Министерство образования и науки Российской Федерации
федеральное государственное бюджетное образовательное учреждение высшего образования
«Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ»

Институт **Компьютерных технологий и защиты информации**

Кафедра **Компьютерных систем**

УТВЕРЖДАЮ

Ответственный за ОП

Верш И.С.Вершинин
«31» 08 2017 г.
Регистрационный № 4010-17/Б-014

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

для проведения промежуточной аттестации по дисциплине
«Конструирование программного обеспечения»
(наименование дисциплины, практики)

Индекс по учебному плану: **Б1.В.ДВ.02.01**

Направление подготовки: **09.03.01 «Информатика и вычислительная техника»**

Квалификация: **бакалавр**

Профили подготовки: **Программное обеспечение средств вычислительной техники и автоматизированных систем**

Виды профессиональной деятельности: **научно-исследовательская; проектно-конструкторская**

Заведующий кафедрой ПМИ С.С. Зайдуллин

Разработчик профессор кафедры ПМИ, д.т.н. Л.Ю.Емалетдинова
Ст. преподаватель кафедры ПМИ, к.т.н. Н.М. Вдовичев

Казань 2017 г.

Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине (модулю)

Конструирование программного обеспечения

(наименование дисциплины)

Содержание фонда оценочных средств (ФОС) соответствует требованиям федерального государственного стандарта высшего образования (ФГОС ВО) по специальности 09.03.01 «Информатика и вычислительная техника», учебному плану специальности 09.03.01 «Информатика и вычислительная техника».

Разработанные ФОС обладают необходимой полнотой и являются актуальными для оценки компетенций, осваиваемых обучающимися при изучении дисциплины «Конструирование программного обеспечения». Разработанные ФОС полностью соответствуют задачам будущей профессиональной деятельности обучающихся, установленных ФГОС ВО по специальности 09.03.01 «Информатика и вычислительная техника». В составе ФОС присутствуют оценочные средства в виде тестовых заданий и контрольных вопросов различного уровня сложности, которые позволяют провести оценку порогового, продвинутого и превосходного уровней освоения компетенций по дисциплине.

ФОС обладают необходимой степенью приближенности к задачам будущей профессиональной деятельности обучающихся, связанной со способностью разрабатывать модели компонентов информационных систем, включая модели баз данных и модели интерфейсов "человек - электронно-вычислительная машина" (ПК-1);

Существенные недостатки отсутствуют.

Заключение. Учебно-методическая комиссия делает вывод о том, что представленные материалы соответствуют требованиям ФГОС ВО по специальности 09.03.01 «Информатика и вычислительная техника» и рекомендуются для использования в учебном процессе.

Рассмотрено на заседании учебно-методической комиссии института КТЗИ от 31 августа 2017 г., протокол №.8

Председатель УМК института КТЗИ _____ В.В. Родионов

Содержание

Введение

Фонд оценочных средств промежуточной аттестации предназначен для оценки запланированных результатов по дисциплине «Конструирование программного обеспечения».

В соответствии с требованиями ФГОС ВПО для аттестации обучающихся на соответствие их персональных достижений поэтапным требованиям соответствующей образовательной программы разработан фонд оценочных средств промежуточной аттестации (ФОС ПА), включающий типовые задания, контрольные вопросы, тесты, позволяющие оценить уровень приобретённых студентом компетенций, знаний и умений, оценить уровень владения полученными навыками.

Фонд оценочных средств является составной частью учебных и методических документов, обеспечивающих реализацию конкретной образовательной программой.

Задачи ФОС ПА по дисциплине «Конструирование программного обеспечения»:

- оценка достижений студентов в процессе изучения дисциплины в соответствии с разработанными и принятыми критериями по каждому виду контроля;
- управление процессом приобретения студентами необходимых знаний, умений, навыков и формирования компетенций, определённых в ФГОС ВПО по соответствующему направлению подготовки.

Основные принципы ФОС ПА по дисциплине «Конструирование программного обеспечения»:

- валидность (объекты оценки соответствуют поставленным целям обучения);
- надёжность (точность, степень постоянства, стабильности, устойчивости результатов оценивания при повторных предъявлениях);
- системность оценивания (циклический характер оценивания);
- соответствие содержания материалов оценочных средств уровню и стадии обучения;
- наличие чётко сформулированных критериев оценки для каждого контрольного мероприятия;
- максимальная объективность используемых процедур и методов оценки;
- использование ФОС ПА не только в качестве средства оценивания, но и обучения.

