Фонд оценочных средств для промежуточной аттестации по дисциплине (модулю)
«Технологии прикладного программирования»

Направление подготовки (специальность)
09.03.04 «Программная инженерия»

Направленность (профиль)
«Разработка программно-информационных систем»

Присваиваемая квалификация
Бакалавр

Форма обучения
Очная

Год набора
2025

Челябинск, 2025 г.

09.03.04 Программная инженерия, Разработка программно-информационных систем, бакалавр, *Технологии прикладного программирования*, 2025, очная

Фонд оценочных средств дисциплины (модуля) одобрен и рекомендован

Проректор по учебной работе утверждено 24.02.2025 А.А. Саламатов

Ученым советом института информационных технологий

Протокол заседания № 6 от 20.02.2025

Председатель Ученого совета института информационных технологий согласовано Ю. В. Петриченко

Заседанием кафедры информационных технологий и экономической информатики

Протокол заседания № 6 от 20.02.2025

И. о. заведующего кафедрой согласовано С.А. Скрипов

Автор (составитель) А.В. Митянина

Структура рабочей программы соответствует приказу ректора ФГБОУ ВО «ЧелГУ» от «13» апреля 2021 г. № 247-1

Содержание

1. Паспорт фонда оценочных средств

Направление подготовки: 09.03.04 Программная инженерия
Направленность: Разработка программно-информационных систем
Дисциплина: Технологии прикладного программирования
Семестры: 5
Форма промежуточной аттестации: экзамен
Для оценивания результатов обучения используется балльно-рейтинговая система.

2. Перечень формируемых компетенций

Изучение дисциплины «Технологии прикладного программирования» направлено на формирование компетенций, приведённых в 1.

Таблица 1. Результаты обучения по дисциплине.

Коды компетенции согласно ФГОС (ОПОП ВО)	Содержание компетенций согласно ФГОС (ОПОП ВО)	Индикаторы достижения компетенции согласно ОПОП	Перечень планируемых результатов обучения по дисциплине
1	2	3	4
ПК-2	Владение навыками использования различных технологий промышленной разработки программного обеспечения с применением инструментов автоматизации сборки, интеграции, тестирования и развертывания ПО	ПК-2.1. Демонстрирует знание основных принципов и технологий промышленной разработки программного обеспечения ПК-2.2. Демонстрирует умения разрабатывать программное обеспечение с применением инструментов автоматизации сборки, интеграции, тестирования и развертывания ПО ПК-2.3. Имеет практический опыт промышленной разработки программного обеспечения	Знать:основные принципы и технологии промышленной разработки программного обеспечения с использованием языка Java Уметь:разрабатывать программное обеспечение на языке Java с применением инструментов автоматизации сборки, интеграции, тестирования и развертывания ПО Владеть:навыками промышленной разработки программного обеспечения на языке Java

3. Содержание оценочных средств по дисциплине

3.1. Виды оценочных средств

Таблица 2. Виды оценочных средств.

№ п/п	Код компетенции/ планируемые результаты обучения	Контролируемые темы/ разделы	Наименование оценочного средства для текущего контроля	Наименование оценочного средства на промежуточной аттестации/№ задания
1	ПК-2.1. Демонстрирует знание основных принципов и технологий промышленной разработки программного обеспечения Знать:основные принципы и технологии промышленной разработки программного обеспечения с использованием языка Java	Объектно-ориентированная разработка прикладных программ Технологии, фреймворки и жизненный цикл прикладных программ	Тест	Задания теста № 1-3, 8-11, 48-63
2	ПК-2.2. Демонстрирует умения разрабатывать программное обеспечение с применением инструментов автоматизации сборки, интеграции, тестирования и развертывания ПО Уметь:разрабатывать программное обеспечение на языке Java с применением инструментов автоматизации сборки, интеграции, тестирования и развертывания ПО	Объектно-ориентированная разработка прикладных программ Технологии, фреймворки и жизненный цикл прикладных программ	Тест	Задания теста № 35-47, 73-81, 110-128
3	ПК-2.3. Имеет практический опыт промышленной разработки программного обеспечения Владеть:навыками промышленной разработки программного обеспечения на языке Java	Объектно-ориентированная разработка прикладных программ Технологии, фреймворки и жизненный цикл прикладных программ	Тест	Задания теста № 4-7, 12-34, 64-72, 82-109

Типовые задания, критерии и показатели оценивания в рамках текущего контроля представлены в рабочей программе дисциплины (модуля). Полные комплекты оценочных средств и контрольно-измерительных материалов хранятся на кафедре.

3.2. Содержание оценочных средств

База тестовых вопросов

№ п/п	Формулировка вопроса	Варианты ответов (полужирным шрифтом – верные варианты)

1.	Каков результат выполнения фрагмента следующего кода? System.out.println(0.0/0.0 + 12);	**a. NaN** b. 10 c. Infinity d. Ошибка компиляции e. Ошибка времени выполнения
2.	Выберите из перечисленных ниже только примитивные типы данных Java:	**a. byte** **b. short** c. Integer d. String e. bool f. Object
3.	Размерность (ширина в байтах) примитивных типов в Java:	**a. платформенно-независима и строго определена спецификацией** b. зависит от типа платформы, на которой установлена JVM c. зависит от реализации JVM

4.	Что выведет на консоль следующая программа?
	public class Test {
	public static void main(String argv[]) {
	int x = 5, y = 0;
	try {
	System.out.print(x/y);
	}
	catch(Exception e) {
	System.out.print("Exception");
	}
	catch(ArithmeticException e) {
	System.out.print(" ArithmeticException");
	}
	}}
		a. Infinity Exception
		b. Exception
		c. ArithmeticException
		d. Exception ArithmeticException
		e. Ошибка компиляции: Unreachable catch block for ArithmeticException

5.	Какой результат вызова следующего метода с использованием параметров a = 0, b = 3?
	public void divide(int a, int b) {
	try {
	int c = a / b;
	} catch (Exception e) {
	System.out.print("Exception ");
	} finally {
	System.out.print("Finally");
	}
	}
		a. Будет выведено: "Exception Finally"
		b. Будет выведено: "Exception "
		c. Будет выведено: "Finally"
		d. Ошибка компиляции

6.	Какой результат вызова следующего метода с использованием параметров a = 5, b = 0?
	public void divide(int a, int b) {
	try {
	try {
	int c = a / b;
	} catch (NullPointerException e){
	System.out.print("NullPointerException ");
		a. Будет выведено: "NullPointerException Finally"
		b. Будет выведено: "Finally ArithmeticException Exception"
		c. Будет выведено: "Finally Exception"

