {  
    int health, armo;  
    monstr(int he, int arm);  
public:  
    monstr(int he=50, int arm=10);  
    intcolor;  
}
```

Укажите свойства и методы, доступные внешним функциям

- а) health, armo
monstr(int he, int arm);
monstr(int he=50, int arm=10);
- б) intcolor;
monstr(int he=50, int arm=10);
- в) health, armo, color
monstr(int he=50, int arm=10);
- г) intcolor;
monstr(int he, int arm);

1.10. Какое значение возвращает деструктор?

- а) 0/1
- б) true/false
- в) никакого

2. Тестовое задание закрытого типа с выбором нескольких правильных ответов

2.1. Членами класса могут быть

- а) только переменные, объявленные как private
- б) только функции, объявленные как public
- в) переменные, объявленные как private
- г) функции, объявленные как private
- д) только переменные и функции, объявленные как private
- е) переменные и функции, объявленные как public

2.2. Отметьте правильные утверждения

- а) конструкторы класса не наследуются
- б) конструкторов класса может быть несколько, их синтаксис определяется программистом
- в) конструкторов класса может быть несколько, но их синтаксис должен подчиняться правилам перегрузки функций
- г) конструктор возвращает указатель на объект
- д) конструктор не возвращает значение

2.3. Какие ключевые слова используются для создания и обработки исключительных ситуаций?

- а) try
- б) delete
- в) catch
- г) return
- д) throw

2.4. Выберите правильные утверждения

- а) деструктор - это метод класса, применяемый для удаления объекта
- б) деструктор - это метод класса, применяемый для освобождения памяти, занимаемой объектом
- в) деструктор - это отдельная функция головной программы, применяемая для освобождения памяти, занимаемой объектом
- г) деструктор не наследуется
- д) деструктор наследуется, но должен быть перегружен

2.5. Выберите правильные утверждения:

- а) если элементы класса объявлены как private, то они доступны только наследникам класса, но не внешним функциям
- б) если элементы класса объявлены как private, то они недоступны ни наследникам класса, ни внешним функциям
- в) если элементы объявлены как public, то они доступны наследникам класса, но не внешним функциям
- г) если элементы объявлены как public, то они доступны и наследникам класса, и внешним функциям

2.6. Выберите правильные соответствия между спецификатором базового класса, ключом доступа в объявлении производного класса и правами доступа производного класса к элементам базового

- а) ключ доступа - public; в базовом классе: private; права доступа в производном классе - protected

- б) ключ доступа - любой; в базовом классе: private; права доступа в производном классе - нет прав
- в) ключ доступа - protected или public ; в базовом классе: protected; права доступа в производном классе - protected
- г) ключ доступа - private; в базовом классе: public; права доступа в производном классе - public
- д) ключ доступа – любой; в базовом классе: public; права доступа в производном классе – такие же, как ключ доступа

2.7. Выберите правильные утверждения:

- а) одна функция может быть дружественной нескольким классам
- б) дружественная функция не может быть обычной функцией, а только методом другого класса
- в) дружественная функция объявляется внутри класса, к элементам которого ей нужен доступ
- г) дружественная функция не может быть методом другого класса

2.8. Выберите правильные утверждения:

- а) по умолчанию члены класса имеют атрибут private
- б) по умолчанию члены класса имеют атрибут public;
- в) члены класса имеют доступ только к элементам public;
- г) элементы класса с атрибутом private доступны только членам класса

2.9. Полиморфизм реализован через механизмы:

- а) перегрузки функций,
- б) наследования методов, шаблонов;
- в) наследования методов, виртуальных функций, шаблонов
- г) перезагрузки виртуальных функций, шаблонов
- д) - перегрузки функций, наследования, виртуальных функций.

2.10. Какие методы присутствуют в любом классе?

- а) ввода
- б) вывода
- в) никакие методы не являются обязательными
- г) конструктор
- д) деструктор

Тестовые задания открытого типа

3. Тестовые задания открытого типа, т.е. с указанием ответов

3.1. Свойство языка программирования, позволяющее объединить и защитить данные и код в объекте и скрыть реализацию объекта от пользователя это(Ответ записать словом в именительном падеже)

3.2. Возможность объектов с одинаковой спецификацией иметь различную реализацию это

3.3.... - это тип данных, определяемый пользователем и сочетающий в себе данные и функции их обработки

3.4.... – это метод, который освобождает память, занимаемую объектом

3.5.... - это механизм, посредством которого производный класс получает элементы родительского и может дополнять либо изменять их свойства и методы

3.6.... функции – это определение функции, в которой типу обрабатываемых данных присвое-

но условное обозначение

3.7. Функции базового класса, которые могут быть переопределены в производном классе это функции

3.8. В какой строке в кодепредставленном на картинке объявляется экземпляр класса MyClass?

