МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский ядерный университет «МИФИ»
Обнинский институт атомной энергетики –
филиал федерального государственного автономного образовательного учреждения высшего образования
«Национальный исследовательский ядерный университет «МИФИ»
(ИАТЭ НИЯУ МИФИ)

Утверждено на заседании
УМС ИАТЭ НИЯУ МИФИ
Протокол №2-8/2024От 30.08.2024

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

Версионирование и коллоборативная разработка с помощью Git
Шифр, название дисциплины

01.04.02 «Прикладная математика и информатика»
Шифр, название специальности/направления подготовки

Математическое моделирование и прикладной анализ данных
Название программы магистратуры

магистр
(Квалификация (степень) выпускника)

Форма обучения: очная

г. Обнинск 2024 г.

Программа составлена в соответствии с требованиями образовательного стандарта высшего образования национального исследовательского ядерного университета «МИФИ» по направлению подготовки 01.04.02 – Прикладная математика и информатика. (квалификация (степень) магистр).

Программу составил:

_____ С.В. Ермаков, доцент, к.ф.-м.н, доцент

Рецензент:

_____ Г.Е. Деев, доцент, к.ф.-м.н, доцент

Программа рассмотрена на заседании ОИКС

(протокол № 5/7 от «30» июля от 2024 г.)

Руководитель направления подготовки 01.04.02
«Прикладная математика и информатика»

_____ Ермаков С.В.

«____»_____2024 г.

1. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

В результате освоения ООП магистратуры обучающийся должен овладеть следующими результатами обучения по дисциплине:

Коды компетенций	Результаты освоения ООП *Содержание компетенций**	Перечень планируемых результатов обучения по дисциплине**
ОПК-2	Способен совершенствовать и реализовывать новые математические методы решения прикладных задач	З-ОПК-2 Знать основные понятия, математические методы решения прикладных задач, принципы математического моделирования и методы верификации. У-ОПК-2 Уметь применять полученную теоретическую базу для решения практических задач В-ОПК-2 Владеть основными математическими методами решения прикладных задач
ОПК-1	Способен решать актуальные задачи фундаментальной и прикладной математики	З-ОПК-1 Знать актуальные задачи фундаментальной и прикладной математики, методы математического моделирования. У-ОПК-1 Уметь использовать методы математического моделирования для решения задач фундаментальной и прикладной математики. В-ОПК-1 Владеть методами математического моделирования и основами их использования.

2. Место дисциплины в структуре ООП магистратуры

Дисциплина реализуется в рамках вариативной части.

Для освоения дисциплины необходимы компетенции, сформированные в рамках изучения следующих дисциплин: Linux, Python для анализа данных

Дисциплина изучается на 1 курсе в 1 семестре.

3. Объем дисциплины в зачетных единицах с указанием количества академических часов, выделенных на контактную работу обучающихся с преподавателем (по видам занятий) и на самостоятельную работу обучающихся

Общая трудоемкость (объем) дисциплины составляет 4 зачетных единиц (з.е.), 144 академических часа.

3.1. Объём дисциплины по видам учебных занятий (в часах)

	Семестр		
	№ 1	**№ 2**	**Всего**
	Количество часов на вид работы:		
Контактная работа обучающихся с преподавателем			

Аудиторные занятия *(всего)*	**32**		**32**	
В том числе:				
лекции	16		16	
практические занятия	16		16	
лабораторные занятия				
Промежуточная аттестация				
В том числе:				
зачет	Зачет с оц.		Зачет с оц.	
экзамен	-		-	
Самостоятельная работа обучающихся *(всего)*	**112**		**112**	
В том числе:				
проработка учебного (теоретического) материала	28		28	
выполнение индивидуальных заданий	28		28	
подготовка ко всем видам контрольных испытаний текущего контроля успеваемости (в течение семестра)	28		28	
подготовка ко всем видам контрольных испытаний промежуточной аттестации (по окончании семестра)	28		28	
Всего (часы):	**144**		**144**	
Всего (зачетные единицы):	**4**		**4**	

4. Содержание дисциплины, структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

4.1. Разделы дисциплины и трудоемкость по видам учебных занятий (в академических часах)

№ п/п	**Наименование раздела /темы дисциплины**	**Общая трудоём-кость всего** *(в часах)*	**Виды учебных занятий, включая самостоятельную работу обучающихся и трудоемкость** *(в часах)*				**Формы текущего контроля успевае-мости**
			Аудиторные учебные занятия			**СРО**	
			Лек	**Сем/Пр**	**Лаб**		
1.			16	16	-	112	
1.1.	Введение	8	2	2		12	
1.2.	Работа с локальным репозиторием	8	2	2	-	12	
1.3.	Работа с удаленным репозиторием	8	2	2	-	12	
1.4.	Работа с ветками	8	2	2	-	12	
1.5.	Работа с историей	8	2	2		12	