	} finally {
	System.out.print("Finally ");
	}
	} catch (ArithmeticException e){
	System.out.print("ArithmeticException");
	} catch (Exception e){
	System.out.print("Exception");
	}
	}
		d. Ошибка компиляции
		e. Будет выведено:
		"Finally "
		**f. Будет выведено:
		"Finally ArithmeticException"**

7.	Что будет выведено на консоль в результате выполнения следующего кода:
	public class Test {
	public static void main(String argv[]) {
	int[] array = null;
	try{
	System.out.print(array.length);
	}
	catch(NullPointerException e) {
	System.out.print("NullPointerException");
	}
	catch(Exception e) {
	System.out.print(" Exception");
	}
	}}
		a. 0 NullPointerException
		b.
		null NullPointerException
		c. NullPointerException
		d. NullPointerException Exception
		e.
		null NullPointerException Exception
		f. Ошибка компиляции
		g. 0

8.	Укажите недостатки оператора assert, встроенного в Java	**a. Необходимо включать флагом -ea** **b. Может проверить только логические выражения** c. Потребляет много процессорного времени d. Не дает определить строку в которой произошла ошибка e. Доступен только в JavaEE f. Не совместим с JUnit
9.	Какие этапы включает в себя техника разработки TDD?	**a. Написание теста** **b. Написание кода** **c. Рефакторинг кода** d. Отладка e. Написание документации f. Написание технического задания g. Публикация
10.	Для чего используются тестовые двойники в модульном тестировании?	**a. Для отвязывания кода от внешних сервисов** **b. При опосредованном вводе/выводе** c. При высокой цикломатической сложности d. Когда результат работы кода не известен заранее

		e. Для доступа к инкапсулированным в модуле данным
11.	Какой аннотацией помечаются методы соответствующие тестам в JUnit?	**a. @Test** **b. Test**

12.	Что будет выведено на консоль после выполнения следующего кода:
	public class Example {
	public static void main(String[] args) {
	boolean a = true;
	boolean b = false;
	boolean c = true;
	if (a || b && c)
	System.out.print("Hello ");
	if (a & !b & c)
	System.out.println("World");
	}}
		a. На консоль будет выведено: "Hello "
		b. На консоль будет выведено: "World"
		c. На консоль будет выведено: "Hello World"
		d. Ошибка компиляции
		e. Ошибка времени выполнения

13.	Что выведет на консоль следующая программа:
	public class TestClass {
	public static boolean methodOne() {
	System.out.println("methodOne ");
	return false;
	}
	public static boolean methodTwo() {
	System.out.println("methodTwo ");
	return true;
	}
	public static boolean methodThree() {
	System.out.println("methodThree ");
	return true;
	}
	public static void main(String[] args) {
	System.out.println(methodOne() || methodTwo() || methodThree());
	}}
		a. methodOnemethodTwomethodThreetrue
		b. methodOnemethodTwotrue
		c. methodOnetrue
		d. true
		e. false
		f. Возникнет ошибка компиляции

14.	Что будет выведено на консоль в результате выполнения следующего фрагмента кода:
	int a = 9;
	int b = 2;
	System.out.println(-a % b);
	System.out.println(a % -b);
	System.out.println(a % b == a % -b);
		a. 11true
		b. -11true
		c. -1-1false
		d. Ошибка компиляции или времени выполнения

15.	Возможна ли перегрузка операторов в Java для пользовательских типов данных?	a. Возможна при любых условиях **b. Нет** c. Возможна при определенных условиях для некоторых операторов

16.	Что выведется на консоль в результате выполнения данной программы:
	public class Main {
	public static void main(String[] args) {
	int i = 0;
	int j = 1;
	System.out.println(i += (j < i) ? (2) : (3));
		a. 1
		b. 2
		c. 3
		d. Ошибка компиляции
		e. Ошибка времени выполнения

	}}	

17.	Что будет выведено на консоль в результате выполнения следующей программы?
	public class Example {
	public static void main(String[] args) {
	int i = 011;
	i <<= 34;
	System.out.println(i);
	}}
	(Если на ваш взгляд, результатом компиляции или выполнения данной программы будет ошибка, то введите слово "ошибка")
		a. 36

18.	Что будет выведено на консоль в результате выполнения следующей программы?
	public class Example {
	public static void main(String[] args) {
	short s = -8;
	s >>>= 2;
	System.out.println(s);
	}}
	(Если на ваш взгляд, результатом компиляции или выполнения данной программы будет ошибка, то введите слово "ошибка")
		a. -2

19.	Что будет выведено на консоль в результате выполнения данной программы:
	public class Main {
	public static void main(String[] args) {
	p1: {
	p2: {
	p3: {
	System.out.print("p3.1 ");
	if (true) break p2;
	System.out.print("p3.2 ");
	}
	System.out.print("p2 ");
	}
	System.out.print("p1 ");
	}}}
		a. p3.1
		b. p3.1 p3.2 p2 p1
		c. p3.1 p2
		d. p3.1 p1
		e. Ошибка компиляции или времени выполнения

20.	У каких из приведенных ниже операторов всегда вычисляются все операнды?	**a. %** b. && c. || d. ? : **e. <<**

21.	Что произойдет в результате выполнения этой программы:
	public class Main {
	public static void main(String[] args) {
	byte b = 0;
	while (--b < 0);
	System.out.println(b);
	}}
		a. На консоль будет выведено: 0
		b. На консоль будет выведено: -128
		c. На консоль будет выведено: 127
		d. На консоль будет выведено: 128
		e. Произойдет зацикливание программы

22.	Что произойдет в результате выполнения этой программы:
	public class Main {
	public static void main(String[] args) {
		a. На консоль будет выведено: 0
		b. На консоль будет

	byte b = 0;
	while (++b > 0);
	System.out.println(b);
	}}
		выведено: -127
		c. На консоль будет выведено: 127
		d. На консоль будет выведено: -128
		e. Произойдет зацикливание программы

23.	Каким будет значение переменной count после выполнения следующего фрагмента кода:
	int count = 1, i = 0;
	do { count *= ++i; if (count > 5) break;
	}
	while (i <= 4);
		a. 6

24.	Каким термином лучше всего можно описать отношение между классами Department и Employee? class Department { private int departmentld; private Set<Employee> employees;} class Employee { private int employeeld; private String employeeName;}	a. Ассоциация b. Декомпозиция **c. Агрегация** d. Обобщение e. Ни один из перечисленных ответов

