

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**Федеральное государственное автономное  
образовательное учреждение высшего образования  
«Национальный исследовательский Нижегородский государственный университет  
им. Н.И. Лобачевского»**

Институт информационных технологий, математики и механики  
*(факультет / институт / филиал)*

---

УТВЕРЖДЕНО  
президиумом Ученого совета ННГУ  
от 14.12.2021 г протокол № 4

**Рабочая программа дисциплины**

**Инструменты программирования**

---

*(наименование дисциплины (модуля))*

Уровень высшего образования

Бакалавриат

---

*(бакалавриат / магистратура / специалитет)*

Направление подготовки / специальность

**09.03.04 Программная инженерия**

---

*(указывается код и наименование направления подготовки / специальности)*

Направленность образовательной программы

**Разработка программно-информационных систем**

---

*(указывается профиль / магистерская программа / специализация)*

Форма обучения

очная

---

*(очная / очно-заочная / заочная)*

Нижегород

2022 год

## 1. Место дисциплины в структуре ОПОП

Дисциплина относится к части, формируемой участниками образовательных отношений.

№ варианта	Место дисциплины в учебном плане образовательной программы	Стандартный текст для автоматического заполнения в конструкторе РПД
2	Блок 1. Дисциплины (модули) Часть, формируемая участниками образовательных отношений	Дисциплина <b>Б1.В.14 Инструменты программирования</b> относится к части ООП направления подготовки <b>09.03.04 Программная инженерия</b> , формируемой участниками образовательных отношений.

## 2. Планируемые результаты обучения по дисциплине, соотнесенные с планируемыми результатами освоения образовательной программы (компетенциями и индикаторами достижения компетенций)

Формируемые компетенции (код, содержание компетенции)	Планируемые результаты обучения по дисциплине (модулю), в соответствии с индикатором достижения компетенции		Наименование оценочного средства
	Индикатор достижения компетенции (код, содержание индикатора)	Результаты обучения по дисциплине	
<b>ПК-14:</b> <i>Способен применять методы контроля и качества проекта и программных продуктов</i>	<b>ПК-14.1.</b> <i>Знает основы теории тестирования и методы оценки качества программных систем</i>	<i>Знать:</i> <i>Основные текстовые форматы.</i> <i>Понятие регулярного выражения, основные правила формирования регулярных выражений.</i> <i>Основные команды командной строки.</i> <i>Возможности скриптовых языков для автоматизации сборки и компиляции программ.</i> <i>Понятие интегрированной среды разработки и основных компонент среды разработки.</i> <i>Понятие системы контроля версий, ее назначение. Цикл разработки программ с использованием системы контроля версий.</i> <i>Процедуру организации отладки и тестирования программ.</i> <i>Инструменты инспекции кода.</i>	<i>Собеседование,</i>

	<p>ПК-14.2. Умеет формулировать задачи и требования к результатам работы и методам их выполнения</p> <p>ПК-14.3. Умеет выявлять конфликты интересов и требований к системе</p> <p>ПК-14.4. Умеет проводить тестирование ПО</p>	<p>Уметь:</p> <p>Работать с одним из текстовых редакторов, рассмотренном в лекционном материале.</p> <p>Работать с командной строкой.</p> <p>Разрабатывать скрипты для автоматизации сборки и компиляции программ.</p> <p>Работать с интегрированной средой разработки.</p> <p>Работать с системой контроля версий.</p> <p>Отлаживать и тестировать программы. Разрабатывать автоматические тесты.</p> <p>Использовать инструменты инспекции кода.</p> <p>Пользоваться навыками работы с одним широко известных текстовых редакторов.</p> <p>Пользоваться навыками работы с командной строкой.</p> <p>Пользоваться навыками использования скриптовых языков для автоматизации сборки и компиляции программ.</p> <p>Пользоваться навыками отладки и тестирования программ, а также разработки автоматических тестов на примере GoogleTest.</p>	Задача
	<p>ПК-14.5. Владеет методами проверки корректности и эффективности проектных решений</p> <p>ПК-14.6. Владеет методами контроля версий и верификацию выпусков программного продукта</p>	<p>Использовать инструменты инспекции кода.</p> <p>Пользоваться навыками использования системы контроля версий Git.</p> <p>Пользоваться навыками использования инструментов инспекции кода на примере GitHub..</p>	Задача

