Министерство науки и высшего образования Российской Федерации

Муромский институт (филиал)

федерального государственного бюджетного образовательного учреждения высшего образования

«Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»

(МИ ВлГУ)

Кафедра *ИС*

«УТВЕРЖДАЮ»
Заместитель директора по УР
_____Д.Е. Андрианов
25.05.2021

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Технологии программирования

Направление подготовки *09.03.02 Информационные системы и технологии*

Профиль подготовки *Информационные системы и технологии*

Семестр	Трудоемкость, час./зач. ед.	Лекции, час.	Практические занятия, час.	Лабораторные работы, час.	Консультация, час.	Контроль, час.	Всего (контактная работа), час.	СРС, час.	Форма промежуточного контроля (экз., зач., зач. с оц.)
1	**288 / 8**	**36**		**32**	**5,6**	**0,35**	**73,95**	**160,4**	**Экз.(53,65)**
Итого	**288 / 8**	**36**		**32**	**5,6**	**0,35**	**73,95**	**160,4**	**53,65**

Муром, 2021 г.

1. Цель освоения дисциплины

Цель дисциплины: Обучение студентов методам проектирования программ, основам алгоритмизации и технологиям программирования на примере языка С++.

Задачи дисциплины:

В результате освоения курса «Технологии программирования» студенты должны иметь представление:

- О принципах создания программ на языках высокого уровня;
- О простых и сложных типах данных;
- Об алгоритмизации и основных способах представления алгоритмов;
- Об основных структурных элементах программы (циклы, условия, процедуры и функции);
- Об основных типах вычислительных процессов;
- Об особенностях создания программ на языке С++.

2. Место дисциплины в структуре ОПОП ВО

Дисциплина «Технологии программирования» обеспечивает понимание основ проектирования и написания программ для ЭВМ на языках программирования высокого уровня на примере языка С++. Курс базируется на знаниях, полученных студентами в процессе изучения основных дисциплин школьного общеобразовательного курса, а также дисциплин «Информатика», «Математика». Углубление и расширение вопросов, изложенных в данном курсе, будет осуществляться во время работы студентов над дисциплинами: «Объектно-ориентированное программирование» «Функциональное программирование», «Архитектура микропроцессоров и язык Ассемблер», «Системное программное обеспечение» и других, а также при написании бакалаврских работ.

3. Планируемые результаты обучения по дисциплине

Планируемые результаты обучения по дисциплине, соотнесенные с планируемыми результатами освоения ОПОП (компетенциями и индикаторами достижения компетенций)

Формируемые компетенции (код, содержание компетенции)	Планируемые результаты обучения по дисциплине, в соответствии с индикатором достижения компетенции		Наименование оценочного средства
	Индикатор достижения компетенции	Результаты обучения по дисциплине	
ОПК-3 Способен решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности;	ОПК-3.2 Применяет знания приемов безопасной работы в сети Интернет при поиске информации, связанной с профессиональной деятельностью	Умеет применять знания приемов безопасной работы в сети Интернет при поиске информации, связанной с профессиональной деятельностью (ОПК-3.2)	вопросы к устному опросу, тесты, задания к лабораторным работам
ОПК-6 Способен разрабатывать алгоритмы и программы, пригодные для практического	ОПК-6.3 Программирует, отлаживает и тестирует прототипы программно-технических комплексов	Владеет навыками программирования, отладки и тестирования прототипов программно-технических комплексов (ОПК-6.3)	к устному опросу, тесты, задания к лабораторным работам

применения в области информационных систем и технологий;	ОПК-6.2 Применяет методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области информационных систем и технологий	Умеет применять методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области информационных систем и технологий (ОПК-6.2)	
	ОПК-6.1 Демонстрирует знания алгоритмизации, языков и технологий программирования, пригодных для практического применения в области информационных систем и технологий	Демонстрирует знания алгоритмизации, языков и технологий программирования, пригодных для практического применения в области информационных систем и технологий (ОПК-6.1)	

4. Структура и содержание дисциплины

Общая трудоемкость дисциплины составляет 8 зачетных единиц, 288 часов.

4.1. Форма обучения: очная

Уровень базового образования: среднее общее.
Срок обучения 4г.

4.1.1. Структура дисциплины

№ п\п	Раздел (тема) дисциплины	Семестр	Контактная работа обучающихся с педагогическим работником							Самостоятельная работа	Форма текущего контроля успеваемости (по неделям семестра), форма промежуточной аттестации(по семестрам)
			Лекции	Практические занятия	Лабораторные работы	Контрольные работы	КП / КР	Консультация	Контроль		
1	Принципы решения задач на ЭВМ	1	2								устный опрос, тестирование
2	Методы проектирования базовых и прикладных ИТ	1	2							7	Устный опрос, тестирование
3	Представление алгоритмов обработки информации	1	2							2	Устный опрос, тестирование
4	Написание программ для ЭВМ	1	2								Устный опрос, тестирование
5	Основы С++	1	4								Устный опрос, тестирование
6	Механизмы ввода-вывода	1	2		8					2	Устный опрос, тестирование, лабораторные работы
7	Ветвления и циклы	1	4								Устный опрос, тестирование,
8	Массивы, строки и указатели	1	4		8						Устный опрос, тестирование, лабораторные работы
9	Функции	1	4		8					25	Устный опрос, тестирование, лабораторные работы
10	Директивы препроцессора	1	2								Устный опрос, тестирование
11	Объектно-ориентированное программирование	1	4		8					93	Устный опрос, тестирование,лабораторные работы

12	Обработка исключительных ситуаций	1	2								Устный опрос, тестирование
13	Отладка программ. Методы тестирования	1	2							31,4	Устный опрос, тестирование
Всего за семестр		288	36		32			5,6	0,35	160,4	Экз.(53,65)
Итого		288	36		32			5,6	0,35	160,4	53,65

4.1.2. Содержание дисциплины

4.1.2.1. Перечень лекций

Семестр 1

Раздел 1. Принципы решения задач на ЭВМ

Лекция 1.

Введение. принципы решения задач на ЭВМ (2 часа).

Раздел 2. Методы проектирования базовых и прикладных ИТ

Лекция 2.

Методы проектирования базовых и прикладных информационных технологий (2 часа).

Раздел 3. Представление алгоритмов обработки информации

Лекция 3.

История развития языков программирования. Синтаксис языка C++. Алфавит языка (2 часа).

Раздел 4. Написание программ для ЭВМ

Лекция 4.

Основы C++. История возникновения. Алфавит и идентификаторы. Основные типы данных. Объявление переменных. Время существования и область видимости переменных (2 часа).

Раздел 5. Основы C++

Лекция 5.

Основы C++. Комментарии. Операции, выражения и операторы. Приведение типов. Ключевое слово typedef. Перечисления (2 часа).

Лекция 6.

Механизмы ввода-вывода (2 часа).

Раздел 6. Механизмы ввода-вывода

Лекция 7.

Ветвления и циклы. Операторы выбора (2 часа).

Раздел 7. Ветвления и циклы

Лекция 8.

Операторы цикла. Оператор безусловного перехода (2 часа).

Лекция 9.

Указатели. Адресная арифметика. Работа с динамической памятью (2 часа).

Раздел 8. Массивы, строки и указатели

Лекция 10.

Одномерные массивы. Многомерные массивы (2 часа).

Лекция 11.

Указатели и массивы. Строки. Динамические массивы. Ссылки (2 часа).

Раздел 9. Функции

Лекция 12.

Объявление, определение и вызов функции. Передача аргументов по значению. Передача аргументов через указатель. Передача аргументов по ссылке (2 часа).

Лекция 13.

Аргументы по умолчанию. Рекурсивные функции (2 часа).

Раздел 10. Директивы препроцессора

Лекция 14.

Встраиваемые функции. Перегрузка функций. Шаблоны функций (2 часа).

Раздел 11. Объектно-ориентированное программирование

Лекция 15.

Директива #include. Директива #define. Директивы условной компиляции (2 часа).

Лекция 16.

Основные принципы ООП. Объявление класса. Использование класса. Объекты (2 часа).

Раздел 12. Обработка исключительных ситуаций

Лекция 17.

Конструкторы. Деструкторы. Структуры. Объединения (2 часа).

Раздел 13. Отладка программ. Методы тестирования

Лекция 18.

Генерация и перехват исключений. Генерация исключений внутри функций. Повторная гене-рация исключения. Стандартные и собственные классы исключений (2 часа).

4.1.2.2. Перечень практических занятий

Не планируется.

4.1.2.3. Перечень лабораторных работ

Семестр 1

Раздел 6. Механизмы ввода-вывода

Лабораторная 1.

Типы данных и структуры программы. Вычисление простого и условного выражений (4 часа).

Лабораторная 2.

Табулирование функции (4 часа).

Раздел 8. Массивы, строки и указатели

Лабораторная 3.

Работа с битами. Одномерные и двумерные массивы (4 часа).

Лабораторная 4.

Структуры (4 часа).

Раздел 9. Функции

Лабораторная 5.

Функции. Перегрузка функций. Шаблоны функций (4 часа).

Лабораторная 6.

Файлы и строки (4 часа).