25.	Какие из приведенных символов будут выведены на консоль в результате выполнения программы:
	public class MyClass {
	public static void main(String[] args) {
	B b = new C();
	A a = b;
	if (a instanceof A) System.out.println("A");
	if (a instanceof B) System.out.println("B");
	if (a instanceof C) System.out.println("C");
	if (a instanceof D) System.out.println("D");
	}}
	class A {}
	class B extends A {}
	class C extends B {}
	class D extends C {}
		a. A
		b. B
		c. C
		d. D
		e. Ничего не будет выведено

26.	Какие из строк можно вставить вместо комментария в следующем коде, чтобы он успешно скомпилировался и выполнился без ошибок?
	class A {}
	class B extends A {}
	class C extends B {
	public static void main(String[] args) {
	A obj1 = new A();
	C obj2 = new C();
	// какую строку нужно сюда вставить?
	C obj3 = (C) obj1;
	}}
		a. obj1 = obj2;
		b. obj1 = (B) obj2;
		c. obj1 = new C();
		d. obj1 = (C) new B();
		e. Код и так компилируется и выполняется без ошибок

27.	Как наиболее точно можно назвать способ повторного использования классов, при котором: объект существующего класса включается в создаваемый класс и при этом методы встроенного объекта становятся доступны извне через методы создаваемого класса?	a. композиция b. наследование **c. делегирование** d. агрегация e. полиморфизм

28.	Что выведет на консоль следующая программа?
	class A {
	public A(String s) {
		a. ABA
		b. ABEmptyA
		c. AB

	System.out.print("A");
	}}class B extends A {
	public B() {
	System.out.print("Empty");
	}
	public B(String s) {
	System.out.print(s);
	}
	public static void main(String[] args) {
	new B("AB");
	}}
		d. ABEmpty
		e. Ошибка компиляции

29.	Что выведет на консоль следующая программа?
	class SuperClass {
	private SuperClass() {
	System.out.print("1");
	}
	protected SuperClass(String str) {
	System.out.print("2");
	}}
	public class SubClass extends SuperClass {
	public SubClass() {
	this("smth");
	}
	public SubClass(String str) {}
	public static void main(String[] args) {
	SubClass a = new SubClass();
	}}
		a. 1
		b. 2
		c. Ошибка компиляции
		d. Ошибка времени выполнения

30.	При наследовании подкласс и суперкласс могут иметь... (выберите все ответы, которые вы считаете правильными)	**a. одноименные поля** **b. одноименные методы, но с разными списками параметров** **c. одноименные методы с одинаковыми списками параметров** d. нет правильных ответов
31.	Что будет выведено на консоль в результате выполнения следующей программы? class Building { Building() { System.out.print("building "); } Building(String name) { this(); System.out.print("buildingName " + name); }}public class House extends Building { House() { System.out.print("house "); } House(String name) { this(); System.out.print("houseName " + name); } public static void main(String[] args) { House h = new House("MyHouse "); }}	a. building houseName MyHouse house b. building buildingName MyHouse house houseName MyHouse c. buildingName MyHouse house houseName MyHouse d. house houseName MyHouse **e. building house houseName MyHouse** f. houseName MyHouse house

32.	Что выведет на консоль следующий код?
	class A {
	public A() {
	System.out.print("A");
	}}
	class B extends A {
		a. C
		b. BC
		c. ABC
		d. CBA
		e. Ошибка компиляции или времени выполнения

	public B() {
	System.out.print("B");
	}}
	class C extends B {
	public C() {
	System.out.print("C");
	}
	public static void main(String[] args) {
	C c = new C();
	}}	

33.	Что выведет на консоль следующая программа?
	class A {}
	class B extends A {}
	class C extends B {}
	public class Test {
	public static void m(A a) {
	System.out.println("a");
	}
	public static void m(B b) {
	System.out.println("b");
	}
	public static void main(String[] args) {
	m(new C());
	}}
		a. a
		b. b
		c. Результат может варьироваться от запуска к запуску
		d. Ошибка компиляции
		e. Ошибка времени выполнения

34.	Даны следующие классы: class Parent {} class DerivedOne extends Parent {} class DerivedTwo extends Parent {} Выберите единственное верное высказывание о следующем блоке кода: Parent p = new Parent(); DerivedOne dl = new DerivedOne(); DerivedTwo d2 = new DerivedTwo(); dl = (DerivedOne)d2;	a. Ошибка компиляции b. Ошибка выполнения: исключение RuntimeException **c. Ошибка выполнения: исключение ClassCastException** d. Успешный запуск и отработка
35.	Какие из приведенных утверждений истинны?	**a. абстрактный метод класса не может быть определен как final** **b. final-методы суперкласса нельзя переопределять в подклассах** c. абстрактные методы суперкласса нельзя переопределять в подклассах **d. если класс имеет хотя бы один абстрактный метод, то он должен быть объявлен как абстрактный класс** e. если класс имеет хотя бы один final метод, то он должен быть объявлен как final-класс
36.	Какие из приведенных утверждений истинны?	a. Подклассы наследуют только public- и protected-

		методы и поля суперкласса b. Подклассы наследуют конструкторы суперкласса c. Подклассы наследуют инициализаторы суперкласса d. Подклассы наследуют все методы суперклассов кроме абстрактных методов **e. Все утверждения ложные**
37.	Выберите только истинные утверждения: У любого пользовательского класса всегда есть ...	**a. суперкласс** b. хотя бы один интерфейс, который он реализует **c. хотя бы один явный конструктор или конструктор по умолчанию** d. хотя бы один метод или поле e. хотя бы один подкласс
38.	Что произойдет при компиляции следующего кода? class A extends B {} class B extends C {} class C extends A {}	a. Компиляция пройдет успешно b. Компилятор зациклится **c. Ошибка компиляции**
39.	Укажите метод, вызов которого не является нежелательным (Deprecated)	a. Thread.stop() b. Thread.suspend() **c. Thread.interrupt()** d. Thread.resume()
40.	Укажите основной недостаток следующего кода: while (!condition) {}Code Formatted by ToGoTutor	**a. Потребляет много процессорного времени** b. Потребляет много оперативной памяти c. Заканчивает работу сразу после выполнения условия d. Требует чтобы текущий поток был разбужен вызовом notify() e. Не дает возможности отслеживать несколько условий одновременно

41.	Что выведет на экран следующий код:
	public class Concurrent {
	public static void main(String[] a) {
	new Thread() {
	public void run() {
	System.out.print("A");
		a. AB
		b. BA
		c. AB или BA
		d. Нет правильного ответа

