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

УТВЕРЖДЕНО
решением Ученого совета ННГУ
протокол от
«30» ноября 2022 г. № 13

Рабочая программа дисциплины / Work program of the course

Программирование на скриптовых языках
Programming in Scripting Languages

Уровень высшего образования / Level of higher education
магистратура / master's degree

Направление подготовки / Training direction
09.04.04 Программная инженерия / Software Engineering

Направленность подготовки / Training profile
Инженерия программного обеспечения
Software Engineering

Форма обучения / Form of education
очная / full-time

Нижний Новгород / Nizhniy Novgorod
2023

1. The place of discipline in the structure of the whole educational program (ОПОП)

Discipline «Programming in Scripting Languages» is an elective course in part, formed by education participants in Block 1 «Disciplines (modules)» of training direction 09.04.04 «Software Engineering» of training profile «Digital transformation technologies». Discipline is taught in the second semester. Discipline complexity is 10 credit units, duration is 360 hours, course credit and exam are required.

№ варианта	Место дисциплины в учебном плане образовательной программы	Стандартный текст для автоматического заполнения в конструкторе РПД
1	Блок 1. Дисциплины (модули) Часть, формируемая участниками образовательных отношений	Дисциплина Б1.В.ДВ.02.01 «Программирование на скриптовых языках» относится к части ООП направления подготовки 09.04.04 «Программная инженерия», формируемой участниками образовательных отношений (дисциплины по выбору Б1.В.ДВ.2).

2. Planned learning results for discipline, mapped on planned results of the whole educational program (competencies and their achievement indicators)

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****	
ПК-12. Владеет методами поддержки разработки архитектуры ИС	ПК-12.1. Знает инструменты и методы проектирования архитектуры ИС	Знать инструменты и методы проектирования архитектуры ИС	*Собеседование Вопросы к экзамену*
	ПК-12.2. Умеет проектировать архитектуры ИС	Уметь проектировать архитектуры ИС	*Собеседование Задание Вопросы к экзамену*
	ПК-12.3. Имеет практический опыт проверки (верификации) архитектуры ИС	Иметь практический опыт проверки (верификации) архитектуры ИС	*Собеседование Задание Вопросы к экзамену*

3. Discipline Structure and Content

3.1. Discipline Complexity

The volume of discipline (module) includes:
10 credit units, total duration is **360** hours, which include:
67 hours of **face-to-face** learning with a teacher:
32 hours of lectures,
32 hours of seminars,
3 hours of course results evaluation
257 hours of **independent work** of student

3.2 Discipline Content

Наименование и краткое содержание разделов и тем дисциплины, **форма промежуточной аттестации по дисциплине** **Names and annotation of chapters and topics of discipline,** **form of course results evaluation**	**Всего (часы)** **Total (hours)**	в том числе / in details				
		контактная работа (работа во взаимодействии с преподавателем), часы / **face-to-face work (learning with a teacher)** из них / including				**Самостоятельная работа студента (часы) / Independent work (hours)**
		Занятия лекционного типа / Lectures	**Занятия семинарского типа / Seminars**	**Занятия лабораторного типа / Lab works**	**Всего контактных часов / Total face-to-face hours**	
Введение в язык программирования Python Introduction into Python programming language	10	2		2	4	6
Объектно-ориентированные средства в языке Python Object-oriented programming in Python	18	4		2	6	12
Парсинг систем разметки на языке Python Parsing markup languages in Python	18	2		4	6	12
Использование COM объектов в скриптовых языках Using COM objects in scripting languages	10	2		2	4	6
Виды тестирования, оценка качества тестирования, средства статического анализа Kinds of testing, testing quality evaluation, static analysis tools	24	4		2	6	18
Автоматизация тестирования графического пользовательского интерфейса Graphical User Interface (GUI) testing automation	27	2		4	6	21
Текущий контроль (КСР) Evaluation	1				1	
Итого / Total	**360**	**32**	**0**	**32**	**67**	**257**
Промежуточная аттестация –Экзамен /Зачет / Kind of course evaluation is a course credits.						

Практические занятия (семинарские занятия /лабораторные работы) организуются, в том числе в форме практической подготовки, которая предусматривает участие обучающихся в выполнении отдельных элементов работ, связанных с будущей профессиональной деятельностью.

Практическая подготовка предусматривает: выполнение тем практических заданий, подготовку вопросов к экзамену.

На проведение практических занятий (семинарских занятий /лабораторных работ) в форме практической подготовки отводится 32 часа.

Практическая подготовка направлена на формирование и развитие:

- практических навыков в соответствии с профилем ОП: создание и сопровождение архитектуры программных средств, разработка и тестирование программного обеспечения;
- компетенций – ПК-12

Текущий контроль успеваемости реализуется в формах опросов на занятиях лабораторного типа.

Промежуточная аттестация проходит в традиционных формах (зачет/экзамен).