Раздел 11. Объектно-ориентированное программирование

Лабораторная 7.

Динамические структуры данных (4 часа).

Лабораторная 8.

Разработка объектно-ориентированной программы (4 часа).

4.1.2.4. Перечень тем и учебно-методическое обеспечение самостоятельной работы

Перечень тем, вынесенных на самостоятельное изучение:

1. Запись алгоритмов в виде структурограмм.
2. Компиляторы и интерпретаторы: особенности, преимущества, недостатки.
3. Рекомендации по оформлению программ на C++.
4. Использование библиотеки STL для работы с классами - контейнерами.
5. Возвращение функциями указателей. Указатели на указатели.
6. Обработка аргументов командной строки с использованием параметров функции main().
7. Спецификаторы const и volatile.

8. Копирующий конструктор.
9. Перегрузка операторов.
10. Наследование. Доступ к членам родительского класса. Множественное наследова-ние. Виртуальные функции.
11. Использование потоков ввода-вывода для работы с файлами.
12. Шаблоны классов.
13. Пространства имён.
14. Принципы написания эффективного кода программ для поддержки систем в работоспособном состоянии.
15. Виды тестов и принципы тестирования программных продуктов.

Для самостоятельной работы используются методические указания по освоению дисциплины и издания из списка приведенной ниже основной и дополнительной литературы.

4.1.2.5. Перечень тем контрольных работ, рефератов, ТР, РГР, РПР

Не планируется.

4.1.2.6. Примерный перечень тем курсовых работ (проектов)

Не планируется.

4.2 Форма обучения: заочная

Уровень базового образования: среднее профессиональное.
Срок обучения 3г 6м.

Семестр	Трудоем-кость, час./ зач. ед.	Лек-ции, час.	Практи-ческие занятия, час.	Лабора-торные работы, час.	Консуль-тация, час.	Конт-роль,час.	Всего (контак-тная работа), час.	СРС, час.	Переат-теста-ция	Форма промежу-точного контроля (экз., зач., зач. с оц.)
1	**288 / 8**	**12**		**12**	**6**	**0,6**	**30,6**	**212,75**	**36**	**Экз.(8,65)**
Итого	**288 / 8**	**12**		**12**	**6**	**0,6**	**30,6**	**212,75**	**36**	**8,65**

4.2.1. Структура дисциплины

№ п\п	Раздел (тема) дисциплины	Семестр	Контактная работа обучающихся с педагогическим работником							Самостоятельная работа	Форма текущего контроля успеваемости (по неделям семестра), форма промежуточной аттестации(по семестрам)
			Лекции	Практические занятия	Лабораторные работы	Контрольные работы	КП / КР	Консультация	Контроль		
1	Принципы решения задач на ЭВМ	1	4							17	устный опрос, тестирование
2	Методы проектирования базовых и прикладных ИТ	1	2							40	Устный опрос, тестирование
3	Представление алгоритмов обработки информации	1	4							15	Устный опрос, тестирование
4	Написание программ для ЭВМ	1	2							0	Устный опрос, тестирование
5	Основы С++	1								4	Устный опрос, тестирование

6	Механизмы ввода-вывода	1			4					14	Устный опрос, тестирование, лабораторные работы
7	Ветвления и циклы	1								30	Устный опрос, тестирование,
8	Массивы, строки и указатели	1			4					20	Устный опрос, тестирование, лабораторные работы
9	Функции	1			4					20	Устный опрос, тестирование, лабораторные работы
10	Директивы препроцессора	1								15	Устный опрос, тестирование
11	Объектно-ориентированное программирование	1								14	Устный опрос, тестирование,лабораторные работы
12	Обработка исключительных ситуаций	1								23,75	Устный опрос, тестирование
Всего за семестр		252	12		12	+		6	0,6	212,75	Экз.(8,65)
Итого		252	12		12			6	0,6	212,75	8,65
Итого с переаттестацией		288									

4.2.2. Содержание дисциплины

4.2.2.1. Перечень лекций

Семестр 1

Раздел 1. Принципы решения задач на ЭВМ

Лекция 1.

Язык С++. Приоритеты операций. Арифметические, логические, побитовые операции (2 часа).

Лекция 2.

Типы данных. Операторы if, while, for. Массивы (2 часа).

Раздел 2. Методы проектирования базовых и прикладных ИТ

Лекция 3.

Указатели. Адресная арифметика. Динамическое распределение памяти (2 часа).

Раздел 3. Представление алгоритмов обработки информации

Лекция 4.

Структуры. Функции. Определение и вызов функций (2 часа).

Лекция 5.

Механизмы ввода-вывода. Директивы препроцессора (2 часа).

Раздел 4. Написание программ для ЭВМ

Лекция 6.

Объектно-ориентированное программирование (2 часа).

4.2.2.2. Перечень практических занятий

Не планируется.

4.2.2.3. Перечень лабораторных работ

Семестр 1

Раздел 1. Механизмы ввода-вывода

Лабораторная 1.

Типы данных и структуры программы. Вычисление простого и условного выражений (4 часа).

Раздел 2. Массивы, строки и указатели

Лабораторная 2.

Табулирование функции (4 часа).

Раздел 3. Функции

Лабораторная 3.

Работа с битами. Одномерные и двумерные массивы (4 часа).

4.2.2.4. Перечень тем и учебно-методическое обеспечение самостоятельной работы

Перечень тем, вынесенных на самостоятельное изучение:

1. Запись алгоритмов в виде структурограмм.
2. Компиляторы и интерпретаторы: особенности, преимущества, недостатки.
3. Рекомендации по оформлению программ на C++.
4. Использование библиотеки STL для работы с классами - контейнерами.
5. Возвращение функциями указателей. Указатели на указатели.
6. Обработка аргументов командной строки с использованием параметров функции main().
7. Спецификаторы const и volatile.
8. Копирующий конструктор.
9. Перегрузка операторов.
10. Наследование. Доступ к членам родительского класса. Множественное наследование. Виртуальные функции.
11. Использование потоков ввода-вывода для работы с файлами.
12. Шаблоны классов.
13. Пространства имён.
14. Принципы написания эффективного кода программ для поддержки систем в работоспособном состоянии.
15. Виды тестов и принципы тестирования программных продуктов.

Для самостоятельной работы используются методические указания по освоению дисциплины и издания из списка приведенной ниже основной и дополнительной литературы.

4.2.2.5. Перечень тем контрольных работ, рефератов, ТР, РГР, РПР

1. Методы сортировки.
2. Использование отладчика GDB.
3. Компилятор GCC.
4. Сборка программ с использованием Makefile.
5. Восходящее программирование.
6. Нисходящее программирование.
7. Модульное программирование.
8. Способы доказательства правильности.
9. Отладка программ.
10. Стили оформления программ на C++.
11. Интерфейс Windows API.
12. Язык текстовой разметки XML.
13. Работа с файлами в C++.
14. Работа со строками в C++.
15. Регулярные выражения в C++.
16. Стандартная библиотека шаблонов (STL).

17. Сложные структуры данных: бинарные деревья.
18. Полнотекстовый поиск.
19. Системы контроля версий.
20. Макросы в C++.
21. Работа с динамической памятью в C++.
22. Проблемы переноса программ на 64-битные платформы.
23. Виды списков и способы их реализации в C++.

4.2.2.6. Примерный перечень тем курсовых работ (проектов)

Не планируется.

5. Образовательные технологии

В процессе изучения дисциплины Технологии программирования применяется контактная технология преподавания (за исключением самостоятельно изучаемых студентами вопросов). При проведении лабораторных работ применяется имитационный или симуляционный подход, когда преподавателем разбирается на конкретном примере проблемная ситуация, все шаги решения задачи студентам демонстрируются при помощи мультимедийной техники. Затем студенты самостоятельно решают аналогичные задания.

Во время выполнения лабораторных работ каждому студенту выдается конкретное задание, тем самым формируется способность обучающихся к самостоятельной работе при решений определенных задач, связанных с изучением основ программирования на языке C++.

6. Оценочные средства для текущего контроля успеваемости, промежуточной аттестации по итогам освоения дисциплины.

Фонды оценочных материалов (средств) приведены в приложении.

7. Учебно-методическое и информационное обеспечение дисциплины.

7.1. Основная учебно-методическая литература по дисциплине

1. Павловская Т.А. Программирование на языке высокого уровня C# [Электронный ресурс]/ Павловская Т.А.— Электрон. текстовые данные.— Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016.— 245 с. - http://www.iprbookshop.ru/73713.html

2. Баженова И.Ю. Введение в программирование [Электронный ресурс]: учебное пособие/ Баженова И.Ю., Сухомлин В.А.— Электрон. текстовые данные.— Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020.— 326 с. - http://www.iprbookshop.ru/97539.html

7.2. Дополнительная учебно-методическая литература по дисциплине

1. Канунова Е.Е., Стулов Н. Н Информатика и программирование: учебное пособие. В 2-х ча-стях [Гриф]. Ч.1. / Канунова Е.Е., Стулов Н. Н, Стулов Н. Н - Муром: ИПЦ МИ ВлГУ, 2008. - 74с. - 60 экз.