```
1 #include "stdafx.h"
2 #include <iostream>
3 using namespace std;
4
5 class MyClass
6 {
7 public:
8     void show()
9     {
10         cout << "Тест на знание классов в C++";
11     }
12 };
13
14 int main(int argc, char* argv[])
15 {
16     MyClass mc;
17     mc.show();
18     system("pause");
19     return 0;
20 }
```

3.9. На изображении представлен отрывок кода, в котором объявляется класс policeman. Укажите имя класса-наследника

```
#include "worker.h"
#include <string>
class policeman : public worker {
public:
    policeman(
        string name,
        string second_name,
        int hours
    ) : human(
        name,
        second_name
    )
};
```

3.10. Программист написал простую программу, приведенную на изображении, но забыл дописать строчку, которая выводит сумму на экран. Что нужно написать в 25 строке?

```

1  #include <conio.h>
2  #include <iostream.h>
3  class summa
4  {
5      int x,y;
6      public:
7          void get_xy()
8          {
9              cout << "Введите X: ";
10             cin >> x;
11             cout << "Введите Y: ";
12             cin >> y;
13         }
14
15         int sum_xy()
16         {
17             return x+y;
18         }
19     }
20
21     int main()
22     {
23         summa s1;
24         s1.get_xy();
25         // ВЫВОД СУММЫ
26     }

```

3.11. Данные, характеризующие состояние объекта называются - ... объекта

3.12. Изменение состояния объекта в ответ на какое-либо действие это ...

3.13. Действие, которое может выполнить объект это ...

3.14. Характеристику объекта описывает ...

3.15. Какой будет результат выполнения следующего кода?

```

class A {public:
    int y;
    intinc(int x) { return ++y; };
    intinc(short x) { return x + y; };;
Aobj;
int y = 5;
obj.y = 6;
cout<<obj.inc(y);

```

3.16. С каким модификатором доступа должны быть описаны члены класса, чтобы к ним имели доступ только объекты самого класса?

3.17. С каким модификатором доступа должны быть описаны члены класса, чтобы к ним

имели доступ только объекты самого класса и объекты класса потомка?

3.18. С каким модификатором доступа должны быть описаны члены класса, чтобы к ним имели доступ любые объекты?

3.19. Что будет напечатано в результате работы программы:

```
class A { public: A()
{ cout<< "A";
};
virtual A* new_A()
{ return new A();
}
};
class B : public A { public: B()
{cout<< "B" ;
};
A* new_A()
{ return new B();
}
};
void fun(A* p1, A* p2) { A* p3 = p1->new_A();
A* p4 = p2->new_A();
}
int main()
{A* p1 = new A;
B* p2 = new B;
fun(p1,p2);
}
```

3.20. Шаблон A и его специализации объявлены следующим образом:

```
template <class T>
class A{ public: A()
{ printf("1");
}
};
template <>
class A<int>
{ public: A()
{ printf("2");
}
};
template <>
class A<char*>
{ public: A()
{printf("3");
}
};
```

Какой будет результат после выполнения кода

A<int>a;

A<char> a1;

A<long> a2;

3.21. Какой результат будет после выполнения программы?

```
int main()
{ try
{ try
{ try
{ throw 1;
}
catch (int) { cout<< "Exception 1"; }
}
catch (int) { cout<< "Exception 2"; }
}
catch (int){ cout<< "Exception 3"; }
return 0;
}
```

3.22. Что будет выведено на экран в результате выполнения программы?

```
template <class T>
class A
{ public : T x1;
        T x2;
        T func(T x3)
        { x3 += x1 + x2;
          return x3;
        }
};
int main()
{
  A <int> A1;
  A1.x1 = 3;
  A1.x2 = 10;
  cout<< A1.func(1); ....
}
```

3.23. Какой результат будет у следующего выражения?

```
int main()
{ try
{ try
{ try
{ throw 1;
}
catch (float) { cout << "Exception 1"; }
}
catch (int) { cout << "Exception 2"; }
}
catch (int){ cout << "Exception 3"; }
return 0;
}
```

3.24. Дано описание класса и переменной A

```
class A { public: int a; };
```

```
A obj;
```

Как правильно обратиться к переменной a?

3.25. Какое ключевое слово необходимо поставить на место «???» в приведенном примере так, чтобы класс C не имел доступа к неоткрытым (non-public) членам класса A?

```
class A {};  
class B :  
??? A {};  
class C :  
public B {};
```

3.26. Что выведется на экран в результате работы следующего кода?

```
class Base  
{ public: Base(int _x = 10){x = _x;}  
void f()  
{cout<<x*2<<endl;}  
protected: int x;  
};  
class Derived:  
public Base  
{ public: void f()  
{cout<<x/2<<endl;}  
};  
int main()  
{  
Derived d;  
Base* pB= &d;  
pB->f();  
return 0;  
}
```

3.27. Какой уровень доступа имеет переменная «x» из класса «derived» в приведенном примере?

```
class basex  
{ protected: int x;  
};  
class derived :  
private basex {};
```

3.28. Что выведется на экран в результате работы следующего кода?

