НЕГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ «САНКТ-ПЕТЕРБУРГСКИЙ ГУМАНИТАРНЫЙ УНИВЕРСИТЕТ ПРОФСОЮЗОВ»

Кафедра Информатики и математики
(полное наименование кафедры)

УТВЕРЖДЕН
на заседании кафедры

Протокол №1 от 01.06.2020

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ПО ДИСЦИПЛИНЕ

Объектно-ориентированное программирование

(наименование дисциплины)

09.03.03 «Прикладная информатика»
(код наименования направления подготовки /специальности/)

Прикладная информатика в экономике
(направленность/профиль/)

Санкт-Петербург

1. Общие положения

Фонд оценочных средств (ФОС) по дисциплине используется в целях нормирования процедуры оценивания качества подготовки и осуществляет установление соответствия учебных достижений запланированным результатам обучения и требованиям образовательной программы дисциплины. Предметом оценивания являются знания, умения, навыки и (или) опыт деятельности, характеризующие этапы формирования компетенций у обучающихся. Процедуры оценивания применяются в процессе обучения на каждом этапе формирования компетенций посредством определения для отдельных составных частей дисциплины методов контроля – оценочных средств. Основным механизмом оценки качества подготовки и формой контроля учебной работы студентов являются текущий контроль успеваемости и промежуточная аттестация.

1.1. Цель и задачи текущего контроля студентов по дисциплине

Цель текущего контроля – систематическая проверка степени освоения программы 09.03.03 «Прикладная информатика» дисциплины «Объектно-ориентированное программирование» уровня достижения планируемых результатов обучения - знаний, умений, навыков, в ходе ее изучения при проведении занятий, предусмотренных учебным планом. Задачи текущего контроля:

1. обнаружение и устранение пробелов в освоении учебной дисциплины;
2. своевременное выполнение корректирующих действий по содержанию и организации процесса обучения;
3. определение индивидуального учебного рейтинга студентов;
4. подготовка к промежуточной аттестации.

В течение семестра при изучении дисциплины реализуется традиционная система поэтапного оценивания уровня освоения. За каждый вид учебных действий студенты получают оценку.

1.2. Цель и задачи промежуточной аттестации студентов по дисциплине.

Цель промежуточной аттестации – проверка степени усвоения студентами учебного материала, уровня достижения планируемых результатов обучения и сформированности компетенций на момент завершения изучения дисциплины. Промежуточная аттестация проходит в форме экзамена.

Задачи промежуточной аттестации:

1. определение уровня освоения учебной дисциплины;
2. определение уровня достижения планируемых результатов обучения и сформированности компетенций;
3. соотнесение планируемых результатов обучения с планируемыми результатами освоения образовательной программы в рамках изученной дисциплины.

2. Перечень компетенций с указанием этапов их формирования в процессе освоения дисциплины

Таблица 1.

№ п\п	Контролируемые темы дисциплины	Код формируемой компетенции	Код и наименование индикатора достижения	Наименование оценочного средства
1	Основные понятия дисциплины	ПК-3	ПК-3.1. Знает концепцию объектно-ориентированного программирования, методику анализа и проектирования объектно-ориентированных программ, основные понятия, синтаксис и семантику конструкций языка программирования.	Опрос, участие в коллоквиуме, кейсы
2	Введение в объектно-ориентированный анализ и проектирование	ПК-3	ПК-3.1. Знает концепцию объектно-ориентированного программирования, методику анализа и проектирования объектно-ориентированных программ, основные понятия, синтаксис и семантику конструкций языка программирования.	Опрос, участие в коллоквиуме, кейсы
3	Классы	ПК-3	ПК-3.1. Знает концепцию объектно-ориентированного программирования, методику анализа и проектирования объектно-ориентированных программ, основные понятия, синтаксис и семантику конструкций языка программирования. ПК-3.2. Умеет проектировать, программировать и отлаживать объектно-ориентированные программы ПК-3.3. Владеет навыками объектно-ориентированного программирования, отладки и тестирования программного обеспечения навыками программирования в современных средах.	Опрос, тестирование, коллоквиум, кейсовые задания
4	Шаблоны и обработка исключительных ситуаций	ПК-3	ПК-3.1. Знает концепцию объектно-ориентированного программирования, методику анализа и проектирования объектно-ориентированных программ, основные понятия, синтаксис и семантику конструкций языка про-	Опрос, тестирование, коллоквиум, практические задания