2. Канунова Е.Е., Стулов Н. Н Информатика и программирование: учебное пособие. В 2-х частях [Гриф]. Ч.2. / Канунова Е.Е., Стулов Н. Н, Стулов Н. Н - Муром: ИПЦ МИ ВлГУ, 2008. - 72с. - 60 экз.

3. Информатика и программирование: метод. указания к лабораторным работам / сост.: Е.Е.Канунова- Муром: Изд.-полиграфический центр МИ ВлГУ, 2010. – 78 с. - 100 экз.

4. Информатика [Электронный ресурс]: учебное пособие для СПО/ — Электрон. текстовые данные.— Саратов: Профобразование, 2021.— 171 с. - http://www.iprbookshop.ru/99928.html

5. Фарафонов А.С. Программирование на языке высокого уровня [Электронный ресурс]: методические указания к проведению лабораторных работ по курсу «Программирование»/ Фарафонов А.С.— Электрон. текстовые данные.— Липецк: Липецкий государственный технический университет, ЭБС АСВ, 2013.— 32 с. - http://www.iprbookshop.ru/22912.html

7.3. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень программного обеспечения и информационных справочных систем

В образовательном процессе используются информационные технологии, реализованные на основе информационно-образовательного портала института (www.mivlgu.ru/iop), и инфокоммуникационной сети института:

- предоставление учебно-методических материалов в электронном виде;
- взаимодействие участников образовательного процесса через локальную сеть института и Интернет;
- предоставление сведений о результатах учебной деятельности в электронном личном кабинете обучающегося.

Информационные справочные системы:

- электронная библиотечная системы "IPRBooks" (http://www.iprbookshop.ru/);

Программное обеспечение:

РЕД ОС (Соглашение №140/05-21У от 18.05.2021 года о сотрудничестве в области науки, развития инновационной деятельности)

QT Creator ((L)GPL)

7.4. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины

iprbookshop.ru

mivlgu.ru/iop

8. Материально-техническое обеспечение дисциплины

Лаборатория разработки информационных систем

12 персональных компьютеров; проектор View Sonic PG603X DLP; экран настенный Lumien

Кабинет основ экологического права

Комплект учебно-наглядных пособий; стендовые материалы; проектор LP640LCL; ноутбук ASUS; экран DRAPPER Apex STAR

9. Методические указания по освоению дисциплины

Для успешного освоения теоретического материала обучающийся: знакомится со списком рекомендуемой основной и дополнительной литературы; уточняет у преподавателя, каким дополнительным пособиям следует отдать предпочтение; ведет конспект лекций и прорабатывает лекционный материал, пользуясь как конспектом, так и учебными пособиями.

До выполнения лабораторных работ обучающийся изучает соответствующий раздел теории. Перед занятием студент знакомится с описанием заданий для выполнения работы, внимательно изучает содержание и порядок проведения лабораторной работы. Лабораторная работа проводятся в компьютерном классе. Обучающиеся выполняют индивидуальную задачу компьютерного моделирования в соответствии с заданием на лабораторную работу. Полученные результаты исследований сводятся в отчет и защищаются по традиционной методике в классе на следующем лабораторном занятии. Необходимый теоретический материал, индивидуальное задание, шаги выполнения лабораторной работы и требование к

отчету приведены в методических указаниях, размещенных на информационно-образовательном портале института.

Самостоятельная работа оказывает важное влияние на формирование личности будущего специалиста, она планируется обучающимся самостоятельно. Каждый обучающийся самостоятельно определяет режим своей работы и меру труда, затрачиваемого на овладение учебным содержанием дисциплины. Он выполняет внеаудиторную работу и изучение разделов, выносимых на самостоятельную работу, по личному индивидуальному плану, в зависимости от его подготовки, времени и других условий.

Форма заключительного контроля при промежуточной аттестации – экзамен. Для проведения промежуточной аттестации по дисциплине разработаны фонд оценочных средств и балльно-рейтинговая система оценки учебной деятельности студентов. Оценка по дисциплине выставляется в информационной системе и носит интегрированный характер, учитывающий результаты оценивания участия студентов в аудиторных занятиях, качества и своевременности выполнения заданий в ходе изучения дисциплины и промежуточной аттестации.

Программа составлена в соответствии с требованиями ФГОС ВО по направлению *09.03.02 Информационные системы и технологии* и профилю подготовки *Информационные системы и технологии*
Рабочую программу составил *к.т.н., доцент, Канунова Е.Е.*_____

Программа рассмотрена и одобрена на заседании кафедры *ИС*

протокол № 17 от 27.04.2021 года.
Заведующий кафедрой *ИС* _____*Андрианов Д.Е.*
(Подпись)

Рабочая программа рассмотрена и одобрена на заседании учебно-методической комиссии факультета

протокол № 9 от 24.05.2021 года.
Председатель комиссии ФИТР _____ *Рыжкова М.Н.*
(Подпись) (Ф.И.О.)

Приложение

Фонд оценочных материалов (средств) по дисциплине
Технологии программирования

1. Оценочные материалы для проведения текущего контроля успеваемости по дисциплине

1. Темы для устного опроса:
Рейтинг-контроль №1
-Основные этапы решения задач на ЭВМ
-Уровни и методы проектирования программ
-Структурное программирование
-Основные свойства алгоритма
-Способы записи алгоритма
-Вербальное представление алгоритма.
-Построчная запись алгоритма
-Представление алгоритма в виде блок-схемы
-Представление алгоритма на языке программирования
-Языки программирования
-Системы программированияс
-История возникновения языка C++
-Алфавит и идентификаторы
-Правила образования идентификаторов
-Ключевые слова
-Символы операций и разделители
-Литералы
-Основные типы данных
-Логический тип (bool)
-Символьные типы (char, wchar_t)
-Целые типы (short, int, long)
-Вещественные типы (float, double, long double)
-Тип void
-Объявление переменных
-Время существования и область видимости переменных
--Комментарии
-Операции, выражения и операторы
-Арифметические операции
-Инкремент и декремент
-Операции отношения
-Логические операции
-Операции присваивания
-Побитовые операции
-Сдвиги
-Приведение типов
-Ключевое слово typedef
-Перечисления

Рейтинг-контроль №2

-Встроенные потоки ввода–вывода
-Форматированный ввод-вывод
-Форматирование с использованием флагов
-Использование манипуляторов ввода-вывода
-Установка ширины поля, точности и символов заполнения
-Функции ввода-вывода printf и scanf

-Ветвления и циклы
-Операторы выбора
-Условный оператор (if)
-Тернарный оператор ?:
-Оператор switch
-Операторы цикла
-Цикл for
-Цикл while
-Цикл do…while
-Операторы управления циклами – break и continue
-Оператор безусловного перехода goto
-Одномерные массивы
-Многомерные массивы
-Указатели
-Адресная арифметика
-Указатели и массивы
-Строки
-Ввод строк с клавиатуры
-Функции работы со строками
-Работа с динамической памятью
-Динамические массивы
-Ссылки
-Объявление, определение и вызов функции
-Передача аргументов
-Передача аргументов по значению
-Передача аргументов через указатель
-Передача аргументов по ссылке
-Аргументы по умолчанию
-Рекурсивные функции
-Встраиваемые функции
-Перегрузка функций
-Шаблоны функций

Рейтинг-контроль №3

-Директива #include
-Директива #define
-Директивы условной компиляции
-Основные принципы ООП
-Объявление класса
-Использование класса. Объекты
-Конструкторы
-Деструкторы
-Структуры
-Объединения
-Генерация и перехват исключений
-Генерация исключений внутри функций
-Повторная генерация исключения

Общее распределение баллов текущего контроля по видам учебных работ для студентов

Рейтинг-контроль 1	Устный опрос (2 вопроса)	До 10 баллов
Рейтинг-контроль 2	Устный опрос (2 вопроса)	До 10 баллов
Рейтинг-контроль 3	Устный опрос (2 вопроса)	До 10 баллов
Посещение занятий студентом	Отметка в журнале посещений	До 10 баллов
Дополнительные баллы (бонусы)		0
Выполнение семестрового плана самостоятельной работы	Защита лабораторных работ	До 20 баллов

2. Промежуточная аттестация по дисциплине

Перечень вопросов к экзамену / зачету / зачету с оценкой.
Перечень практических задач / заданий к экзамену / зачету / зачету с оценкой (при наличии)

ОПК-6, ОПК-3
Блок (знать).