```
class Base  
{ public: Base(int _x=10)  
{x=_x;}  
virtual void f()  
{cout<<x*2<<endl;}  
protected: int x;};  
class Derived: public Base  
{ public: void f()  
{cout<<x/2<<endl;}  
};  
int main()  
{  
Derived d;  
Base* pB= &d;
```

```

pB->f();
return 0;
}

```

3.29. Какое отношение между классами реализовано в данном фрагменте программы?

```

class A
{ public: A() { a = 0; };
virtual int f()
{ return a;
}
int a;
};
class B: public A
{ public: B()
{ a = 1;
};
virtual int f()
{ return a;
}
};

```

3.30. Как называется выделение существенных характеристик объекта, отличающих его от других видов объектов?

Ответы (ключи) на тестовые задания представить в форме таблицы

Номер во- проса	Варианты ответа	Номер во- проса	Варианты ответа	Номер во- проса	Варианты ответа
1.1	а	1.2	б	1.3	в
1.4	в	1.5	б	1.6	б
1.7	б	1.8	а	1.9	б
1.10	в				
2.1	В,г,е	2.2	А,в,д	2.3	А,в,д
2.4	Б,г	2.5	Б,г	2.6	Б,в,д
2.7	А,в	2.8	А,г	2.9	А,г
2.10	Г,д				
3.1	инкапсуляция	3.2	полиморфизм	3.3	класс
3.4	деструктор	3.5	наследование	3.6	шаблон
3.7	виртуальные	3.8	16	3.9	Policeman
3.10	cout<<<< s1.sum_xy();	3.11	атрибуты	3.12	событие
3.13	метод	3.14	свойство	3.15	7
3.16	private	3.17	protected	3.18	public
3.19	AABAAB	3.20	211	3.21	Exception 1
3.22	14	3.23	Exception 2	3.24	obj.a
3.25	private	3.26	20	3.27	private
3.28	5	3.29	наследование	3.30	абстракция

Комплект заданий для контрольной работы

Индикаторы достижения: ОПК-7.1

Задание 1

Разработать класс для рисования графических примитивов с названием Canvas.

В конструкторе класса должен создаваться связный список (можно использовать класс Vector из STL), в который будут добавляться объекты классов графических примитивов и их тип

1. Точка
2. Отрезок
3. Эллипс (+ свойство закрашен/не закрашен)
4. Прямоугольник (+ свойство закрашен/не закрашен)
5. Многоугольник (+ свойство закрашен/не закрашен)

Эллипс и прямоугольник задаются координатами левого верхнего и правого нижнего углов (для эллипса – это прямоугольник в который он вписан)

Координаты начинаются в левом верхнем углу.

В деструкторе список объектов должен уничтожаться.

Методы:

1. Добавление объекта в список
2. Удаление объекта из списка
3. Очистка списка
4. Вывод информации об объектах находящихся в списке
5. Метод Show для построения изображения из объектов списка (будет сделан в конце реализации класса)

Для каждого примитива должен быть разработан соответствующий класс.

Все классы (кроме класса точка) должны использовать класс точка для хранения координат точек.

Описание классов сделать в файле Canvas.h

Реализацию методов классов сделать в файле Canvas.cpp

В основном файле подключить модуль Canvas.h и создать объекты.

Задание 2

Разработать класс описания книги.

Поля:

1. Название
2. Автор(ы)
3. Год издания
4. Издательство
5. Количество страниц

На основе класса описания книги (с помощью наследования) разработать класс библиотечного каталога книг, дополнительные поля:

1. Общее количество экземпляров
2. Количество экземпляров в наличии
3. Список читателей, взявших книгу с датой выдачи

Методы:

1. Создание/удаление книги
2. Добавление/удаление книги в каталог
3. Вывод информации по книге и читателям, взявшим книгу по идентификатору

4. Поиск книги в каталоге по названию и по автору с выдачей идентификатора
5. Выдача книги читателю
6. Возврат книги
7. Вывод списка читателей, не вернувших книги в течение года

На основе класса библиотечного каталога разработать программу библиотечного каталога

Задание 3

Создать два класса: вектор и матрица. Определить конструкторы (по умолчанию, с параметрами, копирования), деструкторы.

Память под вектор и матрицу выделяется динамически.

Определить функцию умножения матрицы на вектор как дружественную.

Задание 4

Разработать иерархию классов для нахождения корней квадратного уравнения $ax^2 + bx + c = 0$
А - базовый класс для класса В, В базовый для класса С.

Компоненты классов А,В,С - коэффициенты a,b,c. Корни уравнения и методы решения находятся в классе С.

Задание 5

Задать класс Point2D с полями x,y и конструктором с параметрами

Задать класс Z с полем z и конструктором с параметром

Задать класс Point3D наследующий классы Point2D и Z, определить конструктор для задания трех координат и метод для вывода информации о координатах.

Критерии оценки (в баллах):

- 9-10 баллов выставляется обучающемуся, если выполнены 5 заданий, а также студент **знает верно и в полном объеме:** основные языки программирования и современные программные среды разработки информационных систем и технологий; современные методы разработки и реализации алгоритмов решения прикладных задач с использованием различных языков программирования; технологию разработки и отладки программ; синтаксис и семантику языков программирования, виды вычислительных процессов, типы данных.
- 6-8 баллов выставляется обучающемуся, если выполнены 3 или 4 задания, а также студент **знает с незначительными замечаниями:** основные языки программирования и современные программные среды разработки информационных систем и технологий; современные методы разработки и реализации алгоритмов решения прикладных задач с использованием различных языков программирования; технологию разработки и отладки программ; синтаксис и семантику языков программирования, виды вычислительных процессов, типы данных.