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

Таблица 1

№ п.п.	№ раздела / модуля	Составляющие формируемых компетенций[1]
1	Модуль 1	ПК-1.З,ПК-1.У,ПК-1.В
2	Модуль 2	ПК-1.З,ПК-1.У,ПК-1.В
3	Модуль 3	ПК-1.З,ПК-1.У,ПК-1.В

2. Виды промежуточной аттестации по дисциплине

Дисциплина изучается в пятом семестре на третьем курсе и завершается следующими видами промежуточной аттестации: в 5 семестре – зачет.

3. Оценочные средства для промежуточной аттестации

Таблица 2

№ п.п.	№ раздела / модуля	Код ФОС ПА	Вид оценочных средств	Примечание	Формируемые компетенции
1	Модули 1-3	ФОСПА-1	ТПА-1, ТПА-2, ТПА-3	Тест промежуточной аттестации (для зачета)	ПК-1
2	Модули 1-3	ФОСПА-1	Билеты для проведения зачета	Письменный ответ и собеседование	ПК-1

4. Описание показателей и критериев оценивания компетенций на различных этапах их формирования, описания шкалы оценивания

Показатели и критерии оценивания сформированности компетенций на зачете, приведены в таблице 3.

Таблица 3

Показатели и критерии оценивания сформированности компетенций на зачете

№ п/п	Этап формирования (семестр)	Код формируемой компетенции (составляющей компетенции)		Критерии оценивания	Показатели оценивания (планируемые результаты обучения)		
					Пороговый уровень	Продвинутый уровень	Превосходный уровень
1.	1	ПК-1	ПК-1.З, ПК-1.У, ПК-1.В	Теоретические навыки	Знание места конструирования программного обеспечения в жизненном цикле ПО, его цели задачи, методы; парадигмы программирования. Знание практических аспектов использования программных сущностей, конвенция именования, понятие стиля программирования; знание и понимание целей создания методов.	Знание порогового уровня. Знание понятий связность и сцепленность модулей. Знание и понимание целей создания методов, связность на уровне методов, способы улучшения связности, способы и принципы организации кода; знание и понимание принципов проектирования классов	Знание продвинутого уровня. Знание и понимание понятия защитное «программирование», знание понятий гибких и «тяжелых» практик программирования; знание и понимание понятий «шаблон проектирования», понятия процедурного типа

2.	1	ПК-1	ПК-1.З, ПК-1.У, ПК-7.В	Практические навыки	Уметь проектировать и кодировать методы, управлять областью видимости и временем жизни переменных, составлять и придерживаться соглашений об именовании; проектировать многомодульное программное обеспечение. Владение навыками проектирования методов, создания выразительного кода и соблюдения стиля кодирования, разработки многомодульного ПО.	Уметь проектировать и кодировать методы с декомпозицией на уровне методов, уметь организовывать программный код; умение управлять связностью и зависимостями классов для снижения сложности ПО. Владеть навыками декомпозиции и организации программного кода, проектирования методов и классов, навыками управления зависимостями программных модулей	Уметь применять методы защитного программирования, уметь применять «методы экстремального программирования»; умение применять шаблон проектирования при разработке классов, уметь объявлять и использовать процедурные типы. Владеть навыками программирования с защитой от ошибок, проектирования с применением шаблонов, методикой программирования с опережающей разработкой тестов, навыком использования процедурных типов

5. Методические материалы, определяющие процедуру оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

Формирование оценки по результатам текущего контроля успеваемости и промежуточной аттестации по итогам освоения дисциплины «Программно-аппаратные средства обеспечения информационной безопасности» приведено в таблице 5.