			граммирования. ПК-3.2. Умеет проектировать, программировать и отлаживать объектно-ориентированные программы ПК-3.3. Владеет навыками объектно-ориентированного программирования, отладки и тестирования программного обеспечения навыками программирования в современных средах.	
Результат достижения планируемых результатов изучения дисциплины				экзамен

3. Описание показателей и критериев оценивания компетенций

3.1. Критерии оценивания (текущий контроль)

1. Оценка **«отлично»** выставляется студенту, если студент имеет глубокие знания учебного материала по теме практического задания, в логической последовательности излагает материал; смог ответить на все уточняющие и дополнительные вопросы;
2. Оценка **«хорошо»** выставляется, если студент показал знание учебного материала, смог ответить почти полностью на все заданные дополнительные и уточняющие вопросы;
3. Оценка **«удовлетворительно»** выставляется, если студент в целом освоил материал; однако, ответил не на все уточняющие и дополнительные вопросы;
4. Оценка **«неудовлетворительно»** выставляется студенту, если он имеет существенные пробелы в знаниях основного учебного материала по теме практического задания, который полностью не раскрыл содержание вопросов, не смог ответить на уточняющие и дополнительные вопросы.

3.2. Критерии оценивания (зачет)

Знания, умения, навыки и компетенции студентов оцениваются следующими оценками: «зачтено», «не зачтено».

«Зачтено» выставляется студенту при условии, что студент твердо знает программный материал, грамотно и последовательно его излагает, увязывает с практикой, владеет необходимыми умениями и навыками в выполнении практических заданий и решении задач.

«Не зачтено» выставляется студенту при условии, что студент не знает значительной части основного программного материала, в ответе допускает существенные ошибки, неправильные формулировки, не владеет необходимыми умениями и навыками в выполнении практических заданий и решении задач.

ПРОМЕЖУТОЧНАЯ АТТЕСТАЦИЯ

Вопросы для подготовки к промежуточной аттестации по дисциплине (зачету)

1. Принципы объектного подхода, абстрагирование, ограничение доступа, модульность, иерархия, типизация параллелизм, устойчивость.
2. Инкапсуляция и спецификация правил доступности элементов класса.
3. Конструкторы и деструкторы.
4. Наследование. Иерархия классов.
5. Одиночное и множественное наследование.
6. Способы реализации множественного наследования, их достоинства и недостатки.
7. Полиморфизм.
8. Ранее и позднее связывание.
9. Объект и процесс. Инициализация и взаимодействие объектов и процессов. Сообщения. Реализация механизмов посылки сообщений.
10. Формальные модели объектов и классов: автоматная и алгебраическая модели объектов, исчисления типов.
11. Определения системы, домена, подсистемы, элемента, связи, среды.
12. Описание классов и их взаимосвязей с использованием унифицированного языка моделирования UML.
13. Моделирование статической структуры системы. Диаграммы классов. Механизм пакетов.
14. Моделирование поведения системы. Диаграммы взаимодействия. Диаграммы состояний. Диаграммы действий.
15. Моделирование реализации системы. Диаграммы компонентов. Диаграммы размещения.
16. Имена и их объявления. Инициализация значений.
17. Область и время действия имени.
18. Классы памяти.
19. Явное управление видимостью, пространства имен.
20. Определение класса. Личная и общая части определения класса.
21. Функции-элементы класса и функции-друзья.
22. Объекты класса.
23. Интерфейс и реализация контейнерных классов для моделирования структур данных.
24. Статические члены объектов класса.
25. Вложенные и локальные классы.
26. Примеры описания и использования классов.
27. Базовый и производный классы. Функции-элементы и функции-друзья.
28. Виртуальные базовые классы.
29. Особенности доступа при множественном наследовании. Правила преобразования указателей.
30. Виртуальные функции. Таблицы виртуальных функций.
31. Чистые виртуальные функции и абстрактные базовые классы.
32. Полиморфные контейнерные классы, итераторы и аппликаторы.

33. Абстрактные и конкретные контейнерные классы.
34. Виды классов: конкретный тип, абстрактный тип, узловой класс, интерфейсный класс.
35. Инициализация баз и членов.
36. Полный объект конечного производного класса.
37. Шаблоны классов и функций.
38. Наследование шаблонных классов. Правила отождествления параметров шаблона.
39. Применение шаблонных классов для создания контейнерных классов.
40. Стандартные средства контроля подтверждений.
41. Проверка предусловий и постусловий, вычисления инвариантов.
42. Контроль асинхронных событий. Сигналы.
43. Возбуждение ситуации, описание блоков с контролем и реакций на ситуации.
44. Система классов для описания исключительных ситуаций.
45. Обработка исключительных ситуаций при обработке исключительных ситуаций.
46. Инструментальные программные средства для C++.
47. Библиотеки классов для C++.