1. Расположите типы данных в порядке невозрастания их размеров
- char
- float
- double
- long double

2. Расположите типы данных в порядке неубывания их размеров
- char
- float
- double
- long double

3. Укажите назначение функций обработки строк
- strcpy
- strcmp

strlen
- strchr

4. Укажите размеры типов данных в компиляторе Visual Studio 32bit
- char
- short
- int
- double

5. Укажите тип приведённых литералов
- 0xFF
- -2e3
- false
- 'a'
- "true"

6. Укажите тип приведённых литералов
- 0x85
- 3.0
- true
- '1'
- "1"

7. В каких вариантах допущены ошибки
- char str[20]; str = "Hello";
- char str[20] = "Hello";
- char str[50]; strcpy("Hello", str);
- char str[5]; strcpy(str, "Hello, world");
- char str[2]; strcpy(str, "3");

8. В каких случаях будет истинным условие if (!strcmp(str1, str2))
- Строки str1 и str2 равны
- Строки str1 и str2 не равны
- Строка str1 "больше" строки str2
- Строка str1 "меньше" строки str2
- Такая запись недопустима

9. В каких случаях доступ к элементам массива осуществляется неправильно?
- int a[10]; a[10];
- short a[5]; a[24%5];
- int a[3][4]; a[2];
- int a[5]; a[2][2];

10. В каких случаях инициализация массива осуществляется без синтаксических ошибок?
- int a[2] = {1, 2, 3};
- float a[3] = {2.0, 3, 4};
- short b[4] = {2};
- char a[4] = "12";
- double d[3]; d = {1.2, 2.4, 3.4};

11. В каких случаях указано объявление функции, без определения?
- void print();
- int abs(int a) {return a > 0 ? a : -a; };
- void empty() {};
- float max(float v1, float v2) { return v1 > v2 ? v1 : v2; }
- size_t DuplicatesCount(int* arr, size_t size);

12. В каких случаях функция вызывается некорректно (тип параметров и самой функции значения не имеет)?
- myFunc(a, b);
- myFunc(int a, float b);
- int myFunc(a, b);
- myFunc((int) a, (float) b);
- int myFunc(int a, float b);

13. Где прекращает существование переменная i в приведённом фрагменте кода?

```
void func() {
if (a > b) {
short i;
// A
for (i = 0; i < 10; i++) {
cout << i;
// B
}
// C
}
// D
}
```

- A
- B
- C
- D

14. Значение какого типа будет возвращать функция func в следующем коде?

```
template <class MyType> MyType func(int a, MyType b) {
   double res = pow (b, 1.0/a);
   return res;
}
void main() {
   float f;
   short a = 20;
   f = func(5, a); // значение какого типа вернёт func?
}
```

- int
- float
- double
- short
- неопределённого

15. Как выглядит многострочный комментарий в C++?

- //
- /* */
- (* *)
- { }
- #

16. Какие аргументы функции func передаются по ссылке? void func (int a, float* b, int& c, float** d, smth &e);

- a
- b
- c
- d
- e

17. Какие управляющие структуры являются необходимыми с точки зрения теории структурного программирования

- Последовательное выполнение
- Ветвление
- Безусловный переход
- Цикл
- Завершение

18. Какие этапы решения задач на ЭВМ существуют

- Постановка задачи
- Поиск задачи
- Составление программы
- Тестирование и отладка
- Утилизация программы
- Формализация задачи

19. Методы проектирования, в которых вместо реальных модулей используются заглушки

- Нисходящее проектирование
- Восходящее проектирование
- Оба метода

20. Правила написания идентификаторов в C++

- Первым символом может быть буква или подчёркивание
- Первым символом может быть буква или цифра

- Последующими символами могут быть буквы, цифры, символы подчёркивания
- Последующими символами могут быть буквы, цифры
- Идентификатор чувствителен к регистру
- Идентификатор нечувствителен к регистру

ОПК-3:
Блок (уметь).

21. Какие из перечисленных инструкций являются циклами с предусловием?
- for
- while
- do ... while
- switch
- throw
- continue

22. Какие из перечисленных циклов будут бесконечными
- for (int i = 100; i > 0; i--)
- for (int i = -100; i < 0; i--)
- for (int i = 1; i < 3; i += 0.1)
- for (int i = 0; i <= 0; i++)
- for (float i = 1; i != 3; i += 0.6)

23. Какие из перечисленных циклов будут бесконечными?
- for (;;) ;
- for (int i =0; i > 0; ++i) ;
- while (true) ;
- while (NULL) ;
- for (int i =0; i <10; i++) i--;

24. Какие из перечисленных циклов не выполнятся ни разу?
- for (int i = -10; i <= -20; i++)
- for (float x = 0; x >= -10; x -= 0.1)
- int i = -1; do { i++; } while (i < 0)
- int x = 1; while (x--) cout << x;
- int x = 1; while (--x) cout << x;

25. Какие из циклов завершатся при значении i < 10?
- for (int i = 1; i < 100; i++) if (i % 5) break;
- for (int i = 9; i < 100; i += 2) if (i < 10) return;
- for (int i = 3; i < 100; i ++) if (i < 10) continue;
- int i = 100; while (i - 9) i--;

26. Какие массивы объявлены с ошибками?
- int a[10];
- int n; int a[n];
- float a[];
- float a[] = {1,2};
- char a[5, 2];
- char a[20-5];

27. Какие переменные имеют тип указателя на int?

int* a, b;
int** c;
int *d, e;

- a
- b
- c
- d
- e

28. Какие условия будут истинными? Если int x = 5.5;

- if (x == 5)
- if (-5)
- if ("false")
- if (4-4)
- if (NULL)
- if (x = 2)

29. Какими способами можно увеличить значение, хранящееся в переменной A, на 4?

- A = 4;
- A += 4;
- A += A + 4;
- A = A + 4;
- A + 4;
- A ++ 4;

30. Каким ключевым словом обозначаются рекурсивные функции?

- inline
- template
- throw
- recursive
- никаким

31. Каким способом вывести значения переменных int a и char c[10] функцией printf?

- printf(a, c);
- printf("d, s", a, c);
- printf(%a, %c);
- printf("%d, %s", a, c);

Каким типом данных будет заменён в данном случае параметр шаблона ArrType?

```
float arr[10];
template <class ArrType> long func (ArrType* arr, int n);
...
void main() {
   long v = func(arr, 5);
}
```

- int
- float
- long
- void

32. Как объявить двумерный динамический массив?

- int a[][];
- int* a;
- int* a[];
- int** a;

33. Как получить доступ к названию первого товара в магазине (поле Product, строка) в следующем примере

```
struct Shop {
   char Name[100];
   char Product[100][50];
};
Shop sh;
```

- sh.Product;
- sh.Product[0];
- sh.Product[1];
- sh.Product[0][0];
- sh.Product[0][1];

34. Как получить доступ к полю fld?

```
struct Str {
```

```
    int fld;
};
Str* s;
```

- s.fld;
- *s.fld;
- s.*fld
- *(s.fld);
- s->fld;

35. Как получить доступ к фамилии первого студента в группе (Поле Surname) в следующем примере

```
struct Student {
    char Surname[50];
    char FirstName[30];
};
struct Group {
    Student Students[25];
    char Name[20];
};
Group is00;
```

- is00.Students[0];
- is00.Students[0].Surname;
- is00.Students.Surname[0];
- is00.Surname[0];

36. Как правильно выделить память под одномерный динамический массив?

- int* arr; arr = new int [20];
- int* arr; arr = new int(20);
- int** arr; arr = new int* [20];
- int* arr; arr(20) = new;

37. Какую арифметическую операцию нельзя применять к переменным вещественного типа?

- +
- -
- *
- /
- %

38. Переменные какого типа нельзя объявлять?

- unsigned char
- wchar_t
- long double
- void

39. Под какие массивы будет выделено менее 10 байт памяти, предполагая, что используется компилятор Visual Studio 32bit?

- int a[3];
- char str[] = "123456789";
- char a[9];
- short sh[4 + 0.9];

40. Пусть задан трёхмерный массив float b[6][5][4]. Что будет собой представлять элемент a[2][3]?

- Такая запись элемента недопустима
- Число типа float
- Двумерный массив float[5][4]
- Одномерный массив float[4]

41. Пусть задан трёхмерный массив int a[4][5][6]. Что будет собой представлять элемент a[2]?

- Такая запись элемента недопустима
- Число типа int
- Двумерный массив int[5][6]
- Одномерный массив int[5]

42. Пусть объявлены следующие перегруженные функции

```
int func(int, char);
float func(double, char);
double func(float);
```

Какая из них будет вызвана в следующем коде

```
double a, b = 5;
a = func((float) b, '!');
```

- int func(int, char);
- float func(double, char);
- double func(float);
- нет подходящей функции
- неоднозначность в выборе функции

43. Размер каких массивов будет менее 5 элементов?

- int a[5];
- short a[4];
- char a[] = "1234";
- float a[3][2];
- int a[4][1]

44. Сколько раз выполнится тело данного цикла?

```
for (int i = -5; i <= -10; i++)
   cout << i << ' ';
```

- 0
- бесконечное количество раз
- 5
- 6
- 10

45. Тело каких циклов выполнится 10 раз?

