

МИНОБРНАУКИ РОССИИ
Ярославский государственный университет им. П.Г. Демидова

Кафедра дифференциальных уравнений

УТВЕРЖДАЮ

Декан математического факультета

Нестеров П.Н.

20 мая 2025 г.

Рабочая программа дисциплины

Системы контроля версий

Направление подготовки (специальности)
01.03.02 Прикладная математика и информатика

Направленность (профиль)
«Прикладное программирование и информационные технологии»

Форма обучения очная

Программа рассмотрена
на заседании кафедры
от 18.04.2025, протокол № 8

Программа одобрена НМК
математического факультета
протокол № 9 от 05.05.2025

1. Цели освоения дисциплины

Целями освоения данной дисциплины является формирование общекультурных (универсальных) и профессиональных компетенций в области информационных технологий в соответствии с требованиями ФГОС ВПО по данному направлению подготовки. Целью предлагаемого курса является обучение студентов современным подходам к разработке ПО, принятым во многих компаниях и сообществах разработчиков.

Особое внимание уделяется таким аспектам, как совместная работа в команде и средства автоматического регулярного тестирования.

2. Место дисциплины в структуре образовательной программы

Данная дисциплина относится к части образовательной программы, формируемой участниками образовательных отношений, и является элективной дисциплиной. Для изучения дисциплины необходимы компетенции, сформированные у обучающихся в результате изучения дисциплин «Основы программирования», «Практикум по информатике». Данная дисциплина является продолжением практики по информатике. Данная дисциплина составлена с учетом актуальности проблем, связанных с командной разработкой в крупных it-компаниях. Курс рассматривается в контексте будущей профессиональной деятельности студентов, как необходимое средство для комплексов программ. Сформированные в процессе изучения дисциплины компетенции, необходимы студенту при изучении последующих программистских дисциплин и в процессе прохождения практики и написании ВКР.

3. Планируемые результаты обучения по дисциплине, соотнесенные с планируемыми результатами освоения образовательной программы

Процесс изучения дисциплины направлен на формирование следующих элементов компетенций в соответствии с ФГОС ВО, ОП ВО и приобретения следующих знаний, умений, навыков и (или) опыта деятельности:

| Формируемая компетенция (код и формулировка) | Индикатор достижения компетенции (код и формулировка) | Перечень планируемых результатов обучения |
|--|---|--|
| Профессиональные компетенции | | |
| ПК-1 Способен собирать, обрабатывать и интерпретировать данные современных научных исследований, необходимые для формирования выводов по соответствующим научным исследованиям | И-ПК-1.1 Обладает способностью критического отбора данных, связанных с научными исследованиями | Знать: - архитектуру построения разработки приложений; - основные виды и способы проектирования в командной разработке |
| | И-ПК-1.2 Имеет навыки обработки данных с применением современных информационных технологий и алгоритмов | Уметь: - администрировать созданный репозиторий. Владеть: - навыками и приемами работы с репозиториями в командной разработке |
| | И-ПК-1.3 Способен самостоятельно интерпретировать данные научных исследований и | Уметь: - самостоятельно тестировать готовый продукт - формулировать выводы, используя современную научную литературу, и |

| | | |
|---|---|---|
| | формулировать соответствующие выводы | применять их при решении прикладных задач |
| ПК-3 Способен к разработке и применению алгоритмических и программных решений в области системного и прикладного программного обеспечения | И-ПК-3.1 Обладает устойчивыми знаниями в области разработки алгоритмов и программирования | Знать: - основы языка Python; - основные инструменты для различных видов тестирования; - стандарты разработки программного обеспечения. Уметь: - разрабатывать приложения в одной из архитектур систем контроля версий |

4. Объем, структура и содержание дисциплины

Общая трудоемкость дисциплины составляет **2** зачетных единиц, **72** акад. часа.

| № п/п | Темы (разделы) дисциплины, их содержание | Семестр | Виды учебных занятий, включая самостоятельную работу студентов, и их трудоемкость (в академических часах) | | | | | | Формы текущего контроля успеваемости Форма промежуточной аттестации (по семестрам) |
|----------|--|---------|---|--------------|--------------|--------------|-----------------------------|---------------------------|--|
| | | | Контактная работа | | | | | самостоятельная работа | |
| | | | лекции | практические | лабораторные | консультации | аттестационные испытания | | |
| 1 | Вводная лекция. Отличия в процессах разработки учебных приложений и разработки приложений в большой команде разработчиков. Вводный курс языка Python. Системы контроля версий (version control systems). Обзор программ Subversion, git. Примеры работы с публичными репозиториями code.google.com, github.com. Понятия об оформлении кода на примере Google Code Style Guide. | 5 | 3 | 5 | | 1 | | 8 | Фронтальный опрос |
| 2 | Совместная работа с общим репозиторием. Понятие code review. Использование сервиса rietveld (codereview.appspot.com). Утилиты diff, patch | 5 | 3 | 5 | | 1 | | 8 | Фронтальный опрос |
| 3 | Системы учёта ошибок (issue tracking) на примере code.google.com/. Использование средств ведения журнала сообщений (logging) и средств assert для упрощения процесса поиска ошибок. | 5 | 5 | 3 | | 1 | | 8 | Фронтальный опрос |