Таблица 5

Формирование оценки по итогам освоения дисциплины

Наименование контрольного мероприятия	Рейтинговые показатели				
	I аттестация	II аттестация	III аттестация	по результатам текущего контроля	по итогам промежуточной аттестации (зачета /экзамена)
Раздел 1. Введение в КПО	**10**			**10**	
Тест текущего контроля по разделу	10			10	
Раздел 2. Проектирование методов и организация кода		**20**		**20**	
Тест текущего контроля по разделу		10		10	
Защита лабораторных работ		10		10	
Раздел 3. Защитное программирование			**20**	**20**	
Тест текущего контроля по разделу			10	10	
Защита лабораторных работ			10	10	
Промежуточная аттестация (зачет):					**50**
– тест промежуточной аттестации по дисциплине					20
– ответы на контрольные вопросы в письменной форме					30

6. Типовые контрольные задания, оценочные средства освоения учебной дисциплины

6.1. Оценочные средства для промежуточной аттестации ФОСПА

6.1.1.Тестовые вопросы ТПА-1

1. Жизненный цикл программного обеспечения это:
   1) Структура, определяющая последовательность выполнения процессов и их взаимосвязь на протяжении ЖЦ
   2) Весь период разработки и эксплуатации ПО, начиная с момента возникновения идеи и заканчивая прекращением всех видов его использования
   3) Совокупность процессов, связанных с созданием ПО, его реализацией, доставкой пользователю и поддержкой
   4) Совокупность взаимодействующих единиц – элементов, функционирующих совместно для достижения определённых целей.

2. С какими стадиями жизненного цикла связано конструирование ПО:
   1) Формулирование требований
   2) Тестирование и отладка
   3) Анализ предметной области
   4) Проектирование
   5) Эксплуатация и сопровождение
   6) Кодирование
   7) Внедрение

3. Модульное программирование подразумевает:
   1) разделение программы на несколько файлов
   2) разделение программы на несколько компонентов
   3) разделение программы на несколько классов
   4) разделение программы на небольшие независимые части

4. Упорядочить виды сцепления модулей в порядке усиления связи
   1) Сцепление по управлению
   2) Сцепление по общей области
   3) Сцепление по внешним данным
   4) Сцепление по формату
   5) Сцепление по данным

5. Отношение между общей сущностью и ее конкретным воплощением, т.е. один класс является специализацией другого класса:
   1) Отношение ассоциации
   2) Отношение обобщения
   3) Отношение зависимости
   4) Отношение реализации

6. Структурное отношение, показывающее, что объекты одного типа некоторым образом связаны с объектами другого типа, (отношение часть целое):
   1) Отношение реализации
   2) Отношение ассоциации
   3) Отношение зависимости
   4) Отношение обобщения

7. Отношение использования, при котором изменения в спецификации одного класса может повлиять на класс его использующий:
   1) Отношение обобщения
   2) Отношение реализации
   3) Отношение ассоциации
   4) Отношение зависимости

8. Семантическое отношение, при котором класс гарантирует выполнение контракта, определяемого некоторым интерфейсом:
   1) Отношение реализации
   2) Отношение обобщения
   3) Отношение ассоциации
   4) Отношение зависимости

9. Агрегирование является вариантом отношения:
   1) Отношение зависимости
   2) Отношение реализации
   3) Отношение обобщения
   4) Отношение ассоциации

6.1.2.Тестовые вопросы ТПА-2

10. Наследуется только интерфейс:
    1) непереопределяемый метод
    2) переопределяемый метод
    3) абстрактный переопределяемый метод

11. Директива virtual применяется для объявления:
    1) абстрактного переопределяемого метода
    2) непереопределяемый метод
    3) переопределяемого метода

12. Область видимости переменной программы это:
    1) фрагмент программы, в котором переменная известна и может быть использована
    2) общее количество строк, на протяжении которых переменная используется
    3) количество строк между двумя обращениями к переменной

13. Как ограничить видимость переменной одним классом (язык c#)
    1) Объявить переменную класса с директивой public
    2) Объявить переменную класса с директивой private
    3) Объявить переменную класса с директивой protected
    4) Объявить переменную класса с директивой internal

14. «Время жизни» переменной:
    1) фрагмент программы, в котором переменная известна
    2) общее количество строк, на протяжении которых переменная используется
    3) количество строк, в течение которых переменной присвоено значение
    4) время, в течение которого с переменной связано значение

15. Метод имеет функциональная связность когда:
    1) содержит операции, которые выполняются в определенном порядке, каждая из операций использует данные полученные на предыдущих этапах
    2) выбор выполняемой операции осуществляется на основе передаваемого в метод управляющего флага
    3) операции используют одни и те же данные
    4) операции в методе выполняются в определенном порядке, и не имеют другой связи
    5) выполняет одну и только одну операцию
    6) операции объединены в метод на том основании, что все они выполняются в один интервал времени