### 3. Структура и содержание дисциплины

#### 3.1. Трудоемкость дисциплины

	Очная форма обучения
<b>Общая трудоемкость</b>	<b>2 ЗЕТ</b>
<b>Часов по учебному плану</b>	<b>72</b>
<b>в том числе</b>	
<b>аудиторные занятия (контактная работа):</b>	<b>33</b>
- занятия лекционного типа	<b>32</b>
- занятия семинарского типа	
- текущий контроль (КСР)	<b>1</b>

самостоятельная работа	39
Промежуточная аттестация –зачет	

### 3.2. Содержание дисциплины

Наименование и краткое содержание разделов и тем дисциплины (модуля), форма промежуточной аттестации по дисциплине (модулю)	Всего (часы)	в том числе				
		контактная работа (работа во взаимодействии с преподавателем), часы				Самостоятельная работа обучающегося, часы
		из них				
		Занятия лекционного типа	Занятия семинарского типа	Занятия лабораторного типа	Всего контактных часов	
Рабочее место программиста	4	2			2	2
Текстовые форматы	4	2			2	2
Обработка текста и регулярные выражения	4	2			2	2
Текстовые редакторы	4	2			2	2
Автоматизация: командная строка	4	2			2	2
Автоматизация: скриптовые языки	4	2			2	2
Системы контроля версий	4	2			2	2
Интегрированные среды разработки	4	2			2	2
Описание и построение проектов	4	2			2	2
Анализ бинарных модулей	4	2			2	2
Контроль качества кода	4	2			2	2
Отладка	4	2			2	2
Тестирование	6	2			2	4
Непрерывная интеграция	6	2			2	4
Профилирование и оптимизация производительности	6	2			2	4
Командная разработка. Формирование сообщества	5	2			2	3
Текущий контроль (КСР)	1				1	

Промежуточная аттестация –зачет						
Итого	72	32		0	33	39

Текущий контроль успеваемости реализуется в формах опросов на занятиях семинарского типа  
Промежуточная аттестация проходит в традиционной форме (зачет).

#### 4. Учебно-методическое обеспечение самостоятельной работы обучающихся

Используются образовательные технологии в форме лекций и практических занятий. При чтении лекций используются электронных презентаций.

Выполнение практических работ на следующие темы:

- Настройка окружения разработки.
- Проектирование программного интерфейса (API), создание пользовательской документации.
- Разработка функциональности.
- Интеграция кода в общий проект.
- Создание модульных тестов.

Контрольные вопросы и задания для проведения текущего контроля и промежуточной аттестации по итогам освоения дисциплины приведены в п. 5.2.

Для обеспечения самостоятельной работы обучающихся используется электронный курс (Инструменты программирования, <https://e-learning.unn.ru/course/view.php?id=5827>), созданный в системе электронного обучения ННГУ - <https://e-learning.unn.ru/>

#### 5. Фонд оценочных средств для промежуточной аттестации по дисциплине (модулю), включающий:

##### 5.1. Описание шкал оценивания результатов обучения по дисциплине

Уровень сформированности компетенций (индикатора достижения компетенций)	Шкала оценивания сформированности компетенций						
	плохо	неудовлетворительно	удовлетворительно	хорошо	очень хорошо	отлично	превосходно
	Не зачтено		Зачтено				
<u>Знания</u>	Отсутствие знаний теоретического материала. Невозможность оценить полноту знаний вследствие отказа обучающегося от ответа	Уровень знаний ниже минимальных требований. Имели место грубые ошибки.	Минимально допустимый уровень знаний. Допущено много негрубых ошибок.	Уровень знаний в объеме, соответствующем программе подготовки. Допущено несколько негрубых ошибок	Уровень знаний в объеме, соответствующем программе подготовки. Допущено несколько незначительных ошибок	Уровень знаний в объеме, соответствующем программе подготовки, без ошибок.	Уровень знаний в объеме, превышающем программу подготовки.