- for (int i = 0; i <= 9; i++)
- for (int i = 1; i <10; i++)
- int x = 0; while (x <= 30) x+=3;
- for (int i = 0; i < 1; i += 0.1)

46. Укажите варианты, в которых допущены ошибки

- int a = 5; int* p = 0; *p = a;
- int a = 5; int* p = a; *p = 4;
- int a = 5; int* p = &a; *p = 4;
- int a = 5; int* p; p = *a;

47. Укажите двоичное значение выражения ~A ^ B при

A=10010101

B=10101001

- 11000011
- 00110101
- 10010100
- 00101001
- 11100101

48. Укажите двоичное значение выражения ~A ^ B при

A=10010101

B=10101001

- 00101000
- 00101001
- 10100000

- 01010010
- 00000101

49. Укажите инструкции for, записанные с синтаксическими ошибками
- for (int x = 0; x < 5; x+=5)
- for (int x > 0; x<5; x++)
- for (int x = 1; x< 5)
- for (int x = -1; x < 5;)
- for (int x = 0; x != 5; x--)

50. Укажите инструкции if, записанные с синтаксическими ошибками
- if x == 10
- if (x == 10)
- if (x == 10 && y != 100)
- if (x ==10) && (y != 100)
- if ((x==10) && (y != 100))

51. Укажите неверные объявления переменных
- unsigned float a = 100;
- char ch = 50;
- int a = 30, float b = 30.3;
- double = 50;
- float b, c = 0.4;

52. Укажите потоки, по умолчанию выводящие информацию на монитор
- cin
- clog
- cerr
- cout
- cmon

53. Укажите правильные объявления строк
- const char str[50];
- char str[50];
- char str[];
- char str[] = "Hello";
- const char* str = "world";
- char str[5] = "Hello";
- const char str[5] = {'H'};

54. Укажите результат вычисления выражения

a + pow(b, 3) - abs(b) * b

при a = 2, b = -1
- 2
- 3
- 0
- 5
- 4

55. Чему будет равно значение выражения x++ + x++ при x=2
- 4
- 6
- 7
- 5

56. Чему будет равно значение переменной px после выполнения следующего кода

```
int* px = (int*)1000;
px += 4;
cout << px;
```
- 1004
- 1000
- 1008

- 1016

57. Чему будет равно значение переменной x после выполнения кода?

```
void f1(int a) {
   a += a;
};
void f1(int&amp;amp; a) {
   a += a;
};
void main() {
   int x = 3;
   f1(x);
   f2(x);
   cout << x; // Чему равен x?
}
```

- 3
- 6
- 9
- 12

58. Чему будет равно значение переменной x после выполнения кода при x = 0, y = 1

```
x = x ? x++ + y : ++x + y;
```

- 2
- 1
- 3
- 0

59. Чему будет равно значение переменной x после выполнения кода при x = 1, y = 2

```
switch (x)
{
   case 0:
      x = pow(y, x + y);
   case 1:
      x = y + x * 2;
   case 2:
      x = x + y - 3;
}
```

- 3
- 8
- 4
- 0

60. Чему будет равно значение переменной x после выполнения кода при a = 2, b = 1

```
if (a <= b + 1)
   x = a + b;
else
   x = a - b;
   x = x + 1;
```

- 4
- 3
- 2
- 1

61. Чему будет равно значение переменной x после выполнения кода при a = 2

```
if (a = 6 - a)
   x = a;
else
   x = -a;
```

- 4
- -2
- 2
- 0

62. Чему будет равно значение переменной x после выполнения следующего кода?

```
void func(int x) {
   x += x * x;
}
void main() {
   int x = 2;
   func(x);
   cout << x; // Чему равен x?
}
```

- 2
- 4
- 6

63. Чему будет равно значение переменной x после выполнения цикла

```
int x = 1;
for (int i = 0; i <= 5; i++)
   x++;
   x+=2;
```

- 9
- 19
- 16
- 8

64. Чему будет равно значение переменной x после выполнения цикла

```
int x = 1;
for (int i = 0; i <= 5; i++)
   x += i;
```

- 5
- 6
- 15
- 16

65. Чему равно значение выражения

```
1 / 3 * x + y - pow(3, y)
```

при x = 3, y = 1

- -1
- -2
- 0
- 1
- 2

66. Чему равно значение неинициализированной переменной?

- 0
- 1
- NULL
- ничему не равно
- произвольному значению

67. Что называется перегрузкой функции?

- Передача в неё значений, выходящих за допустимые пределы
- Возврат из неё значения, выходящего за допустимые пределы
- Создание универсальной функции, типы аргументов которой определяются при её вызове
- Создание нескольких функций с одним именем и различными параметрами

- Создание нескольких функций с одинаковыми прототипами и различными телами.

68. Что сделает в данном коде оператор break?

```
for (int i = 0; i < 10; i++)
   for (int j = 0; j < 10; j++)
      if (i > 3 &amp;amp;&amp;amp; j < i)
         break;
```

- Прервёт цикл по i
- Прервёт цикл по j
- Прервёт оба цикла
- Перейдёт к следующей итерации цикла i
- Перейдёт к следующей итерации цикла j

69. Что сделает в данном коде оператор continue?

```
for (int i = 0; i < 10; i++)
   for (int j = 0; j < 10; j++)
      if (i > 3 &amp;amp;&amp;amp; j < i)
         continue;
```

- Прервёт цикл по i
- Прервёт цикл по j
- Прервёт оба цикла
- Перейдёт к следующей итерации цикла i
- Перейдёт к следующей итерации цикла j

ПК-30:

Блок (владеть).

70. Основные компоненты систем программирования

- Текстовый редактор
- Файловый менеджер
- Отладчик
- Маркировщик
- Компоновщик
- Транслятор
- Переводчик

71. Что делает директива препроцессора #include?

- Подключает к программе указанную библиотеку
- Вставляет в программу содержимое указанного файла
- Включает в программе указанный тип функций
- Задаёт указанный макрос

72. Какие пять подходов применяются к организации процесса создания и использования программного средства?

- Водопадный
- Исследовательское программирование
- Прототипирование
- Формальные преобразования
- Сборочное программирование
- Семантическое программирование
- Теоретическое программирование

74. Какой подход включает разработку формальных спецификаций программного средства и превращение их в программы путем корректных преобразований?

- Водопадный
- Исследовательское программирование
- Прототипирование

- Формальные преобразования
- Сборочное программирование
- Семантическое программирование
- Теоретическое программирование

75. Какой подход предполагает, что программное средства конструируется, главным образом, из компонент, которые уже существуют ?

- Водопадный
- Исследовательское программирование
- Прототипирование
- Формальные преобразования
- Сборочное программирование
- Семантическое программирование
- Теоретическое программирование

76. Какие стадии жизненного цикла программного средства различают при водопадном подходе его разработки ?

- Разработка программного средства
- Производство программных изделий
- Эксплуатация программного средства
- Анализ технического задания
- Тестирование программного средства

77. При какой модели разработки программного средства используют разбиение всей разработки на стадии, причем переход с одного этапа на другой происходит только в том случае, когда полностью завершена работа на текущем ?

- Каскадная модель
- Спиральная модель
- Кубическая модель
- Объектная модель
- Эксплуатационная модель

78. Что не относится к критериям качества программного средства ?

- функциональность
- надежность
- легкость применения
- эффективность
- сопровождаемость
- мобильнось
- универсальность

79. Свойство, характеризующее степень обладания программным средством всеми необходимыми частями и чертами, требующихся для выполнения своих явных и неявных функций ?

- Завершенность
- Точность
- Автономность
- Устойчивость
- Защищенность
- П-документированность
- Информативность

80. Свойство, характеризующее наличие в программном средстве информации, необходимой и достаточной для понимания назначения программного средства?

- Завершенность
- Точность
- Автономность
- Устойчивость
- Защищенность
- П-документированность

- Информативность

81. Мера, характеризующая способность программы выполнять возложенные на нее функции в течение определенного отрезка времени?

- Эффективность по ресурсам
- Временная эффективность
- Эффективность по устройствам
- Понятность
- Структурированность
- Эдобочитаемость
- Раширяемость

82. Мера, характеризующая способность программы с точки зрения простоты внесения изменений и доработок на всех этапах и стадиях жизненного цикла программы?

- Эффективность по ресурсам
- Временная эффективность
- Эффективность по устройствам
- Понятность
- Структурированность
- Модифицируемость
- Раширяемость

83. Свойство, характеризующее способность программного средства работать на разнообразном аппаратном обеспечении?

- Независимость от устройств
- Временная эффективность
- Эффективность по устройствам
- Понятность
- Структурированность
- Модифицируемость
- Раширяемость

84. Какие методы используются при проектировании программных средств?

- Метод восходящей разработки
- Метод нисходящей разработки
- Конструктивный подход
- Архитектурный подход
- Классический подход
- Объектно-ориентированный метод

85. Процесс многократного повторения программы с целью обнаружения ошибок - это?

- Тестирование
- Отладка
- Проектирование
- Программирование
- Алгоритмизация
- Структурирование

86. Какие методы тестирования существуют?

- Статическое тестирование
- Детерминированное тестирование
- Стохастическое тестирование
- Автономное тестирование
- Частичное тестирование
- Выборочное тестирование

87. Какие виды отладки программ используют в нашей стране?

- Автономная отладка

- Комплексная отладка
- Стохастическая отладка
- Частичная отладка
- Выборочная отладка
- Структурированная отладка

88. Какая документация предназначена для администраторов программного средства?

- Руководство по инсталляции программы
- Общее функциональное описание программного средства
- Справочник по применению программы
- Инструкция по применению программы
- Руководство по управлению программным средством
- Справочник по сопровождению программы

89. Какая документация предназначена для ординарных пользователей программного средства?

- Руководство по инсталляции программы
- Общее функциональное описание программного средства
- Справочник по применению программы
- Инструкция по применению программы
- Руководство по управлению программным средством
- Справочник по сопровождению программы

90. Какая документация содержит краткую характеристику функциональных возможностей программы?