| | | | | | | | | | |
|---|---|---|----|----|--|---|-----|------|---------------------|
| 4 | Средства автоматического и модульного тестирования на примере Python. Разработка проектов с открытым исходным кодом (open source). Различные типы лицензий. Подходы, применяемые в проектах с открытым исходным кодом для работы со сторонними разработчиками. Практическое использование СКВ | 5 | 5 | 3 | | 1 | | 8 | Лабораторная работа |
| | | | | | | | 0,3 | 3,7 | зачет |
| | Всего | | 16 | 16 | | 4 | 0,3 | 35,7 | |

5. Образовательные технологии, в том числе технологии электронного обучения и дистанционные образовательные технологии, используемые при осуществлении образовательного процесса по дисциплине

В процессе обучения используются следующие образовательные технологии:

Вводная лекция – дает первое целостное представление о дисциплине и ориентирует студента в системе изучения данной дисциплины. Студенты знакомятся с назначением и задачами курса, его ролью и местом в системе учебных дисциплин и в системе подготовки в целом. Дается краткий обзор курса, история развития науки и практики, достижения в этой сфере, имена известных ученых, излагаются перспективные направления исследований. На этой лекции высказываются методические и организационные особенности работы в рамках данной дисциплины, а также дается анализ рекомендуемой учебно-методической литературы.

Академическая лекция с элементами лекции-беседы – последовательное изложение материала, осуществляемое преимущественно в виде монолога преподавателя. Элементы лекции-беседы обеспечивают контакт преподавателя с аудиторией, что позволяет привлекать внимание студентов к наиболее важным темам дисциплины, активно вовлекать их в учебный процесс, контролировать темп изложения учебного материала в зависимости от уровня его восприятия.

Практическое занятие – занятие, посвященное освоению конкретных умений и навыков по закреплению полученных на лекции знаний.

Консультации – вид учебных занятий, являющийся одной из форм контроля самостоятельной работы студентов. На консультациях по просьбе студентов рассматриваются наиболее сложные моменты при освоении материала дисциплины, преподаватель отвечает на вопросы студентов, которые возникают у них в процессе самостоятельной работы.

6. Перечень лицензионного и (или) свободно распространяемого программного обеспечения, используемого при осуществлении образовательного процесса по дисциплине

В процессе осуществления образовательного процесса по дисциплине используются:
для формирования материалов для текущего контроля успеваемости и проведения промежуточной аттестации, для формирования методических материалов по дисциплине:

- программы Microsoft Office;
- издательская система LaTeX;
- Adobe Acrobat Reader;
- MikTeX (свободно распространяемое ПО).

7. Перечень современных профессиональных баз данных и информационных справочных систем, используемых при осуществлении образовательного процесса по дисциплине (при необходимости)

В процессе осуществления образовательного процесса по дисциплине используются:

- Автоматизированная библиотечно-информационная система «БУКИ-NEXT»
http://www.lib.uniyar.ac.ru/opac/bk_cat_find.php
- Электронно-библиотечная система «Юрайт» <https://urait.ru>
- Электронно-библиотечная система «Лань» <http://e.lanbook.com/>
- Электронно-библиотечная система «Консультант Студента»:
<https://www.studentlibrary.ru/>

8. Перечень основной и дополнительной учебной литературы, ресурсов информационно-телекоммуникационной сети «Интернет» (при необходимости), рекомендуемых для освоения дисциплины

а) основная литература:

1. Мартин Р. Идеальный программист: как стать профессионалом разработки ПО. / Р. Мартин; [перевел с англ. Е. Матвеев] - СПб.: Питер, 2021. - 214 с.: ил.
2. М. А. Васильева, К. М. Филипченко Система контроля версий. Основы командной разработки — Санкт-Петербург: Лань, 2022 <https://reader.lanbook.com/book/261089>

б) дополнительная литература:

1. Доусон М. Програмируем на Python. / М. Доусон; [пер. с англ. В. Порицкого] - СПб.: Питер, 2016. - 414 с.
2. Северанс, Ч. Р. Python для всех / Ч. Р. Северанс; пер. с англ. А. В. Снастина. - Москва : ДМК Пресс, 2021. - 262 с. - ISBN 978-5-93700-104-7. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : <https://www.studentlibrary.ru/book/ISBN9785937001047.html>
3. С. В. Борзунов, С. Д. Кургалин Языки программирования. Python: решение сложных задач — Санкт-Петербург: Лань, 2023 <https://reader.lanbook.com/book/319394>
4. Федоров Д. Ю. Программирование на языке высокого уровня Python: учебное пособие для вузов — Москва: Издательство Юрайт, 2023.
<https://urait.ru/viewer/programirovanie-na-yazyke-vysokogo-urovnya-python-515076>
5. Самарский А. А. Задачи и упражнения по численным методам. / Самарский А.А., Вабищевич П.Н., Самарская Е.А. - М.: Эдиториал УРСС, 2000. - 208с.
6. Никитина, Т. П. Программирование. Основы Python для инженеров / Т. П. Никитина, Л. В. Королев. — Санкт-Петербург : Лань, 2023. — 156 с. — ISBN 978-5-507-45284-2. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/302720>

в) ресурсы сети «Интернет»

1. <http://svnbook.red-bean.com>
2. http://ptgmedia.pearsoncmg.com/images/0131855182/downloads/Nagel_book.pdf
3. <https://git-scm.com/book/en/v2>
4. <http://www-cs-students.stanford.edu/~blynn/gitmagic/index.html>
5. <https://vimeo.com/28610688>
6. <https://www.openstack.org>

9. Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине

Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине включает в свой состав специальные помещения:

- учебные аудитории для проведения практических занятий (семинаров);
- учебные аудитории для проведения групповых и индивидуальных консультаций;
- учебные аудитории для проведения текущего контроля и промежуточной аттестации;
- помещения для самостоятельной работы;
- помещения для хранения и профилактического обслуживания технических средств обучения.

Помещения для самостоятельной работы обучающихся оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа к электронной информационно-образовательной среде ЯрГУ.

Автор(ы):

Доцент кафедры дифференциальных уравнений

А.Б. Демченко

**Приложение №1 к рабочей программе дисциплины
«Системы контроля версий»**

**Фонд оценочных средств
для проведения текущей и промежуточной аттестации студентов
по дисциплине**

**1. Типовые контрольные задания или иные материалы, используемые в процессе
текущей аттестации**

Примеры вопросов для фронтального опроса по теме 1

1. Приведите определения системы контроля версий (СКВ).
2. Что такое контроль версий, и зачем он нужен?
3. Что такое CVS (Concurrent Versions System, Система совместных версий)?
4. Преимущества и недостатки CVS.

Примеры вопросов для фронтального опроса по теме 2

1. Что такое code review?
2. Из чего состоит review?
3. Как проводить design review?
4. Как проводить code review?
5. Утилиты для review.

Примеры вопросов для фронтального опроса по теме 3

1. Что такое assert?
2. Какие виды assert'ов бывают?
3. Что такое целочисленное переполнение?
4. Что такое переполнение стека?
5. Когда и где стоит использовать assert'ы? Когда можно обойтись без assert'ов?
6. Когда нельзя использовать assert'ы?

Лабораторная работа

Цель работы:

1. Изучить на практике понятия и компоненты систем контроля версий (СКВ), приемы работы с ними. 2. Освоить специализированное ПО и распространенный сервис для работы с распределенной СКВ Git — TortoiseGit и GitHub.com.

Общие указания к выполнению лабораторной работы

1. Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала (в *nix) или из специальной консольной оболочки Git Bash (в Windows). Однако, лабораторная работа ориентирована на применение графической надстройки TortoiseGit (аналог в Linux — RabbitVCS). TortoiseGit работает не как отдельная программа, а встраивается в контекстные меню «Проводника» Windows. Вместе с Git для Windows поставляется также программа gitk (Git GUI) — она гораздо менее популярна и пользоваться ей для ЛР не следует.
2. Задания необходимо читать внимательно и полностью. Благодаря тому, что Git является распределенной СКВ, резервную копию локального хранилища можно сделать простым копированием или архивацией.
3. В отчет можно включать снимки экрана, сообщения и комментарии по своему усмотрению с тем, чтобы было удобно пояснять сделанное, опираясь на отчет.

Задание на лабораторную работу

1. Отработать навыки использования хранилища на локальной машине.