	}
	}
	.start();
	new Thread() {
	public void run() { System.out.print("B");
	}
	}
	.start();
	} }	

42.	Что выведет на экран следующий код:
	public class Concurrent {
	public static void main(String[] a) {
	new Thread() {
	synchronized public void run() {
	System.out.print("A"); }
	}
	.start();
	new Thread() {
	synchronized public void run() { System.out.print("B");
	}
	}
	.start();
	} }
		a. AB
		b. BA
		c. AB или BA
		d. Нет правильного ответа

43.	Какое ключевое слово используется для получения блокировки (монитора) объекта?	**a. synchronized** b. notify c. wait d. monitor e. deadlock
44.	Укажите основной недостаток следующего кода: while (!condition) wait();Code Formatted by ToGoTutor	a. Потребляет много процессорного времени b. Потребляет много оперативной памяти c. Заканчивает работу сразу после выполнения условия **d. Требует чтобы текущий поток был разбужен вызовом notify()** e. Не дает возможности отслеживать несколько условий одновременно
45.	Укажите метод, который временно приостанавливает выполнение потока	a. Thread.stop() **b. Thread.suspend()** c. Thread.interrupt() d. Thread.resume()

46.	Что может вывести на экран следующий код:
	public class Concurrent extends Thread{
	static int counter = 0;
	public void run() {
	Concurrent.counter++;
	}
	public static void main(String[] a) throws Exception {
		a. -1
		b. 0
		c. 1
		d. 2
		e. 3
		f. 4
		g. 10

	new Concurrent().start(); new Concurrent().start(); System.out.println(counter); } }	h. 2147483647 i. -2147483647 j. -2
47.	Grady Booch defines an object as, “An object has state, behavior, and <???>"	**a. identity** b. persistence c. abstraction d. superclass e. encapsulation
48.	Словом "агрегация" точнее всего описывается отношение между ...	a. вашей комнатой и комнатой ваших соседей b. вами и вашими руками **c. вашей комнатой и мебелью в ней** d. вами и вашими друзьями
49.	Почему в некоторых языках программирования (например, в Java) отказываются от поддержки множественного наследования в чистом виде (имеется в виду наследование реализации)?	a. Поддержка множественного наследования ведет к большим потерям производительности, так как для каждого класса необходимо держать сильно-ветвящуюся иерархию его предков b. Множественное наследование практически никогда не используется, в отличии от обычного наследования от одного класса **c. Из-за неоднозначности выбора поведения, в случае если суперклассы некоторого класса содержат методы с одинаковыми сигнатурами** d. Множественное наследование невозможно реализовать с помощью таблицы виртуальных функций, поэтому требуются другие намного более сложные алгоритмы e. Реализация механизма множественного наследования не представляет сложности и в поздних версиях языка Java есть множественное наследование классов без ограничений

50.	Верно ли утверждение: Наследование и композиция взаимоисключающие понятия. То есть при создании иерархии объектов и классов в программе используется либо наследование, либо композиция.	a. Да **b. Нет**
51.	Объект, который НЕ взаимодействует с другими объектами в рассматриваемой предметной области ...	**a. не существует** b. выделяется в отдельный класс c. определяется в отдельный уровень абстракции d. ограниченно участвует в общей иерархии объектов и классов
52.	Как называется способность объекта скрывать свои данные и реализацию от других объектов системы?	**a. Инкапсуляция** b. Наследование c. Полиморфизм d. Композиция e. Абстракция
53.	Класс - это группа объектов, имеющих на рассматриваемом участке предметной области ...	**a. одинаковые атрибуты** **b. одинаковое поведение** c. одинаковые адреса в памяти программы d. одинаковое время жизни e. одинаковую область видимости
54.	Выберите наиболее подходящее определение понятия "класс" в ООП:	a. Тип, описывающий поведение некоторой сущности **b. Тип, описывающий характеристики и поведение группы объектов** c. Тип, который отображает все возможные состояния объекта d. Тип, содержащий набор функций e. Тип, включающий в себя массив всех возможных объектов своего типа
55.	Является ли импортирование пакета java.awt, записанное в программе следующим образом: import java.awt.*; достаточным для использования классов вложенного пакета java.awt.event (без указания названия пакета)?	a. Да, является достаточным **b. Нет, не является достаточным**
56.	Классы какого пакета (или пакетов) импортируются в Java-приложение по умолчанию?	**a. java.lang.*** b. java.io.* c. java.util.* d. java.awt.*

		e. ни один из перечисленных пакетов не импортируется по умолчанию
57.	Верно ли следующее утверждение и если нет, то почему: В случае использования в модуле кода некоторого класса, определенного в другом пакете, его всегда необходимо импортировать.	a. Да, это утверждение верно **b. Нет, необязательно импортировать, можно использовать полное имя класса** c. Нет, необязательно импортировать, если этот класс находится во вложенном пакете по отношению к нашему классу d. Нет, необязательно импортировать, если этот класс объявлен с модификатором public, то его можно использовать по простому имени
58.	Использование спецификации импорта всех классов пакета (например: import java.awt.*;) при импорте больших пакетов может привести к:	**a. увеличению времени компиляции** b. увеличению времени запуска программы c. снижению быстродействия программы в процессе работы d. никак не повлияет на процесс запуска и выполнения или на время компиляции программы
59.	Каким путем абстрагирование позволяет бороться со сложностью систем?	a. путем разделения сложной системы на подсистемы и элементарные части **b. путем выделения важных деталей и существенных связей в сложной системе** c. путем упорядочения родственных подсистем и элементов сложной системы по уровням d. путем строгого разделения этапов создания сложной системы
60.	Каким путем декомпозиция позволяет бороться со сложностью систем?	**a. путем разделения сложной системы на подсистемы и**

		элементарные части b. путем выделения важных деталей и существенных связей в сложной системе c. путем упорядочения родственных подсистем и элементов сложной системы по уровням d. путем обобщения одинакового поведения и состояния одинаковых элементов и подсистем сложной системы
61.	Каким путем иерархия позволяет бороться со сложностью систем?	a. путем разделения сложной системы на подсистемы и элементарные части b. путем выделения важных деталей и существенных связей в сложной системе **c. путем упорядочения родственных подсистем и элементов сложной системы по уровням** d. путем строгого разделения этапов создания сложной системы
62.	Какой признак сложной системы по Саймону наиболее близко соответствует концепции инкапсуляции при создании сложных программных систем.	a. Сложные системы часто являются иерархическими. b. Выбор того, какие компоненты в данной системе считаются элементарными, относительно произволен. **c. Внутриэлементная связь обычно сильнее, чем связь между элементами.** d. Иерархические системы обычно состоят из немногих типов подсистем, по-разному скомбинированных и организованных. e. Любая работающая сложная система является результатом развития работавшей более простой системы.
63.	Выберите все истинные утверждения из приведенных ниже:	**a. В отличие от**