- 3-5 балла выставляется обучающемуся, если выполнены 2 или 3 задания, а также студент **знает на базовом уровне, с ошибками:** основные языки программирования и современные программные среды разработки информационных систем и технологий; современные методы разработки и реализации алгоритмов решения прикладных задач с использованием различных языков программирования; технологию разработки и отладки программ; синтаксис и семантику языков программирования, виды вычислительных процессов, типы данных.
- 0-2 балла выставляется обучающемуся, если выполнено 0 или 1 задание, а также студент **не знает на базовом уровне:** основные языки программирования и современные программные среды разработки информационных систем и технологий; современные методы разработки и реализации алгоритмов решения прикладных задач с использованием различных языков программирования; технологию разработки и отладки программ; синтаксис и семантику языков программирования, виды вычислительных процессов, типы данных.

Задания для творческого рейтинга

Темы индивидуальных проектов

Индикаторы достижения: ОПК-7.1

Задание 1.

Разработать класс "Сотрудник" с закрытыми информационными членами.

Информационные члены класса: Место работы, Должность, Стаж, Личные данные (Ф.И.О, пол, возраст), Заработная плата (накопление на счете, сотрудник её не снимает)

Действия над объектами класса: получение информации о месте работы, занимаемой должности, стаже работы, заработной плате; изменение должности; начисление заработной платы; вывод личных данных; операции сравнения объектов; присваивание.

Статический информационный член класса: Количество сотрудников.

Статический метод: Получение количества сотрудников

На основе класса создать массив из объектов (от 2 до 5), заполнить и вывести информацию о количестве объектов и о каждом в отдельности.

Задание 2.

Разработать класс для рисования графических примитивов с названием Canvas.

В конструкторе класса должен создаваться связный, в который будут добавляться объекты классов графических примитивов и их тип

1. Точка
2. Отрезок
3. Эллипс (+ свойство закрашен/не закрашен)
4. Прямоугольник (+ свойство закрашен/не закрашен)
5. Многоугольник (+ свойство закрашен/не закрашен)

Эллипс и прямоугольник задаются координатами левого верхнего и правого нижнего углов (для эллипса – это прямоугольник в который он вписан)

Координаты начинаются в левом верхнем углу.

В деструкторе список объектов должен уничтожаться.

Методы:

1. Добавление объекта в список
2. Удаление объекта из списка
3. Очистка списка
4. Вывод информации об объектах находящихся в списке
5. Метод Show для построения изображения из объектов списка (будет сделан в конце реализации класса)

Для каждого примитива должен быть разработан соответствующий класс. Все классы (кроме класса точка) должны использовать класс точка для хранения координат точек.

Описание классов сделать в файле Canvas.h

Реализацию методов классов сделать в файле Canvas.cpp

В основном файле подключить модуль Canvas.h и создать объекты.

Задание 3.

Разработать класс описания книги

Поля

1. Название
2. Автор(ы)
3. Год издания
4. Издательство

5. Количество страниц

На основе класса описания книги (с помощью наследования) разработать класс библиотечного каталога книг.

Дополнительные поля

1. Общее количество экземпляров.
2. Количество экземпляров в наличии.
3. Список читателей взявших книгу с датой выдачи.

Методы

1. Создание/удаление книги.
2. Добавление/удаление книги в каталог.
3. Вывод информации по книге и читателям, взявшим книгу по идентификатору.
4. Поиск книги в каталоге по названию и по автору с выдачей идентификатора.
5. Выдача книги читателю.
6. Возврат книги.
7. Вывод списка читателей, не вернувших книги в течение года.

Разработать программу библиотечного каталога.

Задание 4.

Разработать иерархию классов для нахождения корней квадратного уравнения $ax^2 + bx + c = 0$
А - базовый класс для класса В, В базовый для класса С.

Компоненты классов А,В,С - коэффициенты а, b, c. Корни уравнения и методы решения находятся в классе С.

Задание 5.

Разработать класс Point2D с полями x, y и конструктором с параметрами

Разработать класс Z с полем z и конструктором с параметром

Разработать класс Point3D наследующий классы Point2D и Z, определить конструктор для задания трех координат и метод для вывода информации о координатах.

Задание 6.

Реализуйте программу, использующую контейнер Map для создания словаря соответствия английских и русских слов. С помощью данного словаря попробуйте осуществить автоматический перевод английского текста на русский язык.

Критерии оценки (в баллах):

- 16-20 баллов выставляется обучающемуся, если выполнены 6 заданий, а также студент **умеет верно и в полном объеме:** выбирать и применять современные языки программирования для реализации алгоритмов и написания прикладных программ в зависимости от типа прикладной задачи; программировать приложения и создавать программные прототипы решения прикладных задач; разрабатывать программный код в современной среде программирования, анализировать и адаптировать под нужды заказчика.