<u>Умения</u>	Отсутствие минимальных умений. Невозможность оценить наличие умений вследствие отказа обучающегося от ответа	При решении стандартных задач не продемонстрированы основные умения.  Имели место грубые ошибки.	Продемонстрированы основные умения. Решены типовые задачи с негрубыми ошибками. Выполнены все задания, но не в полном объеме.	Продемонстрированы все основные умения. Решены все основные задачи с негрубыми ошибками. Выполнены все задания, в полном объеме, но некоторые с недочетами.	Продемонстрированы все основные умения. Решены все основные задачи. Выполнены все задания, в полном объеме, но некоторые с недочетами.	Продемонстрированы все основные умения, решены все основные задачи с отдельными несущественными недочетами, выполнены все задания в полном объеме.	Продемонстрированы все основные умения, решены все основные задачи. Выполнены все задания, в полном объеме без недочетов
<u>Навыки</u>	Отсутствие владения материалом. Невозможность оценить наличие навыков вследствие отказа обучающегося от ответа	При решении стандартных задач не продемонстрированы базовые навыки.  Имели место грубые ошибки.	Имеется минимальный набор навыков для решения стандартных задач с некоторыми недочетами.	Продемонстрированы базовые навыки при решении стандартных задач с некоторыми недочетами	Продемонстрированы базовые навыки при решении стандартных задач без ошибок и недочетов.	Продемонстрированы навыки при решении нестандартных задач без ошибок и недочетов.	Продемонстрирован творческий подход к решению нестандартных задач.

### Шкала оценки при промежуточной аттестации

Оценка		Уровень подготовки
зачтено	Превосходно	Все компетенции (части компетенций), на формирование которых направлена дисциплина, сформированы на уровне не ниже «превосходно»
	Отлично	Все компетенции (части компетенций), на формирование которых направлена дисциплина, сформированы на уровне не ниже «отлично», при этом хотя бы одна компетенция сформирована на уровне «отлично»
	Очень хорошо	Все компетенции (части компетенций), на формирование которых направлена дисциплина, сформированы на уровне не ниже «очень хорошо», при этом хотя бы одна компетенция сформирована на уровне «очень хорошо»
	Хорошо	Все компетенции (части компетенций), на формирование которых направлена дисциплина, сформированы на уровне не ниже «хорошо», при этом хотя бы одна компетенция сформирована на уровне «хорошо»
	Удовлетворительно	Все компетенции (части компетенций), на формирование которых направлена дисциплина, сформированы на уровне не ниже «удовлетворительно», при этом хотя бы одна компетенция сформирована на уровне «удовлетворительно»
не зачтено	Неудовлетворительно	Хотя бы одна компетенция сформирована на уровне «неудовлетворительно», ни одна из компетенций не сформирована на уровне «плохо»
	Плохо	Хотя бы одна компетенция сформирована на уровне «плохо»

### 5.2. Типовые контрольные задания или иные материалы, необходимые для оценки результатов обучения

### 5.2.1 Контрольные вопросы

Вопрос	Код компетенции (согласно РПД)
1. Общее назначение инструментов, примеры.	ПК-14
2. Признаки "хороших" инструментов, с пояснениями.	
3. Примеры практик Программной инженерии, их суть.	
4. Приведите примеры инструментов, помогающих применять практики.	
5. Диаграмма каскадной модели жизненного цикла.	
6. Диаграмма работы программиста над задачей.	
7. Определение системы контроля версий (СКВ)	
8. Основные функции/возможности современных СКВ	
9. Преимущества DVCS	
10. Centralized Workflow (диаграмма, достоинства и недостатки)	
11. Integration Manager Workflow (диаграмма, достоинства и недостатки)	
12. Dictator and Lieutenants Workflow (диаграмма, достоинства и недостатки)	
13. Модель ветвления GitFlow	
14. Рабочий процесс (модель ветвления), используемый в компании GitHub	
15. Базовые принципы корректной работы с СКВ	
16. Простые истины планирования	
17. Практические рекомендации при учете задач (issue tracking)	
18. Что нельзя протестировать автоматически?	
19. Классификация тестов по назначению.	
20. Современная стратегия тестирования (основные 5 утверждений).	
21. Основные возможности фреймворков модульного тестирования.	
22. Критерии хорошего теста.	

23. Возможности Google Test.	
24. Порядок использования Google Test.	
25. Определение непрерывной интеграции	
26. Задачи выделенного сервера	
27. Эволюция взглядов на непрерывную интеграцию	
28. Travis CI, преимущества и недостатки	
29. BuildBot, преимущества и недостатки	
30. Определение интегрированной среды разработки (ИСР)	
31. Отличия ИСР от редакторов исходного кода	
32. Основные функции/возможности современных ИСР	
33. История развития билд-систем	
34. Плюсы и минусы Makefile	
35. Плюсы и минусы CMake	
36. Преимущества и недостатки простого текста.	
37. Преимущества и недостатки бинарного формата.	
38. Примеры ситуаций, когда удобно использовать TXT, XML, YAML, JSON.	
39. Легковесные языки разметки. Примеры, назначение, преимущества и недостатки.	
40. Синтаксис Markdown (заголовки, стили, списки, ссылки).	
41. Примеры использования Markdown, в том числе нестандартные. В чем преимущество использования Markdown в каждой из этих ситуаций.	
42. Как будет выглядеть команда в Vim для:  – Создания 10 копий текущей строки  – Перевода всей строки в верхний регистр (капитализация)	
43. Предложите регулярное выражение для поиска:  – Имен всех классов в вашем C++ проекте  – Поиска дат в формате `2013-09-18` или `14-01-01`  – IP-адресов, номеров банковских карт, HEX-представления чисел типа `int`	