		интерфейсов абстрактные классы могут содержать не только абстрактные методы, но и методы с реализацией **b. Интерфейсы могут расширять друг друга и формировать собственную иерархию наследования** c. В отличие от интерфейсов абстрактные классы могут иметь только статические константные (final) поля **d. У интерфейсов все методы могут быть только общедоступными (public)** e. Все утверждения ложные

64.	Что выведет на консоль следующая программа?
	public class Fruit{
	public Fruit() {
	System.out.println("Constructor of Fruit");
	}
	void method() {
	System.out.println("Method of Fruit");
	}
	public static void main(String[] args) {
	Fruit f = new Apple();
	f.method();
	}}
	class Apple extends Fruit {
	public Apple() {
	System.out.println("Constructor of Apple");
	}
	protected void method() {
	System.out.println("Method of Apple");
	}}
		a. Constructor of AppleMethod of Apple
		b. Constructor of FruitConstructor of AppleMethod of Apple
		c. Constructor of FruitConstructor of AppleMethod of Fruit
		d. Ошибка компиляции: класс Apple изменил видимость метода при переопределении

65.	Какой принцип ООП необходимо использовать, чтобы заменить конструкции if-then-else в данном фрагменте кода:
	if (animal.IsCat()) {
	/* код */
	}
	else if (animal.IsDog()) {
	/* код */
	}
	else if (animal.IsKoala()) {
	/* код */
	}else if (animal.isHorse()) { /* код */ }
		a. Инкапсуляция
		b. Агрегация
		c. Полиморфизм
		d. Композиция

66.	Дан следующий код:
	interface Printable {
	void print(String s);}
	class ConsolePrinter implements Printable {
		a. Printable p = new ConsolePrinter();
		b. Printable p = new Printable();

	public void print(String s) { System.out.println(s); } } Выберите из приведенных ниже строчек кода те, которые могут быть откомпилированы без ошибок:	**c. ConsolePrinter p = new ConsolePrinter();** d. ConsolePrinter p = new Printable(); e. Во всех приведенных строчках кода будут ошибки компиляции

67.	Что выведет на консоль следующая программа?
	class Dog {
	public void bark() {
	System.out.print("woof ");
	}}
	class Hound extends Dog {
	public void sniff() {
	System.out.print("sniff ");
	}
	public void bark() {
	System.out.print("howl ");
	}}
	public class DogShow {
	public static void main(String[] args) {
	Hound h = new Hound();
	h.bark();
	((Dog) h).bark();
	((Dog) h).sniff();
	}}
		a. Выведет: "howl howl". Затем будет ошибка времени выполнения
		b. Выведет: "howl woof". Затем будет ошибка времени выполнения
		c. Выведет: "howl woof sniff".
		d. Выведет: "howl howl sniff".
		e. Ошибка компиляции на строке: ((Dog) h).bark();
		f. Ошибка компиляции на строке: ((Dog) h).sniff();

68.	Как откомпилируется и что выведет на консоль следующая программа?
	interface Printable1 {
	public void print();
	}
	interface Printable2 {
	public void print();}
	class SuperPrinter implements Printable1 {
	public void print() {
	System.out.println("Hello world!");
	}}
	class SubPrinter extends SuperPrinter implements Printable2 { }
	public class MyClass {
	public static void main(String args[]) {
	SubPrinter printer = new SubPrinter();
	printer.print();
	}}
		a. Во время выполнения произойдет ошибка на строке printer.print(), т.к. в классе SubPrinter нет метода print()
		b. Код успешно откомпилируется и напечает "Hello World!"
		c. Код не откомпилируется, т.к. нет собственной реализации метода print() у класса SubPrinter
		d. Код не откомпилируется по некоторой другой причине
		e. Будет ошибка времени выполнения по некоторой другой причине

69.	Дан интерфейс A:
	interface A {
	void methA();
	}
	interface B {
	void methB();
	}
	Какие способы расширения этих интерфейсов из приведенных ниже допустимы (не вызовут ошибки компиляции)?
		a. В Java нельзя расширять интерфейсы, их можно только реализовывать в классах
		b. interface C implements A { void methC();}
		c. interface C implements A, B { void methC();}
		d. interface C extends A,

		B { void methC();} **e. interface C extends B { void methC();}**

70.	Что выведет на консоль следующая программа:
	class A {
	String name = "a ";
	String test() {
	return "test A ";
	}}
	class B extends A {
	String name = "b ";
	String test() {
	return "test B ";
	}}
	public class Main {
	public static void main(String[] args) {
	A m = new B();
	System.out.println(m.name + m.test());
	}}
		a. a Test A
		b. a Test B
		c. b Test A
		d. b Test B
		e. b Test A Test B
		f. Ошибка компиляции или времени выполнения

71.	При переопределении метода суперкласса в подклассе обязательно необходимо, чтобы у нового метода в подклассе и метода в суперклассе ...	**a. совпадали имена методов** **b. совпадали списки параметров** c. совпадали типы возвращаемых значений d. совпадали модификаторы видимости

72.	Возможна ли компиляция следующего фрагмента кода?
	public class TestClass {
	private int count = 5;
	public int getCount() { return count; }
	public static void main(String[] args) {
	TestClass test = new TestClass();
	test.count = 10;
	System.out.println(test.count);
	}}
		a. Да, код будет откомпилирован без ошибок
		b. Нет, возникнет ошибка компиляции, т.к. в методе main() происходит прямое обращение к private-полю объекта
		c. Нет, возникнет ошибка компиляции, т.к. нельзя создавать объект класса со статическим методом main()
		d. Нет, возникнет ошибка компиляции, т.к. для поля count не определен метод setCount()
		e. Нет, возникнет ошибка компиляции, т.к. в статическом методе main() выполняется обращение к нестатическому полю объекта класса

73.	Какой механизм (или механизмы) в объектно-ориентированных языках (например, в Java) позволяет (-ют) обеспечить инкапсуляцию объектов?	a. Статические методы **b. Модификаторы видимости**

		c. Сборщик мусора d. Обработка исключений e. Переопределение методов
74.	Какой механизм (или механизмы) в языке Java позволяет (-ют) обеспечить инкапсуляцию объектов?	a. Статические методы **b. Модификаторы видимости** **c. Пакеты** d. Обработка исключений e. Переопределение методов