16. Метод имеет коммуникационную связность когда:
    1) выполняет одну и только одну операцию

2) операции используют одни и те же данные
3) выбор выполняемой операции осуществляется на основе передаваемого в метод управляющего флага
4) содержит операции, которые выполняются в определенном порядке, каждая из операций использует данные полученные на предыдущих этапах
5) операции в методе выполняются в определенном порядке, и не имеют другой связи
6) операции объединены в метод на том основании, что все они выполняются в один интервал времени

17. Метод имеет временную связность когда:
1) выбор выполняемой операции осуществляется на основе передаваемого в метод управляющего флага
2) содержит операции, которые выполняются в определенном порядке, каждая из операций использует данные полученные на предыдущих этапах
3) выполняет одну и только одну операцию
4) операции используют одни и те же данные
5) операции объединены в метод на том основании, что все они выполняются в один интервал времени
6) операции в методе выполняются в определенном порядке, и не имеют другой связи

18. Метод имеет процедурную связность когда:
1) операции используют одни и те же данные
2) содержит операции, которые выполняются в определенном порядке, каждая из операций использует данные полученные на предыдущих этапах
3) операции в методе выполняются в определенном порядке, и не имеют другой связи
4) выбор выполняемой операции осуществляется на основе передаваемого в метод управляющего флага
5) выполняет одну и только одну операцию
6) операции объединены в метод на том основании, что все они выполняются в один интервал времени

19. Метод имеет логическую связность когда:
1) выбор выполняемой операции осуществляется на основе передаваемого в метод управляющего флага
2) содержит операции, которые выполняются в определенном порядке, каждая из операций использует данные полученные на предыдущих этапах
3) операции в методе выполняются в определенном порядке, и не имеют другой связи

4) выполняет одну и только одну операцию
5) операции объединены в метод на том основании, что все они выполняются в один интервал времени
6) операции используют одни и те же данные

6.1.2.Тестовые вопросы ТПА-3

20.В каких случаях управление блоку catch в конструкции try {} catch {}:
1) Всегда
2) При исключительной ситуации в блоке try
3) Никогда

21.В каких случаях управление блоку finally в конструкции try {} finally {}:
1) При исключительной ситуации в блоке try
2) Никогда
3) Всегда

22.Для обнаружения событий, которые не должны произойти следует использовать:
1) Утверждения
2) Исключения
3) Процедуры обработки ошибок

23.Главное требование к форматированию кода программ:
1) стиль форматирования удобен для редактирования кода
2) форматирование показывает логическую структуру программы
3) стиль форматирования визуально привлекателен

24.К какому стилю форматирования относится фрагмент кода

```
if ( pixelColor == Color_Red ) {

  statement1;

  statement2;

}
```

1) Операторные скобки для обозначения границ блоков
2) Эмуляция явных блоков

3) Явные блоки
4) Форматирование в конце строки

25. Какой стиль форматирования наиболее трудоемкий (без использования специальных редакторов):
    1) Эмуляция явных блоков
    2) Форматирование в конце строки
    3) Явные блоки
    4) Операторные скобки для обозначения границ блоков

26. Какой вид документации наиболее подробный и актуальный:
    1) Внутренняя документация в исходном коде
    2) Описание и диаграммы классов
    3) Руководство пользователя

27. Какие из комментариев приемлемы в коде программы:
    1) повторение кода
    2) объяснение кода
    3) резюме в коде
    4) описание цели кода
    5) информацию, которую невозможно выразить в форме кода

6.1.4. Вопросы для проведения зачета

1. Программа, программное изделие. Определения. Жизненный цикл ПО, определение. Каскадная и спиральная модель жизненно цикла.
2. Конструирование программного обеспечения, его задачи.
3. Архитектура программного обеспечения. Его компоненты.
4. Сложность программного обеспечения, причины. Основной способ борьбы со сложностью.
5. Уровни проектирования программного обеспечения.
6. Методы проектирования программного обеспечения.
7. Структурное программирование. Определение. Базовые структуры. Основные положения.
8. Модульное программирование. Основная идея и преимущества подхода. Модуль и его атрибуты.
9. Модульная структура программы. Связность и сцепление модулей. Виды сцепления между модулями.
10. Объектно-ориентированное программирование. Основные положения.
11. Абстрактный тип данных. Определение. Преимущества использования и принципы использования.
12. Интерфейс класса. Абстракция и инкапсуляция. признаки хорошей абстракции и инкапсуляции.
13. Отношения между классами. Типы структурных иерархий.