1.1. Настроить Git, указав имя и e-mail разработчика для подписи commit-ов.

Указание. Диалог настроек вызывается пунктом TortoiseGit → Settings контекстного меню любого каталога, нужная вкладка называется «Git». Задавать следует глобальные настройки (для всех хранилищ), установив переключатель «Global».

1.2. Создать хранилище для учебного проекта.

1.3. Совершить несколько commit-ов.

1.3.1. Скопировать sdt.h в каталог хранилища и создать файл main.cpp со включением sdt.h и пустой функцией main().

1.3.2. Добавить в программу ввод двух целых чисел с приглашением.

Указание № 1. После выполнения каждого подпункта необходимо убеждаться, что программа работает, и совершать commit изменений.

Указание № 2. Следите за тем, какие файлы отмечены в списке для commita изменений в них, — кроме main.cpp и, иногда, sdt.h, больше никаких других не нужно.

1.4. Предотвратить автоматическое добавление в хранилище файлов, не нуждающихся в контроле версий, — *.o и *.exe. Правило об игнорировании следует помещать в файл .gitignore в корневом каталоге хранилища («.gitignore in repository root»). Этот файл также попадает под контроль версий, поэтому после создания правил требуется совершить commit изменений в файле .gitignore.

1.5. Добавить в программу вывод суммы введенных чисел и совершить commit.

1.6. Просмотреть историю (журнал) хранилища.

1.7. Просмотреть разность (diff) между пунктами истории 1.3.2 и 1.5.

2. Освоить передачу истории хранилища по сети.

2.1. Организовать общее хранилище на удаленном сервере.

2.1.1. Зарегистрироваться на GitHub.

2.1.2. Создать пустое удаленное хранилище с любым наименованием.

Указание. Вопреки инструкции на GitHub, добавлять в хранилище файл README.md не нужно, удаленное хранилище должно быть пустым.

2.1.3. Разрешить пользователям-преподавателям совершать commit-ы. Требуется на странице хранилища выбрать «Settings» (справа), далее «Collaborators», где ввести имена пользователей, которым будет предоставлен полный доступ к хранилищу (эти имена можно узнать у лаборантов).

2.1.4. Настроить локальное хранилище для синхронизации с удаленным. Необходимо в контекстном меню каталога локального хранилища выбрать TortoiseGit → Settings, где перейти к пункту Remote . Достаточно ввести условное имя удаленного хранилища «origin» и его адрес, который отображается на web-странице удаленного хранилища в разделе «Quick setup» (вариант HTTP). От загрузки сведений о ветвлениях в удаленном хранилище отказаться.

2.2. Передать локальное хранилище на удаленный сервер (push).

Замечание. Здесь и далее при взаимодействии с удаленным сервером потребуется вводить имя пользователя и пароль, с которыми выполнялась регистрация на GitHub.

2.3. Перейти к странице хранилища на GitHub (обновить её) и ознакомиться с возможностями просмотра содержимого через web-интерфейс.

2.4. Загрузить копию удаленного хранилища на локальную машину (clone).

Замечание. Целью является имитация совместной работы с удаленным хранилищем. Для этого на одной машине организуются 2 локальных хранилища: созданное в пункте 1.1 (RepoA) и загруженное с удаленного сервера (RepoB).

Указание. Диалог «Git clone» следует вызывать из контекстного меню каталога вне локального хранилища. В качестве URL потребуется указать адрес удаленного хранилища, а в качестве Directory — имя каталога для нового локального хранилища.

2.5. Сымитировать параллельную работу над проектом.

2.5.1. В локальном хранилище RepoB добавить в программу печать разности введенных чисел, сделать commit и передать изменения на сервер.

2.5.2. В локальном хранилище RepoA добавить над функцией main() комментарий о том, что программа является учебной, сделать commit, но не отправлять изменений на сервер.

2.6. На странице хранилища на GitHub перейти в раздел Commits и ознакомиться с возможностью просмотра истории изменений через web-интерфейс.

2.7. В локальном хранилище RepoA выполнить загрузку с сервера новейших ветвлений и изменений (fetch) и просмотреть журнал хранилища.

Указание. По умолчанию показывается только текущая активная ветвь (по умолчанию — master). Просмотреть все commit-ы во всех ветвях, в том числе в загруженной из удаленного хранилища ветви origin/master, нужно включить флажок «All branches» слева снизу окна журнала.

2.8. Совместить изменения в локальном хранилище с загруженными.