75.	Какой принцип ООП нарушает следующий фрагмент кода?
	class Counter {
	public int count;
	public void increment() {
	count++;
	}
	public int getCount() { return count; }}
	public class Main {
	public static void main(String[] args) {
	Counter counter = new Counter();
	counter.count = 5;
	}}
		a. Композиция
		b. Полиморфизм
		c. Наследование
		d. Инкапсуляция
		e. Абстракция

76.	Как определить длину массива array?	**a. array.length** b. array.size c. array.length() d. array.getLength() e. array.size()
77.	Выберите все истинные утверждения для следующей строки кода: int[] x = new int[25];	a. x[25] == 0 b. значение x[24] не определено c. x[0] == null **d. x.length == 25** **e. x[24] == 0**

78.	Выберите корректные варианты объявления и инициализации массивов:	**a. int array[] = {1,2,3,4,5};**
		b. long array[5];
		c. int[] array = new char[5];
		d. boolean array = new boolean[10];
		e. String[] array = null;

79.	Для запуска Java-приложений на какой-либо платформе (процессор+ОС), для нее должна быть создана:	**a. своя виртуальная Java-машина (JVM)** b. свой Java-компилятор c. ничего из перечисленного
80.	Выберите особенности, которые характерны для Java-апплетов:	a. исполняются на сервере в специальной среде выполнения **b. исполняются в web-браузере со встроенным**

		java-plugin'ом c. имеют доступ ко всем системным ресурсам клиентской станции **d. автоматически загружаются с web-сервера при открытии web-страницы** **e. являются платформенно-независимыми**
81.	Выберите особенности, которые характерны для сервлетов:	**a. исполняются на сервере в специальной среде исполнения** b. исполняются в web-браузере со встроенным java-plugin'ом c. обрабатывают все запросы только в одном потоке **d. функционируют по принципу запрос/ответ** **e. расширяют функциональные возможности сервера**
82.	Выберите ВЕРНЫЕ утверждения о старте Java-приложений:	**a. При запуске Java-приложения всегда сначала стартует виртуальная Java-машина** b. Перед запуском приложения необходима перекомпиляция кода на Java под целевую платформу c. JIT-компиляция байт-кода выполняется до старта JVM **d. Виртуальная Java-машина (JVM) не только исполняет байт-код, но и взаимодействует с операционной системой**
83.	Что относится к функциям Java-компилятора (утилиты javac из комплекта JDK)?	a. JIT-компиляция байт-кода перед стартом Java-программы **b. Компиляция кода на Java в кроссплатформенный байт-код** c. Компиляция кроссплатформенного байт-кода в машинный

		код для целевой платформы d. JIT-компиляция и оптимизация байт-кода во время выполнения Java-программы в виртуальной Java-машине (JVM)
84.	Какие из перечисленных ниже функций выполняет виртуальная Java-машина (JVM)?	**a. интерпретация байт-кода в машинный код целевой платформы** **b. JIT-компиляция байт-кода и исполнение JIT-компилированного кода** **c. взаимодействие с операционной системой, запрос системных ресурсов** **d. Сборка мусора - автоматическое высвобождение памяти, занятой ненужными объектами** e. Компиляция кода на Java в кроссплатформенный байт-код
85.	Что из перечисленного ниже входит в комплект разработки программного обеспечения на Java - Java Development Kit (JDK)?	**a. Виртуальная Java-машина (JVM)** **b. Набор стандартных библиотек** **c. Java-компилятор** d. Интегрированная среда разработки NetBeans e. Интегрированная среда разработки Eclipse
86.	Что необходимо для запуска Java-приложения?	**a. Среда исполнения Java: JRE** b. Java-компилятор (утилита javac из комплекта JDK) c. Никакого специального ПО для запуска java-приложения не требуется
87.	Язык Java ...	a. является кроссплатформенным языком только на уровне компиляции, то есть для Java необходимо создавать компилятор под каждую платформу b. является кроссплатформенным языком на уровне

		выполнения, то есть файлы с исходным кодом можно запускать на различных платформах без компиляции c. не является кроссплатформенным языком **d. является кроссплатформенным языком на уровне промежуточного кода, то есть исполняемые файлы с промежуточным кодом можно запускать на различных платформах без перекомпиляции**
88.	Какой интерфейс каркаса коллекций (пакет java.util) определяет спецификацию методов для хранения и обработки множества пар типа "ключ-значение"?	**a. Map** b. List c. Collection d. SortedSet e. Queue
89.	Экземпляры каких классов стандартного каркаса коллекций (пакет java.util) следовало бы использовать из соображений производительности для хранения последовательности значений, если в программе преобладают операции вставки/удаления элементов списка?	a. ArrayList b. List **c. LinkedList** d. HashMap e. Vector
90.	Экземпляры каких классов стандартного каркаса коллекций (пакет java.util) следовало бы использовать из соображений производительности для хранения последовательности значений, если в программе преобладают операции извлечения элементов по индексу?	**a. ArrayList** b. List c. LinkedList d. HashMap e. Stack
91.	Какой класс каркаса коллекций (пакет java.util) подходит для хранения множества значений в отсортированном порядке?	a. HashSet b. Set c. LinkedHashSet **d. TreeSet**
92.	Какой класс каркаса коллекций (пакет java.util) подходит для хранения множества значений, если в процессе работы необходимо будет извлекать из множества элементы в том порядке, в каком они были в коллекцию добавлены.	a. HashSet b. Set **c. LinkedHashSet** d. TreeSet

93.	Что будет выведено на консоль в результате выполнения следующей программы:
	public class Main {
	public static void main(String[] args) {
	int array[] = { 1, 2, 3, 4, 5 };
	for (int i : array) { i = 0; }
	for (int i : array) { System.out.print(i);
	}
	}}
		a. 00000
		b. 12345
		c. 10000
		d. 02345
		e. Ошибка компиляции или ошибка времени выполнения

94.	Что выведется на консоль в результате выполнения следующей программы:
	public class Main {
	public static void main(String[] args) {
	int s; int array[] = {1,2,3,4,5};
	for(int i : array) { s += i; }
	System.out.println(s);
	}}
		a. 1
		b. 5
		c. 15
		d. Ошибка компиляции
		e. Ошибка времени выполнения