4. Типовые контрольные задания (тесты, рефераты, курсовые работы, кейсы и др.) и методические материалы, процедуры оценивания знаний, умений и навыков

Методические рекомендации по написанию контрольных работ

Важнейшей формой учебной отчетности студента является **контрольная работа**.

Выполнение контрольной работы является промежуточной формой отчетности по изучаемой дисциплине и преследует цель лишь оценить способность студента к самостоятельному поиску источников, формированию содержания и его письменного изложения по указанной проблеме. Это важная составляющая изучения дисциплины, а также эффективная форма контроля знаний. При заочном обучении она выступает как обязательная, основная форма самостоятельной работы. В курсовой работе (в соответствии с учебным планом) студент обязан самостоятельно глубоко разобраться в изучаемых проблемах, усвоить суть темы, уяснить ее содержание и только затем письменно представить свою отчетную работу.

Выполнение контрольной работы является одним из условий допуска студента к сдаче экзамена. Работа должна соответствовать установленным требованиям, то есть в ней должны быть раскрыты все проблемы, определенные темой. Для этого студент обязан самостоятельно проанализировать первоисточники и дать исчерпывающие ответы на вопросы темы. Контрольная работа — серьезное учебное задание, и чтобы написать ее как следует, необходимо использовать те первоисточники и учебные пособия, которые позволяют полнее разобраться в

проблеме. Студент должен регулярно работать в университетской и городской библиотеке, вдумчиво конспектировать лекции преподавателей.

При написании контрольной работы следует обращать особое внимание на грамотное использование терминологии. При употреблении впервые тех или иных терминов и понятий следует давать их определения либо в самом тексте, либо в сносках.

Приступая к контрольной работе, требуется сначала ознакомиться с имеющейся литературой по теме, изучить первоисточники и составить план. Здесь, в отличие от курсовой работы, план предполагает рассмотрение одной, причем довольно широкой, проблемы, и он может состоять из двух-трех вопросов. Минимальное количество первоисточников, привлекаемых для написания курсовой работы — пять наименований.

Как правило, контрольные работы по дисциплине сугубо индивидуальны, то есть их тематика персонифицирована. Однако в отдельных случаях темы контрольных работ могут быть адресованы и сразу нескольким, и группе в целом. Таким приемом преподаватель выявляет степень усвоения какой-то важной учебной проблемы и определяет необходимость проведения дополнительных занятий по какой-либо теме. В настоящее время широко используется методика компьютерного тестирования знаний студентов по дисциплинам, в результате чего появляется возможность быстро проверять знания по наиболее важными темам и объективно оценивать их. Эта форма также может выступать как вид контрольной работы.

В качестве контрольной работы широко применяется самостоятельное изучение монографического исследования по конкретной, крайне важной проблеме, требующей глубокого рассмотрения. Этот вид работы предполагает не простое знакомство с определенным монографическим исследованием, а детальное его изучение. Для этого студенту важно знать некоторые правила работы с первоисточником, которым для него будет являться монография. Следует выяснить фамилию автора, его имя и отчество, ученую степень и звание, а также что побудило его взяться за изучение данной проблемы; обратить внимание на основные вопросы монографии и их разрешение автором, уметь раскрывать их в ходе собеседования с преподавателем.

Студенту следует письменно (предельно кратко) очертить те вопросы (полностью или частично), которые поставлены автором в монографическом исследовании; при изложении их следует указывать страницы источника.

Задания для написания контрольных работ (для заочной формы обучения)

Примерная тематика контрольных работ для студентов заочной формы обучения

Работа состоит из трех частей, которые выполняются последовательно в течении семестра.

Программирование классов с использованием наследования

При выполнении первой части контрольной работы студент должен выполнить следующие этапы работы.

- Написать классы для создания графических объектов. Классы должны иметь

общий абстрактный базовый класс Shape с чистыми виртуальными функциями.
- Использовать множественное наследование. В классах должны быть предусмотрены виртуальные функции для вывода информации об объектах в поток, а Shape должен иметь дружественный перегруженный оператор <<.
- Исходный текст должен быть разделен на три файла .h, .cpp и .cpp с тестовой программой.
- По данной части контрольной работы представляется диаграмма классов графических фигур.