4. Учебно-методическое обеспечение самостоятельной работы обучающихся

Самостоятельная работа обучающихся осуществляется в виде работы с рекомендованной обязательной и дополнительной литературой, подготовке к лекциям, подготовке к зачету и выполнения лабораторных работ. Контрольные вопросы и задания для проведения текущего контроля и промежуточной аттестации по итогам освоения дисциплины приведены в п. 5.2.

5. Фонд оценочных средств для промежуточной аттестации по дисциплине, 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				
Знания	Отсутствие знаний теоретического материала. Невозможность оценить	Уровень знаний ниже минимальных требований. Имели место грубые	Минимально допустимый уровень знаний. Допущено много негрубых	Уровень знаний в объеме, соответствующем программе подготовки. Допущено	Уровень знаний в объеме, соответствующем программе подготовки. Допущено	Уровень знаний в объеме, соответствующем программе подготовки,	Уровень знаний в объеме,

	полноту знаний вследствие отказа обучающегося от ответа	ошибки.	ошибки.	несколько негрубых ошибок	несколько несущественных ошибок	без ошибок.	превышающем программу подготовки.
Умения	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 несущественным недочетами, выполнены все задания в полном объеме.	Продемонстрированы все основные умения,. Решены все основные задачи. Выполнены все задания, в полном объеме без недочетов
Навыки	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	Неудовлетворитель-но	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 / Typical problems and other materials necessary for course results evaluation

5.2.1 Контрольные вопросы к зачету/экзамену / Problems for course results evaluation

Вопрос / Problem	Код формируемой компетенции / Competency code
1. Реализовать скрипт на языке Python для прямого сравнения двух файлов независимо от порядка строк. Имена файлов задаются позиционными аргументами командной строки. Например, следующие файлы одинаковы с точностью до порядка строк: файл1.txt: это тестовый файл файл тестовый сие есть это ещё одна строка файл2.txt: файл тестовый сие есть это ещё одна строка это тестовый файл Таким образом команда **python compare.py файл1.txt файл2.txt** должна выдать сообщение о том, что файлы одинаковы в приведённом смысле. Если они не одинаковы, выводить для каждого файла список строк, уникальных для этого файла. Например: файл1.txt (unique lines): *этой строки нет во втором файле* 1. Implement Python script for 2 files comparison not aware to lines order. File names are given as positional command line arguments. For example, next files are equal by unordered set of lines: file1.txt: this is test file file of test is this? one more line file2.txt: file of test is this? one more line this is test file Hence command **python compare.py file1.txt file2.txt** shall print message that 2 files are equal by mentioned definition.If they are not equal, it shall print set of lines unique for each file. Example: file1.txt (unique lines): *there is no such line in the second file*	ПК-12
2. Написать следующий скрипт на языке Python. Ключом командной строки “--bad-words” задан список “плохих” слов, разделённых точкой с запятой (;). Ключом	ПК-12

-f|--file задан проверяемый файл. Написать скрипт, который выводит все строки файла содержащие хотя бы одно плохое слово. Например, так: > **python check_bad_words.py --bad-words error;warning;assert --file test.txt** Bad word “error” was found at line 3: Unknown error occurred Bad word “warning” was found at line 7: Warning: function printf is deprecated, use printf_s instead В случае, если ключ --file не задан, делать эту проверку для всех файлов *.txt в текущей папке (без углубления в подпапки). 2. Implement the following Python script:command line argument “--bad-words” gives a set of “bad” words separated by semicolon (;). Argument “-f|--file” gives input file. Write script that prints all lines of input file that contain at least one “bad” word. Example: > **python check_bad_words.py --bad-words error;warning;assert --file test.txt** Bad word “error” was found at line 3: Unknown error occurred Bad word “warning” was found at line 7: Warning: function printf is deprecated, use printf_s instead In case “--file” argument is not provided, make this check for all *.txt files in current directory without walking subdirectories.	
3. Написать скрипт, сравнивающий два файла (эталон и проверочный) так, чтобы каждая строка в проверочном файле либо точно совпадала с такой же в эталоне, либо соответствовала эталонному регулярному выражению. В случае неудачного сравнения выводить все строки, в которых есть несовпадение, и ожидаемую строку-эталон для каждой. Скрипт принимает два ключа в командной строке: --gold <имя_эталонного_файла> и --target <имя_проверяемого_файла>. Пример эталона: *Build number: (\d+)* *Product name: My fucking program (R)* Пример удовлетворяющего ему проверочного файла: *Build number: 45107* *Product name: My fucking program (R)* Пример «плохого» проверочного файла: *Build number: None* *Product name: My excellent program (C)* 3. Write Python script that compares 2 files (gold and test ones) so that every line in test file is equal to corresponding line in gold file or it does match regular expression at corresponding line in gold file. In case of failure script shall print every failed line and their gold line or regular expression. Script shall take 2 command line arguments: --gold <gold file name> and --target <test file name>. Example of gold file: *Build number: (\d+)* *Product name: My fucking program (R)* Example of successfully matched test file: *Build number: 45107* *Product name: My fucking program (R)*	ПК-12

Example of non-matched test file: *Build number: None* *Product name: My excellent program (C)*	