1.6.	Перебазирование и слияние	8	2	2	-	12	
1.7.	Работа с чужими репозиториями	8	2	2	-	12	
1.8.	Заключение	6	1	1	-	12	
1.9.	Итоговый тест	6	1	1	-	16	

4.2. Содержание дисциплины, структурированное по разделам (темам)

Лекционный курс

№	Наименование раздела /темы дисциплины	Содержание
1.1.	Введение	Что такое Git
1.2.	Работа с локальным репозиторием	Работа с локальными репозиториями А также узнаем новые команды: git init git status git add git commit git log git show
1.3.	Работа с удаленным репозиторием	Работа с удаленными репозиториями А также узнаем новые команды: git remote git push git pull
1.4.	Работа с ветками	Что такое branch git checkout git diff
1.5.	Работа с историей	Как работать с историей Из-за чего возникают конфликты git show git diff
1.6.	Перебазирование и слияние	перебазирование и слияние. Чем похожи и в чем различие
1.7.	Работа с чужими репозиториями	как работать с чужими репозиториями и делать pull-requests
1.8.	Заключение	полезные ссылки, ресурсы,подведение итогов
1.9	Итоговый тест	Итоговый тест

Практические/семинарские занятия

№	Наименование раздела /темы дисциплины	Содержание
1.1.	Введение	Что такое Git

1.2.	Работа с локальным репозиторием	Работа с локальными репозиториями А также узнаем новые команды: git init git status git add git commit git log git show
1.3.	Работа с удаленным репозиторием	Работа с удаленными репозиториями А также узнаем новые команды: git remote git push git pull
1.4.	Работа с ветками	Что такое branch git checkout git diff
1.5.	Работа с историей	Как работать с историей Из-за чего возникают конфликты git show git diff
1.6.	Перебазирование и слияние	перебазирование и слияние. Чем похожи и в чем различие
1.7.	Работа с чужими репозиториями	как работать с чужими репозиториями и делать pull-requests
1.8.	Заключение	полезные ссылки, ресурсы,подведение итогов
1.9	Итоговый тест	Итоговый тест

Лабораторные занятия

Не предусмотрены.

5. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине

В качестве учебно-методических материалов используется рекомендованная литература.

6. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине

6.1. Паспорт фонда оценочных средств по дисциплине

№ п/п	Контролируемые разделы (темы) дисциплины (результаты по разделам)	Код контролируемой компетенции (или её части) / и ее формулировка	Наименование оценочного средства

1.3	Работа с удаленным репозиторием	ОПК-2:	Контрольная работа № 1
2.1.	Работа с чужими репозиториями	ОПК-1:	Контрольная работа № 2

6.2. Типовые контрольные задания или иные материалы

6.2.1. Зачет с оценкой

В билете два теоретических вопроса и один практический

Теоретические вопросы билета:

1. Как создать новый локальный репозиторий и подключить его к удаленному?
2. Чем отличается команда git init от команды git clone?
3. Как добавить изменения в индекс и выполнить коммит в локальном репозитории?
4. Как отменить изменения, которые ещё не были закоммичены, и как удалить файл из репозитория, но оставить его на диске?
5. Как просмотреть статус локального репозитория и историю коммитов с помощью команд git status и git log?
6. Чем команда git fetch отличается от git pull и когда следует использовать каждую?
7. Как синхронизировать локальный репозиторий с удалённым и отправить изменения с помощью git push?
8. Как создать, переключиться и удалить локальные ветки, а также просматривать список всех веток?
9. Как объединить изменения из одной ветки в другую с помощью git merge, и чем эта команда отличается от git rebase?
10. Как разрешать конфликты при слиянии веток или перебазировании, и в чем заключается разница между этими процессами?
11. Как отменить последний коммит, сохранив изменения в рабочем каталоге, с помощью команды git reset?
12. Что такое теги в Git и как их использовать для маркировки важных коммитов или версий?
13. Как использовать команду git rebase для упорядочивания и чистки истории коммитов, и как избежать потери данных при этом?
14. Как клонировать чужой репозиторий и создать pull request для отправки изменений?
15. Как подключить чужой репозиторий как удалённый в своём локальном репозитории и синхронизировать изменения?
16. Как выполнить форк чужого репозитория на GitHub и потом внести изменения в свой форк?
17. Как просматривать и сравнивать изменения в чужом репозитории с помощью git diff?
18. Как настроить несколько удалённых репозиториев в одном проекте и работать с ними?
19. Как вернуться к предыдущей версии репозитория с помощью git checkout <commit> или git reset?
20. Как использовать команды git branch, git checkout, и git merge для эффективного управления ветвлением в командной разработке?