- Руководство по инсталляции программы
- Общее функциональное описание программного средства
- Справочник по применению программы
- Инструкция по применению программы
- Руководство по управлению программным средством
- Справочник по сопровождению программы

91. Укажите этапы жизненного цикла программного средства?

- Анализ
- Проектирование
- Реализация
- Отладка и тестирование
- Сопровождение
- Документирование

92. На каком этапе жизненного цикла выявляют ошибки программы и проверяют ее работоспособность?

- Анализ
- Проектирование
- Реализация
- Отладка и тестирование
- Сопровождение
- Документирование

93. На каком этапе жизненного цикла осуществляют расширение функциональных возможностей программного средства?

- Анализ
- Проектирование
- Реализация
- Отладка и тестирование
- Сопровождение

- Документирование

94. На каком этапе жизненного цикла непосредственно реализуют кодирование текстов программ?
- Анализ
- Проектирование
- Реализация
- Отладка и тестирование
- Сопровождение
- Документирование

95. Среда, служащая для организации диалога программного средства с пользователем на основе графического многооконного представления данных?
- Графический пользовательский интерфейс
- Консольный пользовательский интерфейс
- Мультимедийный пользовательский интерфейс
- Текстовой пользовательский интерфейс
- Модульный пользовательский интерфейс
- Векторный пользовательский интерфейс

96. Какой раздел стандарта ЕСПД содержит описание программы?
- ГОСТ 19.402.-78
- ГОСТ 19.101.-77
- ГОСТ 19.101.-77
- ГОСТ 19.401.-78
- ГОСТ 19.404.-79
- ГОСТ 19.502.-78
- ГОСТ 19.701.-90

97. Какой раздел стандарта ЕСПД содержит руководство системному программисту?
- ГОСТ 19.503.-79
- ГОСТ 19.101.-77
- ГОСТ 19.101.-77
- ГОСТ 19.401.-78
- ГОСТ 19.404.-79
- ГОСТ 19.502.-78
- ГОСТ 19.701.-90

98. Какой раздел стандарта ЕСПД содержит руководство оператору?
- ГОСТ 19.505.-79
- ГОСТ 19.101.-77
- ГОСТ 19.101.-77
- ГОСТ 19.401.-78
- ГОСТ 19.404.-79
- ГОСТ 19.502.-78
- ГОСТ 19.701.-90

99. Какой раздел стандарта ЕСПД содержит руководство программиста?
- ГОСТ 19.504.-79
- ГОСТ 19.101.-77
- ГОСТ 19.101.-77
- ГОСТ 19.401.-78
- ГОСТ 19.404.-79

- ГОСТ 19.502.-78
- ГОСТ 19.701.-90

100. Какой раздел стандарта ЕСПД содержит пояснительную записку?

- ГОСТ 19.404.-79
- ГОСТ 19.505.-79
- ГОСТ 19.101.-77
- ГОСТ 19.101.-77
- ГОСТ 19.401.-78
- ГОСТ 19.502.-78
- ГОСТ 19.701.-90

Методические материалы, характеризующие процедуры оценивания

На основе типовых заданий из предыдущего раздела программным комплексом информационно-образовательного портала МИ ВлГУ формируются в автоматическом режиме тестовые задания для студентов: 15 вопросов в тесте (8 вопросов Блока 1, 4 вопроса Блока 2 и 3 вопроса Блока 3). Программный комплекс формирует индивидуальные задания для каждого зарегистрированного в системе студента и устанавливает время прохождения тестирования. Результатом тестирования является процент правильных ответов, с учетом индивидуального семестрового рейтинга студента формируется экзаменационная оценка.

Максимальная сумма баллов, набираемая студентом по дисциплине равна 100.

Оценка в баллах	Оценка по шкале	Обоснование	***Уровень сформированности компетенций***
Более 80	«Отлично»	Содержание курса освоено полностью, без пробелов, необходимые практические навыки работы с освоенным материалом сформированы, все предусмотренные программой обучения учебные задания выполнены, качество их выполнения оценено числом баллов, близким к максимальному	***Высокий уровень***
66-80	«Хорошо»	Содержание курса освоено полностью, без пробелов, некоторые практические навыки работы с освоенным материалом сформированы недостаточно, все предусмотренные программой обучения учебные задания выполнены, качество выполнения ни одного из них не оценено минимальным числом баллов, некоторые виды заданий выполнены с ошибками	***Продвинутый уровень***

50-65	«Удовлетворительно»	Содержание курса освоено частично, но пробелы не носят существенного характера, необходимые практические навыки работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий, возможно, содержат ошибки	***Пороговый уровень***
Менее 50	«Неудовлетворительно»	Содержание курса не освоено, необходимые практические навыки работы не сформированы, выполненные учебные задания содержат грубые ошибки	***Компетенции не сформированы***

3. Задания в тестовой форме по дисциплине

Примеры заданий:
ОПК-3, ОПК-6:
Блок (знать).

1. Расположите типы данных в порядке невозрастания их размеров
- char
- float
- double
- long double

2. Расположите типы данных в порядке неубывания их размеров
- char
- float
- double
- long double

3. Укажите назначение функций обработки строк
- strcpy
- strcmp

strlen
- strchr

4. Укажите размеры типов данных в компиляторе Visual Studio 32bit
- char
- short
- int
- double

5. Укажите тип приведённых литералов
- 0xFF
- -2e3
- false
- 'a'
- "true"

6. Укажите тип приведённых литералов
- 0x85
- 3.0
- true

- '1'
- "1"

7. В каких вариантах допущены ошибки
- char str[20]; str = "Hello";
- char str[20] = "Hello";
- char str[50]; strcpy("Hello", str);
- char str[5]; strcpy(str, "Hello, world");
- char str[2]; strcpy(str, "3");

8. В каких случаях будет истинным условие if (!strcmp(str1, str2))
- Строки str1 и str2 равны
- Строки str1 и str2 не равны
- Строка str1 "больше" строки str2
- Строка str1 "меньше" строки str2
- Такая запись недопустима

9. В каких случаях доступ к элементам массива осуществляется неправильно?
- int a[10]; a[10];
- short a[5]; a[24%5];
- int a[3][4]; a[2];
- int a[5]; a[2][2];

10. В каких случаях инициализация массива осуществляется без синтаксических ошибок?
- int a[2] = {1, 2, 3};
- float a[3] = {2.0, 3, 4};
- short b[4] = {2};
- char a[4] = "12";
- double d[3]; d = {1.2, 2.4, 3.4};

11. В каких случаях указано объявление функции, без определения?
- void print();
- int abs(int a) {return a > 0 ? a : -a; };
- void empty() {};
- float max(float v1, float v2) { return v1 > v2 ? v1 : v2; }
- size_t DuplicatesCount(int* arr, size_t size);

12. В каких случаях функция вызывается некорректно (тип параметров и самой функции значения не имеет)?
- myFunc(a, b);
- myFunc(int a, float b);
- int myFunc(a, b);
- myFunc((int) a, (float) b);
- int myFunc(int a, float b);

13. Где прекращает существование переменная i в приведённом фрагменте кода?

```
void func() {
if (a > b) {
short i;
// A
for (i = 0; i < 10; i++) {
cout << i;
// B
}
// C
}
// D
}
```

- A
- B

- C
- D

14. Значение какого типа будет возвращать функция func в следующем коде?

```
template <class MyType> MyType func(int a, MyType b) {
   double res = pow (b, 1.0/a);
   return res;
}
void main() {
   float f;
   short a = 20;
   f = func(5, a); // значение какого типа вернёт func?
}
```

- int
- float
- double
- short
- неопределённого

15. Как выглядит многострочный комментарий в C++?

- //
- /* */
- (* *)
- { }
- #

16. Какие аргументы функции func передаются по ссылке? void func (int a, float* b, int& c, float** d, smth &e);

- a
- b
- c
- d
- e

17. Какие управляющие структуры являются необходимыми с точки зрения теории структурного программирования

- Последовательное выполнение
- Ветвление
- Безусловный переход
- Цикл
- Завершение

18. Какие этапы решения задач на ЭВМ существуют

- Постановка задачи
- Поиск задачи
- Составление программы
- Тестирование и отладка
- Утилизация программы
- Формализация задачи

19. Методы проектирования, в которых вместо реальных модулей используются заглушки

- Нисходящее проектирование
- Восходящее проектирование
- Оба метода

20. Правила написания идентификаторов в C++

- Первым символом может быть буква или подчёркивание
- Первым символом может быть буква или цифра
- Последующими символами могут быть буквы, цифры, символы подчёркивания
- Последующими символами могут быть буквы, цифры
- Идентификатор чувствителен к регистру
- Идентификатор нечувствителен к регистру

ОПК-3:
Блок (уметь).