44. Предложите командную строку для:  – Поиска всех вызовов виртуального метода в директории с исходниками  – Печати всех заголовков первого и второго уровня в файле Markdown (#-нотация)	
45. Приведите примеры внутренних документов	
46. Приведите примеры внешних документов	
47. Распределение ролей при работе над документацией	
48. Виды автоматических проверок документации и способы их реализации	
49. Популярные форматы внутренних документов, их достоинства и недостатки	
50. Содержание/подразбиение README файлов	
51. Ключевые принципы при работе с документацией	
52. Какие преимущества дает автоматизация	
53. Типичные классы задач на автоматизацию (в работе программиста)	
54. Философия UNIX	
55. Преимущества UNIX при автоматизации	
56. Суть деятельности профессионального программиста	

#### 5.2.2. Типовые задачи для оценки сформированности компетенции ПК-14

1. Настройка окружения разработки.
2. Проектирование программного интерфейса (API), создание пользовательской документации.
3. Разработка функциональности.
4. Интеграция кода в общий проект.
5. Создание модульных тестов.

#### 6. Учебно-методическое и информационное обеспечение дисциплины

##### а) Основная литература:

1. Страуструп Б. Курс «Язык программирования C++ для профессионалов». – ИНТУИТ. <http://www.intuit.ru/studies/courses/98/98/info>

2. Сафонов А. Возможности Visual Studio 2013 и их использование для облачных вычислений. Лекция 1: Концепция современной интегрированной среды разработки приложений. – ИНТУИТ.

<http://www.intuit.ru/studies/courses/13996/1223/lecture/23386>

3. Котляров В. Основы тестирования программного обеспечения. – ИНТУИТ.

<http://www.intuit.ru/studies/courses/48/48/info>

б) Дополнительная литература:

1. КанМ. Основы программирования на JavaScript. – ИНТУИТ.

<http://www.intuit.ru/studies/courses/1093/132/info>.

2. Баженова И., Сухомлин В. Курс «Введение в программирование». – ИНТУИТ.

<http://www.intuit.ru/studies/courses/27/27/info>

3. Гниденко, И. Г. Технологии и методы программирования : учебное пособие для прикладного бакалавриата / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — М. : Издательство Юрайт, 2017. — 235 с. — (Серия : Бакалавр. Прикладной курс). — ISBN 978-5-534-02816-4. — Режим доступа : [www.biblio-online.ru/book/E0A213EF-E61B-4F8B-A4E5-D75FD4E72E10](http://www.biblio-online.ru/book/E0A213EF-E61B-4F8B-A4E5-D75FD4E72E10).

в) Программное обеспечение и Интернет-ресурсы

1. Интегрированная среда разработки: Borland Builder, Microsoft Visual Studio, Eclipse, Qt Creator, Xcode. Одна из перечисленных на выбор студента.
2. Клиент системы контроля версий Git: git-scm [<https://git-scm.com>], GitHub Desktop [<https://desktop.github.com>]. Один из перечисленных на выбор студента.
3. GitHub [<http://github.com>].
4. Google Test [<https://github.com/google/googletest>].
5. Travis CI [<https://travis-ci.org>].
6. Redmine [<http://www.redmine.org>]. (свободное ПО).

## **7. Материально-техническое обеспечение дисциплины**

Помещения представляют собой учебные аудитории для проведения учебных занятий, предусмотренных программой (лекционного типа), оснащенные оборудованием и техническими средствами обучения.

Помещения для самостоятельной работы обучающихся оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечены доступом в электронную информационно-образовательную среду.

Программа составлена в соответствии с требованиями ОС ННГУ по направлению 09.03.04 Программная инженерия.

Автор К.В. Корняков

Рецензент (ы) \_\_\_\_\_

Заведующий кафедрой \_\_\_\_\_ Р.Г. Стронгин

Программа одобрена на заседании методической комиссии института информационных технологий, математики и механики от 01.12.2021 года, протокол № 2