Критерий оценки – правильность и полнота ответа на вопросы. Оценка выставляется по шкале от 0 до 40 баллов: теоретические вопросы –30 баллов, 10 баллов– дополнительные вопросы. Зачет считается сданным при оценке не ниже 25 баллов.

6.2.2. Контрольная работа № 1

Создайте новый локальный репозиторий.

Создайте и переключитесь на новую ветку с именем feature-branch.

Добавьте файл с любым содержимым (например, file.txt) и выполните коммит.

Переключитесь обратно на основную ветку (master или main) и создайте новый файл (например, readme.md), затем выполните коммит.

Слияние веток:

Переключитесь обратно на ветку feature-branch и внесите изменения в file.txt.

Выполните коммит этих изменений.

Вернитесь на основную ветку и выполните слияние (merge) с веткой feature-branch.

Разрешите любые возможные конфликты, если они возникнут.

Работа с удалённым репозиторием:

Создайте репозиторий на GitHub (или другой платформе).

Подключите его как удалённый репозиторий к своему локальному.

Отправьте локальные изменения (push) в удалённый репозиторий.

Создайте pull request на GitHub для предложенной вами ветки и объясните, какие изменения были внесены.

История и ревизия:

Используйте команду git log для просмотра истории коммитов.

Отмените последний коммит с помощью команды git reset (оставив изменения в рабочем каталоге) и выполните новый коммит с другим сообщением.

Работа с тегами:

Создайте тег для последнего коммита с именем v1.0.

Просмотрите все теги с помощью git tag.

Задание на оценку:

Проверьте, чтобы все шаги были выполнены, и продемонстрируйте использование каждой команды через Git и GitHub.

6.2.2. Контрольная работа № 2

Дана маленькая библиотека, вычисляющая некоторые тригонометрические функции.

Инициализируйте репозиторий в папке mat_lib

2) Задайте имя пользователя и email глобально
   1. Проверьте, что изменения внесены в файл глобальных настроек .gitconfig
   2. Проверьте, что файл локальных настроек .git/config не был изменен
3) Изучите содержимое папки .git/
   1. Узнайте, на что сейчас указывает HEAD
   2. Просмотрите файл, на который указывает HEAD (в этом пункте есть подвох)
4) Добавьте все файлы в индекс
5) Сделайте первый коммит
   1. Просмотрите объект коммита, найдите хэш объекта-дерева корня репозитория
   2. Просмотрите объект дерева корня репозитория
   3. Проверьте, на что указывает HEAD сейчас
   4. Просмотрите файл, на который указывает HEAD

5. Ответьте на вопрос: на что указывает текущая ветка? Для этого просмотрите на объект, на который указывает ветка.

6) Выполните файл test.py

1. Просмотрите статус файлов, чтобы обнаружить, что появились файлы кэша

7) Сделайте второй коммит

1. Просмотрите объект коммита, найдите хэш родительского коммита
2. Посмотрите, на что сейчас указывает HEAD
3. Проверьте файл, на который указывает HEAD
4. Узнайте, на что указывает текущая ветка. Для этого просмотрите на объект, на который указывает ветка.

б) критерии оценивания компетенций (результатов) – правильная работа кода программы, понимание алгоритма метода оптимизации, умение вывести необходимые для алгоритма формулы.

в) описание шкалы оценивания:

Каждая задача оценивается по шкале от 0 до 10 баллов.

Контрольная работа считается выполненной успешно при суммарной оценке не ниже 18 баллов.

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

Форма аттестации	Наименование оценочного средства	Баллы
Зачет (100 баллов)	Контрольная работа № 1	30
	Контрольная работа № 2	30
	Ответы на билет	40

7. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины

а) основная учебная литература:

1)Scott Chacon, Ben Straub – *Pro Git (Second Edition)* – 2014 – 488 страниц.
2)https://git-scm.com/doc

б) дополнительная учебная литература:

-

8. Перечень ресурсов* информационно-телекоммуникационной сети «Интернет» (далее - сеть «Интернет»), необходимых для освоения дисциплины

https://git-scm.com/doc

9. Методические указания для обучающихся по освоению дисциплины

Вид учебного занятия	Организация деятельности студента
Лекция	Написание конспекта лекций: кратко, схематично, последовательно фиксировать основные положения, выводы, формулировки, обобщения; помечать важные мысли, выделять ключевые слова, термины. Проверка

	терминов, понятий с помощью энциклопедий, словарей, справочников с выписыванием толкований в тетрадь. Обозначить вопросы, термины, материал, который вызывает трудности, пометить и попытаться найти ответ в рекомендуемой литературе. Если самостоятельно не удается разобраться в материале, необходимо сформулировать вопрос и задать преподавателю на консультации, на практическом занятии.
Практические занятия	Проработка рабочей программы, уделяя особое внимание целям и задачам, структуре и содержанию дисциплины. Работа с конспектом лекций, просмотр рекомендуемой литературы. Изучение выбранной предметной области на примерах решения задач семинарских занятий, индивидуальных домашних заданий.
Курсовая работа	Не предусмотрена
Контрольная работа	Ознакомиться с основной и дополнительной литературой, включая справочные издания, зарубежные источники, основополагающие термины. Попрактиковаться в решении аналогичных домашних задач по всем темам контрольных работ.
Лабораторная работа	Не предусмотрена.
Подготовка к зачету	При подготовке к зачету необходимо ориентироваться на конспекты лекций и рекомендуемую литературу.

10. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень программного обеспечения и информационных справочных систем (при необходимости)

Издательская система LaTeX для подготовки докладов, презентаций и учебного материала.

11. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине

Видеопроектор, компьютер, издательская система LaTeX для подготовки докладов, презентаций и учебного материала.

12. Иные сведения и (или) материалы

12.1. Перечень образовательных технологий, используемых при осуществлении образовательного процесса по дисциплине

Часов в интерактивной форме – 8.

В ходе практических занятий происходит публичное обсуждение каждой решаемой задачи. При этом студенты высказывают свои мнения по выбору наиболее простого способа поиска оптимального решения.

После решения домашних работ на консультациях проводится разбор допущенных студентами ошибок.

12.2. Формы организации самостоятельной работы обучающихся (темы, выносимые для самостоятельного изучения; вопросы для самоконтроля; типовые задания для самопроверки

Некоторые темы изучаются студентами самостоятельно. Для изучения используется приведённая в списке основная и дополнительная литература. Контроль освоения материала осуществляется при проверке контрольных работ, домашнего задания и на зачете.

№	Тема и часть, изучаемая (осваиваемая) самостоятельно
1.1	Основы систем контроля версий (СКВ)
1.2	GitHub и другие платформы для хостинга репозиториев
1.3	Основы командной разработки
1.4	Алгоритмы слияния в Git
1.5	CI/CD и автоматизация сборки
1.6	Продвинутые функции Git
1.7	Контейнеризация с Docker
1.1	Основы систем контроля версий (СКВ)
1.2	GitHub и другие платформы для хостинга репозиториев
1.3	Основы командной разработки

Вопросы и задания для самоконтроля по всем темам:

1. Как создать новый локальный репозиторий в Git?
2. Как добавить изменения в индекс (staging area) и выполнить коммит?
3. Как отменить изменения, которые ещё не были закоммичены?
4. Как подключить удалённый репозиторий к локальному с помощью команды git remote add?
5. Как клонировать удалённый репозиторий с помощью git clone?
6. Чем команда git fetch отличается от команды git pull?
7. Как создать, переключиться и удалить ветки в Git?
8. Как объединить изменения из одной ветки в другую с помощью git merge?
9. Чем отличается команда git merge от git rebase?
10. Как просмотреть историю коммитов в Git с помощью git log?
11. Чем отличается git reset от git revert?
12. Как отменить изменения в нескольких коммитах с помощью git reset?
13. Когда следует использовать git rebase, а когда git merge?
14. Как выполнить перебазирование ветки с помощью git rebase?
15. Как разрешать конфликты при слиянии веток?
16. Как форкнуть чужой репозиторий и клонировать его себе?
17. Как подключить чужой репозиторий как удалённый в вашем локальном репозитории?
18. Как отправить изменения в чужой репозиторий через pull request?

12.3. Краткий терминологический словарь

Fork	это копия репозитория, которая создается в вашем аккаунте на GitHub. Форк позволяет вам свободно вносить изменения в проект без того, чтобы затронуть оригинальный репозиторий. Это полезно, когда вы хотите внести улучшения или исправления в чужой проект, не имея прав на прямое изменение его содержимого. После того как вы сделаете изменения в форке, вы можете отправить **pull request** в оригинальный репозиторий, чтобы предложить свои изменения владельцу репозитория.

Pull Request	это запрос на слияние изменений из одной ветки в другую, обычно из вашей форкнутой копии в основной репозиторий. Это основная форма взаимодействия в открытых проектах. Через PR вы можете предложить свои изменения, которые должны быть рассмотрены и, возможно, слиты в основной репозиторий. В pull request можно прикреплять комментарии, обсуждать код, а также проводить ревью перед слиянием изменений.
Branch	это отдельная линия разработки в репозитории. В Git ветки позволяют работать над новыми функциями или исправлениями, не влияя на основную (master или main) ветку. Ветки позволяют изолировать изменения и сливать их обратно в основную ветку (или другую ветку) после того, как работа завершена. В GitHub ветки используются для упрощения процесса разработки, позволяя нескольким разработчикам работать параллельно, не мешая друг другу.данных

1.