Последовательность выполнения задания для первой программы

1. Согласовать с руководителем вариант первой части задания контрольной работы.
2. Нарисовать диаграмму классов графических фигур.
3. Создать классы графических фигур.
4. Организовать наследование классов графических фигур.
5. Добавить поля в классы графических фигур.
6. Добавить описания методов в классы графических фигур.
7. Объявить и описать дружественные операторы ввода/вывода.
8. Реализовать тестирующую программу.
9. Рассмотреть работу программы в режиме отладки.

Варианты фигур для индивидуальных заданий по первой части контрольной работы

1. Квадрат, прямоугольник, текст, текст в прямоугольнике.
2. Окружность, эллипс, текст, текст в эллипсе.
3. Правильный треугольник, правильный пятиугольник (с поворотом), текст, текст в пятиугольнике.
4. Точка, отрезок, треугольник, текст, текст в треугольнике.
5. Прямоугольник, овал, текст, текст в овале.
6. Квадрат, квадрат с тенью, текст, текст в квадрате.
7. Правильный треугольник, правильный треугольник со сглаженными углами, текст, текст в треугольнике.
8. Прямоугольник, таблица, текст, текст в прямоугольнике.
9. Отрезок синусоиды, квадрат с волнистыми краями, текст, текст в квадрате.
10. Прямоугольник, рамка (прямоугольник с толщиной сторон), текст, текст в прямоугольнике.
11. Треугольник, трапеция, текст, текст в трапеции.
12. Прямоугольник, параллелограм, текст, текст в параллелограмме.
13. Окружность, сечение тора, текст, текст в окружности.
14. Круг, сектор, текст, текст в секторе.
15. Окружность, правильные фигуры, текст, текст в фигурах.
16. Круг, кольцо, текст, текст в кольце.
17. Эллипс, дуга эллипса, текст, текст в эллипсе.
18. Равнобедренный треугольник, равнобедренный треугольник со скошенными углами, текст, текст в треугольнике.
19. Квадрат, куб, текст, текст в квадрате.
20. Прямоугольник, параллелограмм, параллелепипед, текст, текст в параллелограмме.

Программирование контейнерных классов

При выполнении второй части контрольной работы студент должен выполнить следующие этапы работы.

- Использовать полиморфный контейнер для размещения объектов ранее созданных классов графических фигур.
- Произвести тестирование заполненного контейнера, применяя для этого наиболее характерный набор методов и итераторов.
- По данной части контрольной работы представляется диаграмма классов графических фигур, контейнера и итераторов, а также наиболее важные отношения UML.

Последовательность выполнения задания для второй программы

1. Согласовать с руководителем вариант задания второй части контрольной работы.
2. Изучить интерфейс контейнера в соответствии с индивидуальным заданием.
3. Опробовать работу контейнера и итераторов на примере стандартных типов данных.
4. Создать контейнер и необходимые итераторы.
5. Создать по два разных объекта каждого класса графических фигур.
6. Добавить указатели на созданные объекты в контейнер и с помощью итераторов вывести информацию о графических фигурах в поток.
7. Провести демонстрацию полезных функций контейнера, используя необходимые методы и итераторы.

Варианты контейнеров для индивидуальных заданий по второй части контрольной работы

1. Вектор.
2. Очередь с двумя концами.
3. Очередь с одним концом.
4. Стек.
5. Список
6. Мультимножество.
7. Множество.

Проектирование и программирование приложений

Индивидуальность задания третьей части контрольной работы полностью определяется вариантами заданий, полученных на первых двух этапах. При выполнении последней части контрольной работы студент должен выполнить следующие этапы работы.

- Создать диалоговое приложение с элементами интерфейса для тестирования полиморфного контейнера, заполненного графическими фигурами.
- Включить в программный код приложения классы графических фигур и операции по заполнению контейнера и его тестирования.
- Обеспечить визуализацию результатов тестирования операций с контейнером с помощью диалоговых окон.
- По данной части контрольной работы представляются диаграмма компонентов итогового проекта.

Последовательность выполнения задания для третьей программы

1. Согласовать с руководителем способ визуализации результатов тестирования контейнера.
2. Создать диалоговое приложение.
3. Добавить диалоговые элементы интерфейса и окна.
4. Создать контейнер и необходимые итераторы.
5. Создать по два разных объекта каждого класса графических фигур.
6. Добавить указатели на созданные объекты в контейнер.
7. Провести демонстрацию полезных функций контейнера, используя элементы интерфейса приложения.