14. Типы структурных иерархий. Наследование. Варианты наследования методов.
15. Инициализация объекта. Копирование объекта, виды копирования.
16. Причины создания классов. Какие классы следует избегать.
17. Принцип единой ответственности. Пример иллюстрирующий нарушение и следование принципу.
18. Принцип открытости\закрытости. Иллюстрация на примере.
19. Принцип подстановки Лисков. Явное и неявное нарушение принципа. Пример. Следствие из принципа.
20. Принцип подстановки Лисков. Проектирование по контракту.
21. Принцип инверсии зависимости. Пример. Упрощенное правило.
22. Принцип разделения интерфейсов.
23. Компоненты. Принцип эквивалентности повторного использования.
24. Компоненты. Принцип совместного повторного использования.
25. Компоненты. Принцип общей закрытости.
26. Компоненты. Принцип ацикличности зависимостей.
27. Компоненты. Принцип устойчивых зависимостей. Определение устойчивости. Метрики устойчивости.
28. Компоненты. Принцип устойчивых абстракций. Метрика абстрактности компонента
29. Объявление переменных. Инициализация переменных.
30. Область видимости переменных. Интервал использования. Средний интервал. Время жизни. Минимизация области видимости.
31. Длительность существования данных. Основные виды.
32. Связывание переменных. Время связывания, основные виды. Единственность цели переменной.
33. Именование переменных и некоторых видов данных.
34. Конвенция программирования, ее основные элементы. Стандартизированные префиксы.
35. Неудачные имена переменных.
36. Методы. Определение. Причины создания.
37. Связность на уровне методов. Определение. Виды связности.
38. Именование методов.
39. Принципы использования параметры методов.
40. Способы записи алгоритмов. Проектирование с псевдокодом, характеристики псевдокода, этапы проектирования с псевдокодом.
41. Организация последовательного кода. Организация операторов следующих в определенном и произвольном порядке.
42. Организация условных операторов.
43. Циклы, различия циклов разных видов. Управление циклом. Использование переменных циклов.
44. Табличные методы. Определение и основная идея. Виды таблиц.
45. Защитное программирование. Виды ошибок и их последствий.
46. Защитное программирование. Определение. Ошибки операторов ввода вывода. Проверка допустимости промежуточных результатов.

47. Утверждения.
48. Процесс обработки ошибок. Способы обработки ошибок. Корректность и устойчивость программного обеспечения.
49. Исключения. Операторы, связанные с исключениями и их применение.
50. Защита от неправильных данных. Стратегия изоляции повреждений.
51. Предотвращение ошибок накопления погрешности. Ошибки, связанные с использованием целых чисел. Ошибки, связанные с использованием чисел с плавающей запятой.
52. Форматирование кода. Стиль программирования. Интерпретация кода человеком, цели форматирования.
53. Способы форматирования кода. Основные стили форматирования.
54. Принципы форматирования управляющих структур, отдельных операторов, размещение объявления данных. Размещение комментариев.
55. Форматирование методов и классов. Организация файлов и программ.
56. Стиль программирования как вид документации. Виды комментариев. Стиль комментирования.
57. Методики комментирования. Перечислить и описать каждую из методик.
58. Экстремальное программирование.
59. Программирование через тестирование.
60. Унифицированный процесс Rational. Основные используемые виды моделей.
61. Шаблоны проектирования. Определение, основные элементы. Классификация шаблонов.
62. Шаблон абстрактная фабрика.
63. Шаблон одиночка.
64. Шаблон адаптер.
65. Шаблон стратегия.

Лист регистрации изменений

№ п.п.	№ страницы	Дата	Содержание изменений	Ф.И.О. подпись	«Согласовано» зав. ведущей кафедрой ПМИ
1	2	3	4	5	6