21. Какие из перечисленных инструкций являются циклами с предусловием?
- for
- while
- do ... while
- switch
- throw
- continue

22. Какие из перечисленных циклов будут бесконечными
- for (int i = 100; i > 0; i--)
- for (int i = -100; i < 0; i--)
- for (int i = 1; i < 3; i += 0.1)
- for (int i = 0; i <= 0; i++)
- for (float i = 1; i != 3; i += 0.6)

23. Какие из перечисленных циклов будут бесконечными?
- for (;;) ;
- for (int i =0; i > 0; ++i) ;
- while (true) ;
- while (NULL) ;
- for (int i =0; i <10; i++) i--;

24. Какие из перечисленных циклов не выполнятся ни разу?
- for (int i = -10; i <= -20; i++)
- for (float x = 0; x >= -10; x -= 0.1)
- int i = -1; do { i++; } while (i < 0)
- int x = 1; while (x--) cout << x;
- int x = 1; while (--x) cout << x;

25. Какие из циклов завершатся при значении i < 10?
- for (int i = 1; i < 100; i++) if (i % 5) break;
- for (int i = 9; i < 100; i += 2) if (i < 10) return;
- for (int i = 3; i < 100; i ++) if (i < 10) continue;
- int i = 100; while (i - 9) i--;

26. Какие массивы объявлены с ошибками?
- int a[10];
- int n; int a[n];
- float a[];
- float a[] = {1,2};
- char a[5, 2];
- char a[20-5];

27. Какие переменные имеют тип указателя на int?

int* a, b;
int** c;
int *d, e;

- a
- b
- c
- d

- e

28. Какие условия будут истинными? Если int x = 5.5;
- if (x == 5)
- if (-5)
- if ("false")
- if (4-4)
- if (NULL)
- if (x = 2)

29. Какими способами можно увеличить значение, хранящееся в переменной A, на 4?
- A = 4;
- A += 4;
- A += A + 4;
- A = A + 4;
- A + 4;
- A ++ 4;

30. Каким ключевым словом обозначаются рекурсивные функции?
- inline
- template
- throw
- recursive
- никаким

31. Каким способом вывести значения переменных int a и char c[10] функцией printf?
- printf(a, c);
- printf("d, s", a, c);
- printf(%a, %c);
- printf("%d, %s", a, c);

Каким типом данных будет заменён в данном случае параметр шаблона ArrType?

```
float arr[10];
template <class ArrType> long func (ArrType* arr, int n);
...
void main() {
   long v = func(arr, 5);
}
```

- int
- float
- long
- void

32. Как объявить двумерный динамический массив?
- int a[][];
- int* a;
- int* a[];
- int** a;

33. Как получить доступ к названию первого товара в магазине (поле Product, строка) в следующем примере

```
struct Shop {
   char Name[100];
   char Product[100][50];
};
Shop sh;
```

- sh.Product;
- sh.Product[0];
- sh.Product[1];
- sh.Product[0][0];
- sh.Product[0][1];

34. Как получить доступ к полю fld?

```
struct Str {
   int fld;
};
Str* s;
```

- s.fld;
- *s.fld;
- s.*fld
- *(s.fld);
- s->fld;

35. Как получить доступ к фамилии первого студента в группе (Поле Surname) в следующем примере

```
struct Student {
   char Surname[50];
   char FirstName[30];
};
struct Group {
   Student Students[25];
   char Name[20];
};
Group is00;
```

- is00.Students[0];
- is00.Students[0].Surname;
- is00.Students.Surname[0];
- is00.Surname[0];

36. Как правильно выделить память под одномерный динамический массив?

- int* arr; arr = new int [20];
- int* arr; arr = new int(20);
- int** arr; arr = new int* [20];
- int* arr; arr(20) = new;

37. Какую арифметическую операцию нельзя применять к переменным вещественного типа?

- +
- -
- *
- /
- %

38. Переменные какого типа нельзя объявлять?

- unsigned char
- wchar_t
- long double
- void

39. Под какие массивы будет выделено менее 10 байт памяти, предполагая, что используется компилятор Visual Studio 32bit?

- int a[3];
- char str[] = "123456789";
- char a[9];
- short sh[4 + 0.9];

40. Пусть задан трёхмерный массив float b[6][5][4]. Что будет собой представлять элемент a[2][3]?

- Такая запись элемента недопустима
- Число типа float
- Двумерный массив float[5][4]
- Одномерный массив float[4]

41. Пусть задан трёхмерный массив int a[4][5][6]. Что будет собой представлять элемент a[2]?

- Такая запись элемента недопустима
- Число типа int
- Двумерный массив int[5][6]
- Одномерный массив int[5]

42. Пусть объявлены следующие перегруженные функции

```
int func(int, char);
float func(double, char);
double func(float);
```

Какая из них будет вызвана в следующем коде

```
double a, b = 5;
a = func((float) b, '!');
```

- int func(int, char);
- float func(double, char);
- double func(float);
- нет подходящей функции
- неоднозначность в выборе функции

43. Размер каких массивов будет менее 5 элементов?

- int a[5];
- short a[4];
- char a[] = "1234";
- float a[3][2];
- int a[4][1]

44. Сколько раз выполнится тело данного цикла?

```
for (int i = -5; i <= -10; i++)
   cout << i << ' ';
```

- 0
- бесконечное количество раз
- 5
- 6
- 10

45. Тело каких циклов выполнится 10 раз?

- for (int i = 0; i <= 9; i++)
- for (int i = 1; i <10; i++)
- int x = 0; while (x <= 30) x+=3;
- for (int i = 0; i < 1; i += 0.1)

46. Укажите варианты, в которых допущены ошибки

- int a = 5; int* p = 0; *p = a;
- int a = 5; int* p = a; *p = 4;
- int a = 5; int* p = &a; *p = 4;
- int a = 5; int* p; p = *a;

47. Укажите двоичное значение выражения ~A ^ B при

A=10010101

B=10101001

- 11000011
- 00110101
- 10010100
- 00101001
- 11100101

48. Укажите двоичное значение выражения ~A ^ B при

A=10010101

B=10101001

- 00101000

- 00101001
- 10100000
- 01010010
- 00000101

49. Укажите инструкции for, записанные с синтаксическими ошибками
- for (int x = 0; x < 5; x+=5)
- for (int x > 0; x<5; x++)
- for (int x = 1; x< 5)
- for (int x = -1; x < 5;)
- for (int x = 0; x != 5; x--)

50. Укажите инструкции if, записанные с синтаксическими ошибками
- if x == 10
- if (x == 10)
- if (x == 10 && y != 100)
- if (x ==10) && (y != 100)
- if ((x==10) && (y != 100))

51. Укажите неверные объявления переменных
- unsigned float a = 100;
- char ch = 50;
- int a = 30, float b = 30.3;
- double = 50;
- float b, c = 0.4;

52. Укажите потоки, по умолчанию выводящие информацию на монитор
- cin
- clog
- cerr
- cout
- cmon

53. Укажите правильные объявления строк
- const char str[50];
- char str[50];
- char str[];
- char str[] = "Hello";
- const char* str = "world";
- char str[5] = "Hello";
- const char str[5] = {'H'};

54. Укажите результат вычисления выражения

a + pow(b, 3) - abs(b) * b

при a = 2, b = -1
- 2
- 3
- 0
- 5
- 4

55. Чему будет равно значение выражения x++ + x++ при x=2
- 4
- 6
- 7
- 5

56. Чему будет равно значение переменной px после выполнения следующего кода

```
int* px = (int*)1000;
px += 4;
cout << px;
```
- 1004

- 1000
- 1008
- 1016

57. Чему будет равно значение переменной x после выполнения кода?

```
void f1(int a) {
   a += a;
};
void f1(int&amp;amp; a) {
   a += a;
};
void main() {
   int x = 3;
   f1(x);
   f2(x);
   cout << x; // Чему равен x?
}
```

- 3
- 6
- 9
- 12

58. Чему будет равно значение переменной x после выполнения кода при x = 0, y = 1

```
x = x ? x++ + y : ++x + y;
```

- 2
- 1
- 3
- 0

59. Чему будет равно значение переменной x после выполнения кода при x = 1, y = 2

```
switch (x)
{
   case 0:
      x = pow(y, x + y);
   case 1:
      x = y + x * 2;
   case 2:
      x = x + y - 3;
}
```

- 3
- 8
- 4
- 0

60. Чему будет равно значение переменной x после выполнения кода при a = 2, b = 1

```
if (a <= b + 1)
   x = a + b;
else
   x = a - b;
   x = x + 1;
```

- 4
- 3
- 2
- 1

61. Чему будет равно значение переменной x после выполнения кода при a = 2

```
if (a = 6 - a)
   x = a;
```

```
else
   x = -a;
```
- 4
- -2
- 2
- 0

62. Чему будет равно значение переменной x после выполнения следующего кода?

```
void func(int x) {
   x += x * x;
}
void main() {
   int x = 2;
   func(x);
   cout << x; // Чему равен x?
}
```
- 2
- 4
- 6

63. Чему будет равно значение переменной x после выполнения цикла

```
int x = 1;
for (int i = 0; i <= 5; i++)
   x++;
   x+=2;
```
- 9
- 19
- 16
- 8

64. Чему будет равно значение переменной x после выполнения цикла

```
int x = 1;
for (int i = 0; i <= 5; i++)
   x += i;
```
- 5
- 6
- 15
- 16

65. Чему равно значение выражения

```
1 / 3 * x + y - pow(3, y)
```
при x = 3, y = 1
- -1
- -2
- 0
- 1
- 2

66. Чему равно значение неинициализированной переменной?
- 0
- 1
- NULL
- ничему не равно
- произвольному значению

67. Что называется перегрузкой функции?
- Передача в неё значений, выходящих за допустимые пределы
- Возврат из неё значения, выходящего за допустимые пределы
- Создание универсальной функции, типы аргументов которой определяются при её вызове

- Создание нескольких функций с одним именем и различными параметрами
- Создание нескольких функций с одинаковыми прототипами и различными телами.