95.	К какому типу программного обеспечения можно отнести виртуальную java-машину (JVM)?	**a. Системное ПО** b. Прикладное ПО
96.	Выберите ВЕРНЫЕ утверждения:	a. Для запуска скомпилированной программы необходимо наличие установленного компилятора b. Трансляторы всегда преобразуют программу из текстовой формы в двоичные коды c. При компиляции первоначальный набор инструкций переводится в исполняемую форму при каждом запуске программы **d. При интерпретации набор инструкций переводится в исполняемую форму при каждом запуске программы** **e. Для запуска интерпретируемой программы необходимо наличие установленного интерпретатора**
97.	Выберите только ВЕРНЫЕ утверждения:	a. Прикладное ПО должно обеспечивать эффективное управление компонентами вычислительной системы **b. Системное ПО обеспечивает работу других программ** **c. Для прикладного ПО важно обеспечение необходимой функциональности в конкретной предметной области** **d. Требования к прикладным программам принципиально отличаются от**

		требований к системным программам e. Прикладное ПО обычно использует аппаратные ресурсы компьютера напрямую **f. Для прикладного ПО быстродействие и занимаемые системные ресурсы не имеют значения до тех пор, пока не влияют на функциональность**
98.	Выберите ВЕРНОЕ утверждение о скорости исполнения интерпретируемых и скомпилированных инструкций программы:	a. Скомпилированные коды исполняются медленнее, чем интерпретируемые **b. Скомпилированные коды исполняются быстрее, чем интерпретируемые** c. Скомпилированные и интерпретируемые коды исполняются всегда примерно с одинаковой скоростью
99.	Как называется свойство объекта, которое отличает его от всех других объектов?	**a. Идентичность** b. Персистентность c. Неизменяемость d. Абстрагированность e. Инкапсулированность

100.	Выберите варианты корректного объявления и инициализации переменных:	a. int goto = 45;
		b. long $a = 5;
		c. float f = 3.14;
		d. double d = 12.5d;
		e. char cc = '\u1020';

101.	Напишите недостающее слово в определении, данном Гради Бучем: <???> обладает состоянием, поведением и идивидуальностью.	**a. объект**

102.	Что выведется на консоль при выполнении следующей программы:
	`class TestClass {`
	`  int i = getInt();`
	`  int k = 20;`
	`  public int getInt() {`
	`  return k + 1;`
	`  }`
	`  public static void main(String[] args) {`
	`  TestClass t = new TestClass();`
	`  System.out.println(t.i + " " + t.k);`
	`  }}`
		a. 1 20
		b. 21 20
		c. Ошибка компиляции
		d. Ошибка времени выполнения
		e. Ничего из перечисленного не произойдет

103.	Какие ключевые слова могут быть применимы как к полям, так и к методам класса?	**a. public** b. abstract **c. static**

		d. void **e. final**

104.	Что выведет на консоль следующая программа?
	class Hello {
	protected String helloWorld = "Hello";
	}
	public class HelloWorld extends Hello {
	public static void main(String[] args) {
	helloWorld += " world!";
	System.out.println(helloWorld);
	}}
		a. Hello world!
		b. world!
		c. null
		d. Возникнет ошибка компиляции
		e. Программа пройдет компиляцию, но при запуске на консоль ничего не будет выведено

105.	Что выведет на консоль следующая программа?
	public class QTest {
	{
	System.out.print("1");
	}
	public static void main(String[] args)
	{ System.out.print("2"); new QTest();
	}
	static { System.out.print("3");
	}}
		a. 23
		b. 213
		c. 2
		d. 123
		e. 321
		f. Ошибка компиляции

106.	После выполнения какой строки следующего кода только один объект в памяти будет доступен для сборки мусора garbage collector'ом?
	1. public class Test {
	2. Test ags = null;
	3. public static void main(String argv[]) {
	4. Test a1 = new Test();
	5. Test a2 = new Test();
	6. Test a3 = new Test();
	7. al.ags = new Test();
	8. a2.ags = al.ags;
	9. a3.ags = a2.ags;
	10. al = null;
	11. a2 = null;
	12. a3 = null;
	13. }
	14. }
		a. 7
		b. 8
		c. 10
		d. 11
		e. 12
		f. В этом методе ни один объект не может быть уничтожен сборщиком мусора

107.	Что выведет на консоль следующая программа?
	public class Compare {
	public static void main(String[] args) {
	String si = new String("Hello");
	String s2 = new String("Hello");
	if (si == s2) { System.out.println("True");
	} else { System.out.println("False");
	} }}
		a. True
		b. False
		c. Ошибка компиляции
		d. Ошибка времени выполнения

108.	Что выведет на консоль следующая программа?
	class A {}
	class B extends A {}
	public class Main {
	public static void main(String[] args) {
	A a = new B();
	B b = (B) a;
	System.out.println(a == b);
	}}
		a. false
		b. true
		c. Ошибка компиляции
		d. Ошибка времени выполнения
		e. Ничего из перечисленного выше

109.	Что выведется на консоль после выполнения следующего кода:
	public class Main {
	static int i;
	public static void main(String[] args) {
	System.out.println(i);
	}}
		a. 0
		b. 1
		c. null
		d. Error: Variable i may not have been initialized

110.	Какие из следующих утверждений о методах классов истинные?	a. private-метод не может быть вызван из другого метода этого же класса b. static-метод не может быть вызван из нестатического метода класса **c. нестатический метод класса не может быть вызван из static-метода** d. метод с видимостью по умолчанию не может быть вызван из private-метода e. все утверждения ложные

111.	Выберите варианты корректного объявления и инициализации переменных:	**a. int _a = 077;**
		b. int 12g = 12;
		c. long l = 0xFFFFFFFFFFFFFFFF L;
		d. double d = 17e-5;
		e. byte b = 255;

112.	Выберите варианты корректного объявления и инициализации переменных:	a. const double PI = 3.14;
		b. byte a = 077;
		c. float f = 0.17f;
		d. long double d = 12.5d;
		e. char cc = 0xFAFA;

113.	Что выведет на консоль следующий код?
	public class Example {
	public static void main(String[] args) {
	byte a = 1;
	byte b = a++;
	byte c = b - a;
	System.out.println(a + " " + b + " " + c);
	}}
		a. 2 1 -1
		b. 1 2 -1
		c. 1 1 0
		d. Ошибка компиляции
		e. Ошибка времени выполнения