- 11-15 баллов выставляется обучающемуся, если выполнены 4 или 5 заданий, а также студент **умеет с незначительными замечаниями:** выбирать и применять современные языки программирования для реализации алгоритмов и написания прикладных программ в зависимости от типа прикладной задачи; программировать приложения и создавать программные прототипы решения прикладных задач; разрабатывать программный код в современной среде программирования, анализировать и адаптировать под нужды заказчика.

- 5-10 баллов выставляется обучающемуся, если выполнены 2 или 3 задания, а также студент **умеет на базовом уровне, с ошибками:** выбирать и применять современные языки программирования для реализации алгоритмов и написания прикладных программ в зависимости от

типа прикладной задачи; программировать приложения и создавать программные прототипы решения прикладных задач; разрабатывать программный код в современной среде программирования, анализировать и адаптировать под нужды заказчика.

- 0-4 балла выставляется обучающемуся, если выполнено 0-1 задание, а также студент **не умеет на базовом уровне:** выбирать и применять современные языки программирования для реализации алгоритмов и написания прикладных программ в зависимости от типа прикладной задачи; программировать приложения и создавать программные прототипы решения прикладных задач; разрабатывать программный код в современной среде программирования, анализировать и адаптировать под нужды заказчика.

МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ, ХАРАКТЕРИЗУЮЩИЕ ЭТАПЫ ФОРМИРОВАНИЯ КОМПЕТЕНЦИЙ ВО ВРЕМЯ ПРОВЕДЕНИЯ ПРО- МЕЖУТОЧНОЙ АТТЕСТАЦИИ

Структура экзаменационного билета

<i>Наименование оценочного средства</i>	<i>Максимальное количество баллов</i>
<i>Вопрос 1</i>	<i>13</i>
<i>Вопрос 2</i>	<i>13</i>
<i>Практическое задание</i>	<i>14</i>

Задания, включаемые в экзаменационный билет

Перечень вопросов к экзамену:

1. Введение в ООП, основные понятия.
2. Абстрактные типы данных; Пространства имен, операция “::”.
3. Синтаксис описания класса. Управление доступом к членам класса.
4. Класс как область видимости; Объявление и определение методов класса.
5. Конструкторы и деструкторы; Конструктор умолчания
6. Конструктор преобразования и конструкторы с двумя и более параметрами
7. Конструктор копирования; Спецификатор explicit
8. Конструктор копирования и операция присваивания
9. Автоматическая генерация конструкторов и деструкторов
10. Список инициализации; Порядок вызова конструкторов и деструкторов.
11. Статические члены класса. Константные члены класса.
12. Модификатор const.
13. Дружественные классы.
14. Статический полиморфизм.
15. Перегрузка бинарных операций.
16. Перегрузка унарных операций.
17. Перегрузка функций.
18. Ассоциация. ER-диаграммы.
19. Одиночное наследование.
20. Правила наследования.
21. Преобразования указателей.

22. Правила видимости при наследовании.
23. Закрытое наследование.
24. Перекрытие имен.
25. Наследование и повторное использование кода.
26. Виртуальные деструкторы.
27. Реализация виртуальных функций;
28. Абстрактные классы.
29. Чистые виртуальные функции;
30. Правила выбора обработчика исключения;
31. Стандартные исключения.
32. Последовательность действий при возникновении исключительной ситуации.
33. Видимость при множественном наследовании.
34. виртуальные базовые классы;
35. Интерфейсы; RTTI;
36. Динамическое приведение типа (dynamic_cast);
37. Параметры шаблона;
38. Шаблоны функций;
39. Специализация шаблонной функции;
40. Шаблонные классы;
41. Эквивалентность типов.
42. Контейнеры;
43. Распределители памяти;
44. STL: контейнеры
45. STL: итераторы
46. STL: алгоритмы
47. Достоинства и недостатки STL-подхода.
48. Классы и структуры Си++
49. Указатель this
50. Указатель на член класса
51. Конструктор умолчания
52. Конструктор преобразования
53. Конструктор копирования
54. Список инициализации
55. Деструкторы, порядок вызова конструкторов и деструкторов
56. Статические члены класса
57. Константные члены класса
58. Дружественные классы
59. Перегрузка операций
60. Правила наследования
61. Правила видимости при наследовании
62. Перекрытие имен при наследовании
63. Виртуальные функции
64. Виртуальные деструкторы
65. Абстрактные классы, чистые виртуальные функции
66. Обработка исключений
67. Виртуальные базовые классы
68. Видимость при множественном наследовании
69. Интерфейсы
70. Динамическая информация о типе
71. Шаблоны функций
72. Шаблонные классы

Практические задания для экзамена:

1. Создать структуру, которая будет содержать данные о студентах института (фамилия, имя, группа, средний балл). С помощью функции-члена класса необходимо вывести на экран фамилию, имя студента и номер группы. Создать аналогичный класс.
2. Создать структуру, которая будет содержать данные о книге (название, издательство, автор, количество страниц).
3. С помощью функции-члена класса необходимо вывести на экран название, издательство и автора книги. Создать аналогичный класс.