68. Что сделает в данном коде оператор break?

```
for (int i = 0; i < 10; i++)
   for (int j = 0; j < 10; j++)
      if (i > 3 &amp;amp;&amp;amp; j < i)
         break;
```

- Прервёт цикл по i
- Прервёт цикл по j
- Прервёт оба цикла
- Перейдёт к следующей итерации цикла i
- Перейдёт к следующей итерации цикла j

69. Что сделает в данном коде оператор continue?

```
for (int i = 0; i < 10; i++)
   for (int j = 0; j < 10; j++)
      if (i > 3 &amp;amp;&amp;amp; j < i)
         continue;
```

- Прервёт цикл по i
- Прервёт цикл по j
- Прервёт оба цикла
- Перейдёт к следующей итерации цикла i
- Перейдёт к следующей итерации цикла j

ПК-30:

Блок (владеть).

70. Основные компоненты систем программирования

- Текстовый редактор
- Файловый менеджер
- Отладчик
- Маркировщик
- Компоновщик
- Транслятор
- Переводчик

71. Что делает директива препроцессора #include?

- Подключает к программе указанную библиотеку
- Вставляет в программу содержимое указанного файла
- Включает в программе указанный тип функций
- Задаёт указанный макрос

72. Какие пять подходов применяются к организации процесса создания и использования программного средства?

- Водопадный
- Исследовательское программирование
- Прототипирование
- Формальные преобразования
- Сборочное программирование
- Семантическое программирование
- Теоретическое программирование

74. Какой подход включает разработку формальных спецификаций программного средства и превращение их в программы путем корректных преобразований?

- Водопадный
- Исследовательское программирование

- Прототипирование
- Формальные преобразования
- Сборочное программирование
- Семантическое программирование
- Теоретическое программирование

75. Какой подход предполагает, что программное средства конструируется, главным образом, из компонент, которые уже существуют ?

- Водопадный
- Исследовательское программирование
- Прототипирование
- Формальные преобразования
- Сборочное программирование
- Семантическое программирование
- Теоретическое программирование

76. Какие стадии жизненного цикла программного средства различают при водопадном подходе его разработки ?

- Разработка программного средства
- Производство программных изделий
- Эксплуатация программного средства
- Анализ технического задания
- Тестирование программного средства

77. При какой модели разработки программного средства используют разбиение всей разработки на стадии, причем переход с одного этапа на другой происходит только в том случае, когда полностью завершена работа на текущем ?

- Каскадная модель
- Спиральная модель
- Кубическая модель
- Объектная модель
- Эксплуатационная модель

78. Что не относится к критериям качества программного средства ?

- функциональность
- надежность
- легкость применения
- эффективность
- сопровождаемость
- мобильнось
- универсальность

79. Свойство, характеризующее степень обладания программным средством всеми необходимыми частями и чертами, требующихся для выполнения своих явных и неявных функций ?

- Завершенность
- Точность
- Автономность
- Устойчивость
- Защищенность
- П-документированность
- Информативность

80. Свойство, характеризующее наличие в программном средстве информации, необходимой и достаточной для понимания назначения программного средства?

- Завершенность
- Точность
- Автономность
- Устойчивость
- Защищенность

- П-документированность
- Информативность

81. Мера, характеризующая способность программы выполнять возложенные на нее функции в течение определенного отрезка времени?

- Эффективность по ресурсам
- Временная эффективность
- Эффективность по устройствам
- Понятность
- Структурированность
- Эдобочитаемость
- Раширяемость

82. Мера, характеризующая способность программы с точки зрения простоты внесения изменений и доработок на всех этапах и стадиях жизненного цикла программы?

- Эффективность по ресурсам
- Временная эффективность
- Эффективность по устройствам
- Понятность
- Структурированность
- Модифицируемость
- Раширяемость

83. Свойство, характеризующее способность программного средства работать на разнообразном аппаратном обеспечении?

- Независимость от устройств
- Временная эффективность
- Эффективность по устройствам
- Понятность
- Структурированность
- Модифицируемость
- Раширяемость

84. Какие методы используются при проектировании программных средств?

- Метод восходящей разработки
- Метод нисходящей разработки
- Конструктивный подход
- Архитектурный подход
- Классический подход
- Объектно-ориентированный метод

85. Процесс многократного повторения программы с целью обнаружения ошибок - это?

- Тестирование
- Отладка
- Проектирование
- Программирование
- Алгоритмизация
- Структурирование

86. Какие методы тестирования существуют?

- Статическое тестирование
- Детерминированное тестирование
- Стохастическое тестирование
- Автономное тестирование
- Частичное тестирование
- Выборочное тестирование

87. Какие виды отладки программ используют в нашей стране?

- Автономная отладка
- Комплексная отладка
- Стохастическая отладка
- Частичная отладка
- Выборочная отладка
- Структурированная отладка

88. Какая документация предназначена для администраторов программного средства?

- Руководство по инсталляции программы
- Общее функциональное описание программного средства
- Справочник по применению программы
- Инструкция по применению программы
- Руководство по управлению программным средством
- Справочник по сопровождению программы

89. Какая документация предназначена для ординарных пользователей программного средства?

- Руководство по инсталляции программы
- Общее функциональное описание программного средства
- Справочник по применению программы
- Инструкция по применению программы
- Руководство по управлению программным средством
- Справочник по сопровождению программы

90. Какая документация содержит краткую характеристику функциональных возможностей программы?

- Руководство по инсталляции программы
- Общее функциональное описание программного средства
- Справочник по применению программы
- Инструкция по применению программы
- Руководство по управлению программным средством
- Справочник по сопровождению программы

91. Укажите этапы жизненного цикла программного средства?

- Анализ
- Проектирование
- Реализация
- Отладка и тестирование
- Сопровождение
- Документирование

92. На каком этапе жизненного цикла выявляют ошибки программы и проверяют ее работоспособность?

- Анализ
- Проектирование
- Реализация
- Отладка и тестирование
- Сопровождение
- Документирование

93. На каком этапе жизненного цикла осуществляют расширение функциональных возможностей программного средства?

- Анализ
- Проектирование
- Реализация
- Отладка и тестирование

- Сопровождение
- Документирование

94. На каком этапе жизненного цикла непосредственно реализуют кодирование текстов программ?
- Анализ
- Проектирование
- Реализация
- Отладка и тестирование
- Сопровождение
- Документирование

95. Среда, служащая для организации диалога программного средства с пользователем на основе графического многооконного представления данных?
- Графический пользовательский интерфейс
- Консольный пользовательский интерфейс
- Мультимедийный пользовательский интерфейс
- Текстовой пользовательский интерфейс
- Модульный пользовательский интерфейс
- Векторный пользовательский интерфейс

96. Какой раздел стандарта ЕСПД содержит описание программы?
- ГОСТ 19.402.-78
- ГОСТ 19.101.-77
- ГОСТ 19.101.-77
- ГОСТ 19.401.-78
- ГОСТ 19.404.-79
- ГОСТ 19.502.-78
- ГОСТ 19.701.-90

97. Какой раздел стандарта ЕСПД содержит руководство системному программисту?
- ГОСТ 19.503.-79
- ГОСТ 19.101.-77
- ГОСТ 19.101.-77
- ГОСТ 19.401.-78
- ГОСТ 19.404.-79
- ГОСТ 19.502.-78
- ГОСТ 19.701.-90

98. Какой раздел стандарта ЕСПД содержит руководство оператору?
- ГОСТ 19.505.-79
- ГОСТ 19.101.-77
- ГОСТ 19.101.-77
- ГОСТ 19.401.-78
- ГОСТ 19.404.-79
- ГОСТ 19.502.-78
- ГОСТ 19.701.-90

99. Какой раздел стандарта ЕСПД содержит руководство программиста?
- ГОСТ 19.504.-79
- ГОСТ 19.101.-77
- ГОСТ 19.101.-77
- ГОСТ 19.401.-78

- ГОСТ 19.404.-79
- ГОСТ 19.502.-78
- ГОСТ 19.701.-90

100. Какой раздел стандарта ЕСПД содержит пояснительную записку?

- ГОСТ 19.404.-79
- ГОСТ 19.505.-79
- ГОСТ 19.101.-77
- ГОСТ 19.101.-77
- ГОСТ 19.401.-78
- ГОСТ 19.502.-78
- ГОСТ 19.701.-90

Полный перечень тестовых заданий с указанием правильных ответов, размещен в банке вопросов на информационно-образовательном портале института по ссылке https://www.mivlgu.ru/iop/question/edit.php?cmid=56751

Оценка рассчитывается как процент правильно выполненных тестовых заданий из их общего числа.