114.	Если переменные объявлены следующим образом: short a = 5; int i = 100; То какого типа будет следующее выражение: (a + i)*0x43L + 0.1	a. short b. int c. long d. float **e. double** f. Ошибка преобразования типов
115.	Если переменные объявлены следующим образом: short a = 5; byte b = 3; char c = 'A'; То какого типа будет следующее выражение: a + b * c	a. byte b. short **c. int**

		d. long e. char f. Ошибка преобразования типов
116.	:Вопрос 07.1::Какой метод отвечает за закрытие потока (InputStream OutputStream)	**a. close()** b. closeStream() c. stop()
117.	В сетевом программировании справедливо одно из определений	**a. Сервер – процесс, принимающий соединения с тем, чтобы выдать ответ на полученный запрос** b. Сервер - юнит в серверной стойке, подключенный к сети дата-центра c. Сервер – процесс, осуществляющий доступ к клиенту d. Сервер - главный процесс, распределяющий работу между клиентами
118.	В сетевом программировании справедливо одно из определений	**a. Клиент – процесс, инициирующий доступ к серверу** b. Клиент - второстепенный процесс, управляемый сервером c. Клиент - главный процесс, распределяющий работу между серверами d. Клиент - физическое или юридическое лицо, которому предоставляются сетевые услуги
119.	Для чего используются объекты блокировки, например, мьютексы?	**a. Для синхронизации выполнения отдельных участков программы** b. Для синхронного доступа к оборачиваемой переменной c. Для приостановки второстепенного потока на время работы основного d. Всё перечисленное
120.	Какое минимальное количество потоков имеет любой выполняемый процесс?	a. 0 **b. 1** c. 2 d. 16 e. Зависит от конкретного

		процесса - нет минимального количества как такового
121.	Многозадачность и многопоточность - это	a. Одно и то же **b. Многозадачность - выполнение операционной системой нескольких программ одновременно. Многопоточность - режим выполнения нескольких подпрограмм одной программы одновременно** c. Многозадачность - способность программы выполнять несколько задач одновременно. Многопоточность - выполнение операционной системой программ в несколько потоков (например, на нескольких процессорах) d. Многозадачность - выполнение задач в множестве временных отрезков. Многопоточность - использование программой нескольких потоков для выполнения подпрограмм e. Ничего из перечисленного
122.	Отметьте проблемы многопоточного программирования	**a. Синхронизация доступа к разделяемым ресурсам** **b. Своевременная остановка потоков по желанию родительского потока** **c. Высвобождение памяти, занимаемой локальными переменными, при завершении потока** **d. Контроль временных отрезков, отводимых потоку для работы** e. Своевременный старт второстепенных потоков f. Передача в поток более одного параметра

		g. Всё перечисленное
123.	Чем в рамках сетевого программирования не является протокол?	**a. Алгоритмом работы клиентской и серверной частей приложения или сервиса** b. Задокументированным алгоритмом взаимодействия клиента и сервера c. Набором правил, задающих синтаксис и семантику взаимодействия сетевых узлов d. Соглашением, контролирующим передачу данных между узлами в сети e. Нет подходящих вариантов
124.	Чем помогают в работе паттерны программирования?	**a. Облегчают понимание чужих решений типовых задач** **b. Предлагают готовое решение типовых задач** c. Ускоряют работу программы d. Автоматически документируют код в процессе написания программы e. Передают часть функционала на реализацию программистам-фрилансерам
125.	Что в общем смысле означает фреймворк в современном программировании?	**a. Специальная программная основа для решения задач** b. Специализированный язык программирования c. Специализированная часть языка программирование d. Расширенная библиотека с большим набором готовых решений
126.	Что такое ООР (англ.)?	**a. Объектно-ориентированное программирование** b. Функциональное программирование c. Продукты с открытым

		исходным кодом d. Операции обработки и преобразования
127.	Что такое кроссплатформенность с точки зрения технологии разработки программ?	**a. Способность приложения без значительных изменений работать под управлением различных операционных систем** b. Способность приложения без перекомпиляции работать под управлением различных операционных систем c. Способность вести разработку в IDE, запускаемой под различными операционными системами d. Способность приложения автоматически адаптировать интерфейс как на десктопе, так и на мобильных платформах
128.	Что является основой современных технологий прикладного программирования из представленного списка (можно выбрать несколько пунктов)?	**a. Объектно-ориентированное программирование** b. Функциональное программирование c. Программирование по контракту d. Язык C++ e. Фреймворк Qt f. Скриптовые языки (PHP, Perl, etc)

4. Порядок проведения и критерии оценивания промежуточной аттестации

4.1. Порядок проведения промежуточной аттестации

Экзамен проводится в виде тестирования. Студент должен ответить на вопросы закрытого типа, которые предполагают выбор вариантов ответа. Всего 20 тестовых вопросов. Продолжительность теста – 35 минут.

4.2. Критерии оценивания промежуточной аттестации по видам оценочных средств

Тест формируется в системе электронного обучения MOODLE.
Максимальный балл за тест — 100 баллов.

Оценка	Отлично/ Зачтено	Хорошо/ зачтено	Удовлетворительно/зачтено	Неудовлетворительно/ незачтено
Баллы	100-90 баллов	89-75 баллов	74-60 баллов	59-0 баллов
Уровень освоения проверяемых компетенций	высокий	средний	базовый	недостаточный

4.3. Результаты промежуточной аттестации и уровни сформированности компетенций

При подведении итогов учитываются результаты только промежуточной аттестации:

0-59 баллов – неудовлетворительно/незачтено;
60-74 баллов – удовлетворительно/зачтено;
75-89 баллов – хорошо/зачтено;
90-100 баллов – отлично/зачтено;

Особенности проведении процедуры оценивания результатов обучения инвалидов и лиц с ограниченными возможностями здоровья обозначены в рабочей программе дисциплины (модуля).

Уровни сформированности компетенций определяется следующим образом:

1. Высокий уровень сформированности компетенций соответствует оценке отлично:

- предполагает формирование компетенций на высоком уровне;
- знание теоретических разделов изучаемой дисциплины на уровне не ниже оценки отлично;
- студент умеет применять на практике знания, полученные в рамках изучения дисциплины
- формируются навыки использования теоретических и практических разделов дисциплины для решения задач профессиональной деятельности;

2. Средний уровень соответствует оценке хорошо:

- предполагает формирование компетенций на среднем уровне;

- знание теоретических разделов изучаемой дисциплины на уровне не ниже оценки хорошо;
- студент умеет применять знания, полученные в рамках изучения дисциплины, для решения задач профессиональной деятельности;

3. Базовый уровень соответствует оценке удовлетворительно:

- предполагает формирование компетенций на базовом уровне;
- знание теоретических разделов изучаемой дисциплины на уровне не ниже оценки удовлетворительно;

4. Недостаточный уровень соответствует оценке неудовлетворительно.