Требования к оформлению контрольной работы подробно представлены в Положении о бюро контрольных работ, размещенном на сайте Университета в личном кабинете на странице в Системе поддержки самостоятельной работы студентов **ПОЛОЖЕНИЕ О БЮРО КОНТРОЛЬНЫХ РАБОТ _ для работ студентов заочной формы обучения.**

Тестовые материалы

ПАСПОРТ ТЕСТОВЫХ ЗАДАНИЙ

1. Общее количество тестовых заданий в базе – 128.
2. Ограничение времени выполнения теста (в минутах) – 30 минут.
3. Автоматическое перемешивание вопросов в тесте: - **да** (нет).
4. Случайный порядок ответов в тестовом задании: - **да** (нет).
5. Критерии оценки результатов тестирования:
   - Неудовлетворительно – 0 –55% правильных ответов.
   - Удовлетворительно -55 – 75% правильных ответов.
   - Хорошо – 75 -90% правильных ответов
   - Отлично – 90% и более правильных ответов

Пример тестовых заданий для текущего контроля представлен ниже:

1. Какая функция объявляется с целью получить доступ к защищенным и внутренним атрибутам другого класса?
   - ◉ дружественная
   - ○ шаблонная
   - ○ макрофункция
   - ○ статическая
   - ○ защищенная

2. Если имеется код
   class A { public: int a, b, c; };
   A *obj;
   как обратиться к переменной b?
   - ○ obj.a
   - ○ obj->a->b

- obj->b
- obj->a.b

3. Размер объекта класса в памяти определяется
   - суммой размеров методов класса и атрибутов класса
   - суммой размеров методов класса
   - суммой размеров атрибутов класса
   - не зависит от размеров атрибутов и методов класса
4. Какой из стандартных классов используется для вывода строк на терминал:
   - strstream
   - ostream
   - ofstream
   - istream
   - ifstream

5. После компиляции программы
   - ее можно выполнять многократно без перекомпиляции
   - перед каждым последующим запуском ее нужно перекомпилировать
   - ее можно выполнять только с одним набором исходных данных

6. Процесс компиляции программы
   - переводит исходный текст в исполняемый файл
   - проверяет программу на наличие ошибок
   - приводит программы к единообразному внешнему виду
   - для языка Си++ необязателен

7. В программе на языке Си++ обязательно имеется функция
   - head
   - start
   - prime
   - main
   - finish

8. Какой правильный заголовок шаблона
   - template <class t1, class t2>
   - template <class t1,t2>
   - template <class t, class t>
   - template <class t,t>

9. Можно ли использовать класс-шаблон в качестве базового класса?
   - (•) да
   - ( ) нет

10. Цель введения . . . функций – автоматизация создания функций, которые могут обрабатывать разнотипные данные.
   - (•) шаблонов
   - ( ) прототипов
   - ( ) подписей (сигнатур)
   - ( ) аргументов

11. Если заданы классы
```
class A {... };
class B : public A { ... };
class C : public A { ... };
```
то что будет выведено при выполнении оператора
```
throw (A);
```
а обработка исключительной ситуации записана
```
catch (B& b) { cout << 1; }
catch (C& c) { cout << 2; }
catch (A& a) { cout << 3; }
catch (...) { cout << 4; }
```
   - (•) 1
   - ( ) 2
   - ( ) 3
   - ( ) 4
   - ( ) 3 4
   - ( ) 2 3 4

12. Что может быть аргументом оператора **throw**?
   - [x] целое число
   - [x] объект класса Exception
   - [x] строка
   - [x] ноль
   - [ ] условный оператор
   - [ ] вызов деструктора объекта
   - [ ] вызов оператора return

13. Какой термин не имеет отношения к объектно-ориентированному программированию?
   - ( ) инкапсуляция
   - (•) индексация

- наследование
- полиморфизм

14. Каким может быть аргумент деструктора?
    - адрес объекта
    - указатель this
    - аргумента не может быть
    - уничтожаемый объект

15. Отметьте все правильные варианты продолжения предложения: виртуальный деструктор
    - [x] может использоваться с абстрактными классами
    - [x] не нужен, если класс не наследуется
    - [ ] должен быть определен как чисто виртуальный в абстрактном классе

16. Что выведет следующая программа ?

```
#include <iostream.h>
int main() {
  int i ;
  for(i = 0; i < 9; i++)
  cout << i+1;
  return 1;
}
```