2.8.1. Использовать действие pull для загрузки изменений с удаленного сервера и автоматического совмещения их с имеющимися локально. Просмотреть журнал изменений (или обновить кнопкой Refresh)

Примечание. Фактически, при обновлении производится слияние ветвей master и origin/master — то есть, двух версий истории, существовавших удаленно и локально. При этом история стала нелинейной и появился лишний commit слияния. Иногда такое усложнение имеет смысл, но в данном случае было бы желательно сохранить историю линейной и просто перенести локальные наработки вслед за новейшими

2.8.2. Отменить неудобный результат действия pull.

Указание. В журнале изменений в контекстном меню commit-a, где был добавлен комментарий (то есть, последнего перед слиянием), выбрать «Reset master to this...» и указать тип отмены «Hard».

Указание. Журнал изменений не всегда обновляется автоматически, используйте кнопку Refresh, если изменения не появились сразу.

2.8.3. Выполнить перенос (rebase) локальных изменений на основу новейшего загруженного состояния проекта.

Указание. В журнале изменений в контекстном меню пункта, на котором находится конец ветви origin/master (прямоугольник в TortoiseGit), следует выбрать пункт «Rebase "master" onto this...» и далее нажать кнопку «Start rebase».

2.9. Передать итоговое состояние локального хранилища RepoA на удаленный сервер, используя команду push.

2.10. Действуя аналогично п. п. 2.7 и 2.8.3, синхронизировать с удаленным локальное хранилище RepoB (в нем не хватает commit-а с комментарием).

Замечание. На данном этапе во всех трех хранилищах (локальных RepoA и RepoB и удаленном на GitHub) должна быть одинаковая линейная история из пяти — шести commit-ов.

3. Изучить действия, связанные с ветвлениями и разрешением конфликтов.

Замечание. Все действия выполняются в одном локальном хранилище, например, в RepoA.

3.1. Добавить в программу печать произведения чисел и совершите commit.

На данном этапе программа может быть такой:

```
1 #include "sdt.h"
2
3 // This program is just an example one under VCS.
4 int main()
5 {
6     int a, b;
7     cout << "Enter A and B: ";
8     cin >> a >> b;
9     cout << "A + B = " << a + b << '\n'
10    << "A - B = " << a - b << '\n'
11    << "A * B = " << a * b << '\n';
12 }
```

3.2. Создать новую ветвь (branch) под названием division. из пункта истории, в котором был добавлен комментарий над main().

3.3. В новой ветви повторить пункт 3.1, заменив умножение делением.

Указание. Переключиться на ветвь можно, выбрав в контекстном меню commit-а, которым эта ветвь оканчивается, пункт «Switch/checkout to this». При создании ветви можно сразу установить флажок «Switch to new branch». Переключаться можно только при чистом (clean) хранилище, то есть, без изменений в рабочей копии.

3.4. Переключиться обратно на ветвь master.

3.5. Выполнить слияние ветви division в ветвь master так, чтобы в последней оказался код для печати и произведения, и частного.

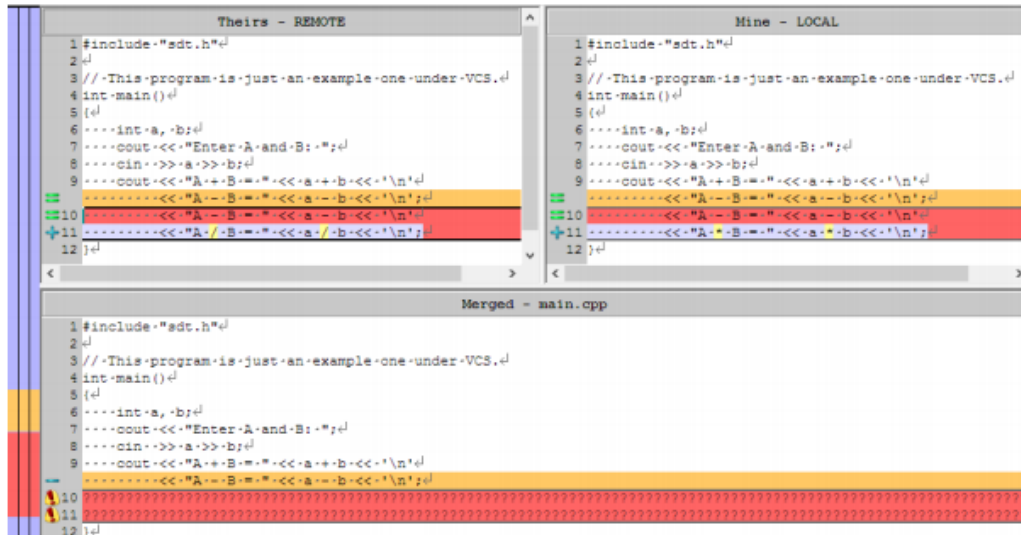
3.5.1. В журнале изменений в контекстном меню пункта-окончания ветви division выбрать пункт «Merge into "master"...» и начать слияние, не меняя настроек.