4. Написать программу, вычисляющую с помощью функции члена класса средний балл студента за сессию. Количество экзаменов является величиной переменной и задается пользователем. Список студентов обрабатывается как класс.
5. Написать программу, вычисляющую с помощью функции члена класса суммарную величину продаж книжного магазина. Количество продаж является величиной переменной и задается пользователем. Список всех продаж, состоящий из товара и цены, обрабатывается как класс.
6. Создать класс, оперирующий информацией о сотруднике фирмы, и содержащий имя, фамилию, отчество, табельный номер, количество отработанных часов за месяц, почасовой тариф. Рабочее время свыше 144 часов считается сверхурочным и оплачивается в двойном размере. С помощью функции-члена класса вывести размер заработной платы каждого сотрудника фирмы за вычетом подоходного налога, который составляет 12% от суммы заработка.
7. Создать класс для хранения информации о студентах института (фамилия, имя, группа, средний балл). Необходимо вывести на экран фамилию, имя студента и средний балл студентов одной группы, отсортированных по алфавиту.
8. На склад игрушек хранятся игрушки. Игрушки имеют название, артикул, имя фирмы, цену и количество дней хранения на складе. Создать класс для хранения вышеперечисленной информации с функцией-членом, которая выводит на печать список игрушек (название игрушки, артикул, имя фирмы, цену – 20% и количество дней хранения на складе), хранящиеся на складе больше 20 дней. Список должен быть отсортирован в порядке уменьшения количества дней хранения на складе.
9. Составить описание класса прямоугольников со сторонами, параллельными осям координат. Предусмотреть возможность перемещения прямоугольников на плоскости, изменение размеров с помощью методов класса.
10. Составить описание класса треугольников в декартовой системе координат. Предусмотреть возможность вращения треугольника вокруг начала координат на заданный угол с помощью методов класса.
11. Напишите программу, которая пересчитывает центы в эквивалентную сумму, состоящую из монет различного достоинства (25, 10, 5 центов и 1 центов). Экземпляры классов необходимо создавать динамически.
12. Написать программу, которая создает и инициализирует 2 объекта разработанного класса, вычисляет и выводит значения площадей основания, боковой и полной поверхностей. Разработать класс `Parallel`, который должен содержать закрытые переменные (`private`), `a`, `b` – стороны основания прямоугольника, `H` – высота параллелепипеда. Класс должен содержать конструктор инициализирующий указанные переменные, а также методы, вычисляющие и возвращающие значения площадей основания `getSo()`, боковой `getSb()` и полной `getSp()`.
13. Написать программу, которая создает и инициализирует 2 объекта разработанного класса, вычисляет и выводит значения площадей основания, боковой и полной поверхностей. Для создания и выполнения второго объекта используйте указатель. Разработать класс `Prisma` (прямая призма, в основании – правильный шестиугольник, вписанный в окружность), который должен содержать закрытые переменные (`private`), `R` – ра-

- диус описанной окружности, H – высота призмы. Класс должен содержать конструктор инициализирующий указанные переменные, а также методы, вычисляющие и возвращающие значения площадей основания `getSo()`, боковой `getSb()` и полной `getSp()`.
14. Написать программу, которая создает и инициализирует 2 объекта класса прямоугольников со сторонами, параллельными осям координат. Предусмотреть возможность перемещения прямоугольников на плоскости, изменение размеров, расчет площади и периметра с помощью методов класса. Использование конструкторов и деструкторов обязательно. Экземпляры классов необходимо создавать динамически.
 15. Написать программу, которая создает и инициализирует 2 объекта класса треугольников в декартовой системе координат. Предусмотреть возможность вращения треугольника вокруг начала координат на заданный угол, расчет площади и периметра с помощью методов класса. Использование конструкторов и деструкторов обязательно. Экземпляры классов необходимо создавать динамически.
 16. Определить оптимальный подбор банкнот для выдачи задаваемой суммы в рублях для банкомата (купюры -1000, 5000, 10000, 20000, 50000). Использование конструкторов и деструкторов обязательно. Экземпляры классов необходимо создавать динамически.
 17. Написать программу, которая создает и инициализирует 2 объекта класса, содержащего информацию о почтовом адресе организации, включающем индекс, страну, город, улицу, дом. Предусмотреть возможность раздельного изменения составных частей адреса, создания и уничтожения объектов этого класса. Использование конструкторов и деструкторов обязательно. Экземпляры классов необходимо
 18. создавать динамически.
 19. Составить описание класса для представления комплексных чисел с возможностью задания вещественной и мнимой частей как числами типов `double`, так и целыми числами. Обеспечить выполнение операций сложения, вычитания и умножения комплексных чисел. Использование конструкторов и деструкторов обязательно.