5.2.2. Типовые задания для оценки сформированности компетенции ПК-12

1. Написать скрипт, использующий библиотеку pywinauto, который запускает браузер Chrome (язык: English), авторизуется в тестовом аккаунте Google Диска (рекомендуется) или Яндекс-Диска, открывает в explorer.exe текущую папку и перетаскивает тестовый файл (например, test.zip) в окно браузера, контролируя завершение загрузки.
2. Используя библиотеку pywinauto, написать скрипт, который открывает Computer Management окно, выводит список имеющихся групп пользователей, заходит в свойства группы Administators и выводит список пользователей, являющихся администраторами.
3. В рамках библиотеки pywinauto реализовать класс RichEditWrapper для backend=”win32” (документация). Методы враппера должны покрывать всю функциональность по работе с текстом и Undo/Redo. Написать модульные тесты и сделать pull request.
4. Реализовать скрипт на языке Python, который считывает XML-структуру проектного файла MS Visual Studio 2008 (.vcproj) и формирует новый проектный файл со следующими свойствами:
   - все .cpp файлы исключены из сборки для всех конфигураций (но остаются в проекте); все .cpp файлы включаются (#include) в один _merged_.cpp, который добавляется к проекту и включается в сборку.
5. Реализовать скрипт на языке Python, который считывает XML-структуру проектного файла MS Visual Studio 2010 (.vcxproj) и формирует новый проектный файл со следующими свойствами:
   - все .cpp файлы исключены из сборки для всех конфигураций (но остаются в проекте); все .cpp файлы включаются (#include) в несколько _merged_N_.cpp файлов (каждый – в свой), которые добавляются к проекту и включаются в сборку. Максимальное N – число логических процессоров на машине.

5.2.3. Типовые вопросы для собеседования для оценки сформированности компетенции ПК-12

1. Основы синтаксиса языка Python: условия, циклы, встроенные типы данных, изменяемые и неизменяемые типы данных.
2. Работа с путями и строками: объединение и разделение строк, замена/удаление подстроки, конкатенация и форматирование строк (сравнение нескольких способов).
3. Составные типы данных (коллекции): вставка, удаление элементов, сортировка, поиск, копирование.
4. Объектно-ориентированные средства языка Python: классы в Python 2.x и Python 3.x, публичные и приватные методы, разновидности методов.
5. Операторы в Python, контекстный менеджер, интроспекция объектов и классов.
6. Чтение-запись в файл, работа с папками, запуск внешних программ и работа с stdin/stdout/stderr.

6. Учебно-методическое и информационное обеспечение дисциплины

а) основная литература:

1. Язык программирования Python. Открытый курс Intuit. [http://www.intuit.ru/studies/courses/49/49/info]

2. Лидия Городняя. Основы функционального программирования. ИНТУИТ: http://www.intuit.ru/studies/courses/29/29/info

---

б) дополнительная литература:

1. Иван Хахаев. Практикум по алгоритмизации и программированию на Python. ИНТУИТ: http://www.intuit.ru/studies/courses/3489/731/info
2. Чарльз Северенс. Введение в программирование на Python. ИНТУИТ: http://www.intuit.ru/studies/courses/12179/1172/info
3. Юрий Денисов. Текстовый ввод-вывод. ИНТУИТ: http://www.intuit.ru/studies/courses/640/496/info

---

в) программное обеспечение и Интернет-ресурсы

1. Официальный сайт о языке Python. [http://www.python.org/].
2. Библиотека автоматизации GUI тестирования pywinauto. [http://pywinauto.github.io/]
3. Язык программирования Python. Открытый курс на Stepik. Институт биоинформатики. [https://stepik.org/course/67/]
4. Python: основы и применение. Открытый курс на Stepik. Институт биоинформатики. [https://stepik.org/course/512/]
5. Free Interactive Python Tutorial (англ.). [https://www.learnpython.org/]
6. Соревновательные задачи на языке Python (англ., HackerRank). [https://www.hackerrank.com/domains/python/py-introduction]
7. Алексей Задойный. Практикум: математика и Python для анализа данных. Открытый курс на Stepik. [https://stepik.org/course/3356/]

7. Материально-техническое обеспечение дисциплины (модуля)

Помещения представляют собой учебные аудитории для проведения учебных занятий, предусмотренных программой, оснащенные оборудованием и техническими средствами обучения: компьютерный класс, проектор, экран.

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

Учебная и научная литература, учебно-методические материалы, представленные в библиотечном фонде, в электронных библиотеках и на кафедре Математического обеспечения и суперкомпьютерных технологий.

Программа составлена в соответствии с требованиями ОС ВО ННГУ с учетом рекомендаций ФГОС ВО по направлению 09.04.04 Программная инженерия.

Автор: DevOps Engineer, NVIDIA, В.В. Рябов

Рецензент: к.т.н., доцент кафедры ИАНИ, Васин Д.Ю.

Заведующий кафедрой: д.ф.-м.н, проф. заведующий кафедрой МОСТ Стронгин Р.Г

Программа одобрена на заседании методической комиссии института информационных технологий, математики и механики
от 30 ноября 2022 года, протокол № 3.