    - цифры от 0 до 8
    - цифры от 1 до 9
    - программа не будет построена из-за ошибок

17. Абстрактный класс – это класс, в котором
    - есть хотя бы один виртуальный метод
    - есть виртуальный конструктор
    - есть виртуальный деструктор
    - есть чисто виртуальный метод

18. Если не объявить тип возврата функции, то какой тип будет принят по умолчанию?
    - int;
    - void;
    - float;
    - это вызовет ошибку при компиляции

19. Какими по умолчанию объявляются элементы структуры?
    - ○ private
    - ◉ public
    - ○ protected
    - ○ по умолчанию не объявляются

20. Какой правильный вызов функции базового класса из объекта производного класса, если в производном классе эта функция была замещена?
    - ○ FunctionName();
    - ◉ Base::FunctionName();
    - ○ Base.FunctionName();
    - ○ такую функцию вызывать нельзя.

21. Методы класса определяют
    - ☑ какие операции можно выполнять с объектами данного класса
    - ☐ какие значения может принимать переменная данного класса
    - ☑ каким образом можно создавать объекты данного класса

22. Что является коллекцией переменных, объединенных с набором связанных функций
    - ◉ класс
    - ○ массив
    - ○ модуль
    - ○ пространство имен

23. Какие основные области применения языка Си++?
    - ○ системное программирование
    - ○ прикладное программирование
    - ◉ системное и прикладное программирование

24. Программа на языке Си++ начинает выполняться с:
    - ○ первой функции в программе
    - ◉ функции main
    - ○ той функции, которая указана как стартовая при компиляции программы

25. Какой правильный вариант создания класса из шаблона?
    - ○ template<char> Array;
    - ○ template Array<char>;
    - ○ Array template<char>;

- Array<char> A

26. Какие требования предъявляются к классу исключительных ситуаций?
    - [ ] он должен быть наследован от специального класса exception
    - [ ] он не может использовать множественное наследование
    - [ ] он должен содержать атрибуты только встроенных типов
    - [x] он может быть произвольным классом

27. Переопределение операции сложения приведет к…(отметьте все правильные варианты)
    - [ ] ее вызову при выполнении операции ++ с объектом класса
    - [x] ее вызову при выполнении операции сложения с объектом класса
    - [ ] преобразованию целых чисел к объекту данного класса при выполнении сложения
    - [x] возможному преобразованию объектов других классов к данному при выполнении операции сложения

28. Если определена операция вычитания для двух объектов класса **A** и операция преобразования к **int**, что будет вызвано при
    A a;
    int x;
    int y = a – x;
    - ( ) преобразование к целому
    - (•) операция вычитания, а затем преобразование к целому
    - ( ) только операция вычитания
    - ( ) произойдет ошибка

29. Какой тип будет у результата выражения?
    - ( ) такой же, как тип операндов выражения
    - (•) зависит от типа операндов выражения и выполняемой операции
    - ( ) такой, как у переменной, которой присваивается значение выражения

30. Механизм позднего связывания применяется для реализации
    - (•) полиморфизма
    - ( ) инкапсуляции
    - ( ) наследования

31. Какой класс может использоваться в качестве типа атрибута класса?
    - ( ) базовый класс данного класса
    - ( ) производный от данного класса
    - ( ) пользовательский класс

- произвольный класс

32.В результате компоновки (редактирования связей) получается:
- отредактированный исходный файл
- список имеющихся в исходном файле ошибок
- объектный файл
- промежуточное представление программы
- исполняемый файл

33.Какова последовательность создания исполняемого файла:
1) Создать файл с исходным текстом программы, который будет иметь расширение **.cpp**.
2) Скомпоновать объектный файл с необходимыми библиотеками.
3) Скомпилировать исходный код.
- 1-3-2
- 2-3-1
- 1-2-3
- 3-2-1
- 2-1-3

34.Если записано
```
class A { public: void f() { cout << 1; } };
class B : public A { public: void f() { cout << 2; } };
```
то что будет напечатано ?
```
B b; b.f();
```
- 2
- 2 1
- 1 2
- 1
- ошибка

35.Какая из записей является правильной записью абстрактного класса?
- abstract class A { virtual f() = 0; };
- class A { virtual f() = 0; };
- class A { virtual f(); };

36.Какой класс объявляется таким образом, чтобы все его функции - члены были дружественными по отношению к другому классу?
- дружественный
- базовый
- абстрактный

- виртуальный
- глобальный