 20. Составить описание класса для объектов-векторов, задаваемых координатами концов в трехмерном пространстве. Обеспечить операции сложения и вычитания векторов с получением нового вектора (суммы или разности), вычисления скалярного произведения двух векторов, длины вектора, косинуса угла между векторами. Использование конструкторов и деструкторов обязательно.
 21. Составить описание класса для определения одномерных массивов целых чисел (векторов). Предусмотреть возможность обращения к отдельному элементу массива с контролем выхода за пределы индексов, возможность задания произвольных границ индексов при создании объекта и выполнения операций поэлементного сложения и вычитания массивов с одинаковыми границами индексов, умножения и деления всех элементов массива на скаляр, печати (вывода на экран) элементов массива по индексам и всего массива. Использование конструкторов и деструкторов обязательно.
 22. Написать программу, которая использует перегрузку методов класса: для целых данных вычисляется куб числа, для действительных (с плавающей точкой) – синус угла.
 23. Составить описание класса для объектов-векторов, задаваемых координатами концов в трехмерном пространстве. Обеспечить операции сложения и вычитания векторов с получением нового вектора (суммы или разности), вычисления скалярного произведения двух векторов, длины вектора, косинуса угла между векторами. Операции реализовать через перегрузку операторов.
 24. Составить описание класса для определения одномерных массивов целых чисел (векторов). Предусмотреть возможность обращения к отдельному элементу массива с контролем выхода за пределы индексов, возможность задания произвольных границ индексов при создании объекта и выполнения операций поэлементного сложения и вычитания массивов с одинаковыми границами индексов, умножения и деления всех элементов массива на скаляр, печати (вывода на экран) элементов массива по индексам и

- всего массива. Использование конструкторов и деструкторов обязательно. Операции реализовать через перегрузку операторов.
25. Написать программу, выполняющую перегрузку операций для подготовки рецептов, если вес задается в килограммах, граммах, миллиграммах.
 26. Написать программу, иллюстрирующую перегрузку двух методов класса, которая позволяет вводить значение угла в десятичном формате или в формате "градусы/минуты/секунды". Один метод класса получает данные типа `double`, а другой – типа строка. Программа возвращает значения синуса, косинуса и тангенса и котангенса данных углов.
 27. Программа вычисляет и выводит на экран сумму двух углов в десятичном формате, а потом в формате "градусы/минуты/секунды".
 28. Написать свой класс, реализующий работу с комплексными числами. Операции сложение, вычитание, деление и умножение должны быть реализованы при помощи перегрузки соответствующих операторов.
 29. Написать свой класс, реализующий матричную арифметику. Операции сложения и умножения матриц должны быть реализованы через перегрузку соответствующих операторов.
 30. Написать свой класс, реализующий матричную арифметику. Операции сложения и умножения матриц, а также сложения матрицы с числом и умножения матрицы на число должны быть реализованы в виде перегруженных методов класса.
 31. Составить описание класса многочленов от одной переменной, задаваемых степенью многочлена и массивом коэффициентов. Предусмотреть методы для вычисления значения многочлена для заданного аргумента, операции сложения, вычитания и умножения многочленов с получением нового объекта-многочлена, печать (вывод на экран) описания многочлена. Операции сложения, вычитания и умножения должны быть реализованы через перегрузку соответствующих операторов.

Тематика курсовых проектов:

1. Объектно-ориентированный подход к разбору и вычислению арифметических выражений.
2. Объектно-ориентированный подход к разбору и обработке XML.
3. Задача коммивояжера. Реализация задачи с применением ООП.
4. Векторный графический редактор с применением ООП.
5. Реализация компьютерной игры, основанной на объектах.
6. Реализация генетического алгоритма на примере игры «жизнь» с применением ООП.
7. Класс для осуществления операций над матрицами.
8. Класс для работы с векторами в трехмерном пространстве.
9. Система управления местами в гостинице с применением ООП.
10. Система учета пациентов больницы с применением ООП.
11. Система управления складом аптеки с применением ООП.
12. Система учета заказов клиентов ресторана с применением ООП.
13. Построение графиков функций с применением ООП.
14. Объектно-ориентированная модель Аквариума.
15. Задача Прима-Краскала.
16. Модель солнечной системы, на основе ООП.

Курсовая работа (курсовой проект) оценивается в соответствии с Положением о курсовых работах (проектах) в федеральном государственном бюджетном образовательном учреждении высшего образования «Российский экономический университет имени Г.В. Плеханова».

Показатели и критерии оценивания планируемых результатов освоения компетенций и результатов обучения, шкала оценивания

Шкала оценивания		Формируемые компетенции	Индикатор достижения компетенции	Критерии оценивания	Уровень освоения компетенций