Действие завершится ошибкой из-за конфликта (conflict): в файле main.cpp строка 10 изменена в обоих commit-ах одинаково, а строка 11 — по-разному, и СКВ не может автоматически выбрать «правильный» вариант. Требуется вручную указать, какие строки должны войти в итоговую версию файла.

3.5.2. Приступить к разрешению конфликта.

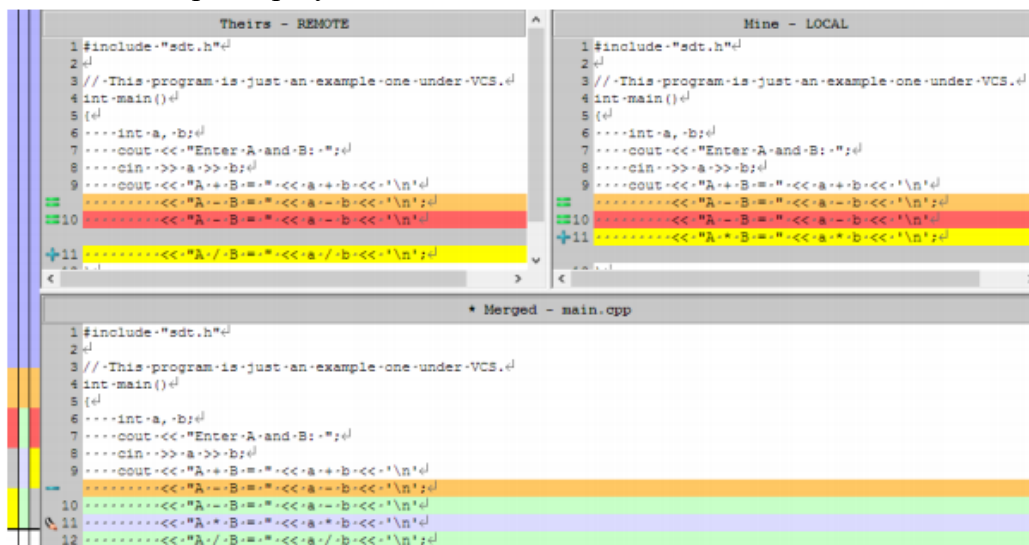
Указание. Следует нажать кнопку Resolve (снизу), а затем выбрав пункт Edit conflicts из контекстного меню main.cpp.

Примечание. Редактор конфликтов похож на программу для просмотра разностей между файлами: слева показывается файл в ветви, откуда делается слияние (division), справа — ветви, куда делается слияние (master), снизу — результат слияния. Знаками равенства в соседних верхних полях отмечаются не только строки, оставшиеся неизменными, но и строки с одинаковыми изменениями: здесь, в строке 10 убрана точка с запятой в конце в обеих ветвях. В нижнем поле каждый восклицательный знак обозначает отдельный конфликт. Примерный вид окна редактирования конфликтов представлен на рисунке ниже.



3.5.3. Разрешить конфликты: — в качестве строки 10 предпочесть строку 10 любой ветви; — на место строки 11 вставить 2 строки: строку с печатью произведения и строку с печатью частного; — лишнюю точку с запятой на строке 11 поля-результата удалить.

Указание. Переносить строки из той или иной версии в итоговую можно, выбирая конфликтные блоки в левом или правом верхнем поле (простым щелчком левой кнопкой мыши) и пользуясь контекстным меню. Например, «Use this text block» переносит выбранный блок в поле-результат; пункт «Use text block from 'mine' before 'theirs'» переносит в результат 2 строки: сначала из правого блока, потом из левого (одну под другой). На рисунке ниже показан возможный верный результат.



3.5.4. Завершить процедуру разрешения конфликтов.

Указание. Следует нажать на кнопку «Mark as resolved», чтобы отметить файл как избавленный от конфликтов, и закрыть программу для их разрешения.

3.5.5. Завершить слияние ветви division в ветвь master, написав осмысленный комментарий к слиянию и совершив commit.

3.5.6. Убедиться, что программа компилируется и верно работает. Если это не так, исправить все ошибки и добиться правильной работы. Совершить commit.

Замечание-указание. Ситуация, когда после слияния программа все-таки оказывается не вполне корректной, случается на практике довольно часто. В этом случае commit, созданный

при слиянии, оказывается логически неправильным, он не имеет ценности без последующего исправления. В Git имеется возможность изменить (amend) уже совершенный commit, пока он не передан на сервер. Это делается при следующем commit-исправлении: следует установить флажок «Amend Last Commit» в диалоге commit — нового commit не появится, а вместо этого изменения будут приписаны предыдущему пункту истории. Можно воспользоваться данной возможностью при выполнении пункта.

3.5.7. Передать все изменения всех ветвей в удаленное хранилище. Указание. По умолчанию передаются только изменения текущей ветви, для передачи изменений всех ветвей следует отметить флажок «Push all branches» диалога push.

Замечание. В данном задании отрабатывается навык слияния ветвей, существующих только в локальном хранилище и вступивших в конфликт с ведома единственного автора. Постоянно возникают и ситуации, когда одну и ту же ветвь, но в локальном и удаленном хранилище независимо изменяют разные авторы. В этом случае действия push и rebase приведут к конфликтам. Их разрешение выполняется совершенно аналогично.

2. Список вопросов и (или) заданий для проведения промежуточной аттестации

1. Что такое системы контроля версий (СКВ) и для решения каких задач они предназначаются?
2. Объясните следующие понятия СКВ и их отношения: хранилище, commit, история, рабочая копия.
3. Что представляют собой и чем отличаются централизованные и децентрализованные СКВ? Приведите примеры СКВ каждого вида.
4. Опишите действия с СКВ при единоличной работе с хранилищем.
5. Опишите порядок работы с общим хранилищем в централизованной СКВ.
6. Что такое и зачем может быть нужна разность (diff)?
7. Что такое и зачем может быть нужно слияние (merge)?
8. Что такое конфликты (conflict) и каков процесс их разрешения (resolve)?
9. Поясните процесс синхронизации с общим хранилищем («обновления») в децентрализованной СКВ.
10. Что такое и зачем могут быть нужны ветви (branches)?
11. Объясните смысл действия rebase в СКВ Git.
12. Как и зачем можно игнорировать некоторые файлы при commit?
13. Что такое code review?
14. Из чего состоит review?
15. Как проводить design review?
16. Как проводить code review?
17. Утилиты для review.
18. Что такое assert?
19. Какие виды assert'ов бывают?
20. Что такое целочисленное переполнение?
21. Что такое переполнение стэка?
22. Когда и где стоит использовать assert'ы? Когда можно обойтись без assert'ов?
23. Когда нельзя использовать assert'ы?
24. Различные типы лицензий.
25. Подходы, применяемые в проектах с открытым исходным кодом для работы со сторонними разработчиками.

Приложение №2 к рабочей программе дисциплины «Системы контроля версий»

Методические указания для студентов по освоению дисциплины

Студенту желательно проявлять активное участие на занятиях, задавать вопросы, поскольку умение обосновывать свою точку зрения, нахождение компромиссного решения в этически выдержанной дискуссии не только важно для лучшего усвоения материала, но и ценится в реальной жизни. Важным моментом при изучении любой дисциплины является организация самостоятельной работы.

Учебный курс строится таким образом, чтобы способствовать созданию у студента понятийно-теоретического ядра и развитию практического навыка.

В освоении дисциплины (модуля) инвалидами и лицами с ограниченными возможностями здоровья большое значение имеет индивидуальная работа, т.е. дополнительное разъяснение учебного материала и углубленное изучение материала с теми обучающимися, которые в этом заинтересованы, и индивидуальная воспитательная работа. Индивидуальные консультации по предмету является важным фактором, способствующим индивидуализации обучения и установлению воспитательного контакта между преподавателем и обучающимся инвалидом или обучающимся с ограниченными возможностями здоровья.

Важная форма учебного процесса – это самостоятельная работа студента.

Самостоятельная работа - планируемая учебная, учебно-исследовательская, научно-исследовательская работа студентов, выполняемая во внеаудиторное (аудиторное) время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия (при частичном непосредственном участии преподавателя, оставляющем ведущую роль за работой студентов).

Самостоятельная работа студентов в ВУЗе является важным видом учебной и научной деятельности студента. Государственным стандартом предусматривается, как правило, 50% часов из общей трудоемкости дисциплины на самостоятельную работу студентов (далее СРС). В связи с этим, обучение в ВУЗе включает в себя две, практически одинаковые по объему и взаимовлиянию части – процесса обучения и процесса самообучения. Поэтому СРС должна стать эффективной и целенаправленной работой студента.