85 – 100 баллов	«отлично»	ОПК-7. Способен разрабатывать алгоритмы и программы, пригодные для практического применения	ОПК-7.1. Разрабатывает и реализует эффективные алгоритмы и программы с использованием современных технологий и языков программирования	Знает верно и в полном объеме: основные языки программирования и современные программные среды разработки информационных систем и технологий; современные методы разработки и реализации алгоритмов решения прикладных задач с использованием различных языков программирования; технологию разработки и отладки программ; синтаксис и семантику языков программирования, виды вычислительных процессов, типы данных. Умеет верно и в полном объеме: выбирать и применять современные языки программирования для реализации алгоритмов и написания прикладных программ в зависимости от типа прикладной задачи; программировать приложения и создавать программные прототипы решения прикладных задач; разрабатывать программный код в современной среде программирования, анализировать и адаптировать под нужды заказчика.	Продвинутый
		ОПК-7. Способен разрабатывать алгоритмы и программы, пригодные для практического применения	ОПК-7.1. Разрабатывает и реализует эффективные алгоритмы и программы с использованием современных технологий и языков программирования	Знает с незначительными замечаниями: основные языки программирования и современные программные среды разработки информационных систем и технологий; современные методы разработки и реализации алгоритмов решения прикладных задач с использованием различных языков программирования; технологию разработки и отладки программ; синтаксис и семантику языков программирования, виды вычислительных процессов, типы данных. Умеет с незначительными замечаниями: выбирать и применять современные языки программирования для реализации алгоритмов и написания прикладных программ в зависимости от типа прикладной задачи; программировать приложения и создавать программные прототипы решения прикладных задач; разрабатывать программ-	Повышенный
70 – 84 баллов	«хорошо»	ОПК-7. Способен разрабатывать алгоритмы и программы, пригодные для практического применения	ОПК-7.1. Разрабатывает и реализует эффективные алгоритмы и программы с использованием современных технологий и языков программирования	Знает с незначительными замечаниями: основные языки программирования и современные программные среды разработки информационных систем и технологий; современные методы разработки и реализации алгоритмов решения прикладных задач с использованием различных языков программирования; технологию разработки и отладки программ; синтаксис и семантику языков программирования, виды вычислительных процессов, типы данных. Умеет с незначительными замечаниями: выбирать и применять современные языки программирования для реализации алгоритмов и написания прикладных программ в зависимости от типа прикладной задачи; программировать приложения и создавать программные прототипы решения прикладных задач; разрабатывать программ-	Повышенный

				ный код в современной среде программирования, анализировать и адаптировать под нужды заказчика.	
50 – 69 баллов	«удовлетворительно»	ОПК-7. Способен разрабатывать алгоритмы и программы, пригодные для практического применения	ОПК-7.1. Разрабатывает и реализует эффективные алгоритмы и программы с использованием современных технологий и языков программирования	Знает на базовом уровне, с ошибками: основные языки программирования и современные программные среды разработки информационных систем и технологий; современные методы разработки и реализации алгоритмов решения прикладных задач с использованием различных языков программирования; технологию разработки и отладки программ; синтаксис и семантику языков программирования, виды вычислительных процессов, типы данных. Умеет на базовом уровне, с ошибками: выбирать и применять современные языки программирования для реализации алгоритмов и написания прикладных программ в зависимости от типа прикладной задачи; программировать приложения и создавать программные прототипы решения прикладных задач; разрабатывать программный код в современной среде программирования, анализировать и адаптировать под нужды заказчика.	Базовый
менее 50 баллов	«неудовлетворительно»	ОПК-7. Способен разрабатывать алгоритмы и программы, пригодные для практического применения	ОПК-7.1. Разрабатывает и реализует эффективные алгоритмы и программы с использованием современных технологий и языков программирования	Не знает на базовом уровне: основные языки программирования и современные программные среды разработки информационных систем и технологий; современные методы разработки и реализации алгоритмов решения прикладных задач с использованием различных языков программирования; технологию разработки и отладки программ; синтаксис и семантику языков программирования, виды вычислительных процессов, типы данных. Не умеет на базовом уровне: выбирать и применять современные языки программирования для реализации алгоритмов и написания прикладных программ в зависимости от типа прикладной задачи; программировать приложения и создавать программные прототипы решения прикладных задач; разрабатывать программный код в современной среде программирования, анализиро-	Компетенции не сформированы

				вать и адаптировать под нужды заказчика.	
--	--	--	--	--	--