Концепцией модернизации российского образования определены основные задачи профессионального образования - подготовка квалифицированного работника соответствующего уровня и профиля, конкурентоспособного на рынке труда, компетентного, ответственного, свободно владеющего своей профессией и ориентированного в смежных областях деятельности, способного к эффективной работе по специальности на уровне мировых стандартов, готового к постоянному профессиональному росту, социальной и профессиональной мобильности.

Решение этих задач невозможно без повышения роли самостоятельной работы студентов над учебным материалом, усиления ответственности преподавателей за развитие навыков самостоятельной работы, за стимулирование профессионального роста студентов, воспитание творческой активности и инициативы.

Формирование такого умения происходит в течение всего периода обучения через участие студентов в практических занятиях, выполнение контрольных заданий и тестов,

написание курсовых и выпускных квалификационных работ. При этом самостоятельная работа студентов играет решающую роль в ходе всего учебного процесса.

Формы самостоятельной работы студентов разнообразны. Они включают в себя:

- разработку программ и приложений; изучение существующих программных продуктов; поиск нестандартных решений с использованием информационно-поисковых систем и глобальной сети Интернет;
- изучение учебной, научной и методической литературы, материалов периодических изданий с привлечением электронных средств официальной, статистической, периодической и научной информации;
- подготовку докладов и рефератов, написание курсовых и выпускных квалификационных работ;
- участие в работе студенческих конференций, комплексных научных исследованиях.

Самостоятельная работа приобщает студентов к научному творчеству, поиску и решению актуальных современных проблем.

Ведущая цель организации и осуществления СРС должна совпадать с целью обучения студента – подготовкой специалиста и бакалавра с высшим образованием. При организации СРС важным и необходимым условием становятся формирование умения самостоятельной работы для приобретения знаний, навыков и возможности организации учебной и научной деятельности.

Целью самостоятельной работы студентов является овладение фундаментальными знаниями, профессиональными умениями и навыками деятельности по профилю, опытом творческой, исследовательской деятельности. Самостоятельная работа студентов способствует развитию самостоятельности, ответственности и организованности, творческого подхода к решению проблем учебного и профессионального уровня.

Задачами СРС являются:

- систематизация и закрепление полученных теоретических знаний и практических умений студентов;
- углубление и расширение теоретических знаний;
- формирование умений использовать нормативную, правовую, справочную документацию и специальную литературу;
- развитие познавательных способностей и активности студентов: творческой инициативы, самостоятельности, ответственности и организованности;
- формирование самостоятельности мышления, способностей к саморазвитию, самосовершенствованию и самореализации;
- развитие исследовательских умений;
- использование материала, собранного и полученного в ходе самостоятельных занятий на семинарах, на практических и лабораторных занятиях, при написании курсовых и выпускной квалификационной работ, для эффективной подготовки к итоговым зачетам и экзаменам.

Выполняя самостоятельную работу под контролем преподавателя студент должен:

- освоить минимум содержания, выносимый на самостоятельную работу студентов и предложенный преподавателем в соответствии с Государственными образовательными стандартами высшего образования по данной дисциплине.

- планировать самостоятельную работу в соответствии с графиком самостоятельной работы, предложенным преподавателем.

- самостоятельную работу студент должен осуществлять в организационных формах, предусмотренных учебным планом и рабочей программой преподавателя.

- выполнять самостоятельную работу и отчитываться по ее результатам в соответствии с графиком представления результатов, видами и сроками отчетности по самостоятельной работе студентов.

студент может:

сверх предложенного преподавателем (при обосновании и согласовании с ним) и минимума обязательного содержания, определяемого ГОС ВО по данной дисциплине:

- самостоятельно определять уровень (глубину) проработки содержания материала;
- предлагать дополнительные темы и вопросы для самостоятельной проработки;
- в рамках общего графика выполнения самостоятельной работы предлагать обоснованный индивидуальный график выполнения и отчетности по результатам самостоятельной работы;

- предлагать свои варианты организационных форм самостоятельной работы;
- использовать для самостоятельной работы методические пособия, учебные пособия, разработки сверх предложенного преподавателем перечня;

- использовать не только контроль, но и самоконтроль результатов самостоятельной работы в соответствии с методами самоконтроля, предложенными преподавателем или выбранными самостоятельно.

Самостоятельная работа студентов должна оказывать важное влияние на формирование личности будущего специалиста, она планируется студентом самостоятельно. Каждый студент самостоятельно определяет режим своей работы и меру труда, затрачиваемого на овладение учебным содержанием по каждой дисциплине. Он выполняет внеаудиторную работу по личному индивидуальному плану, в зависимости от его подготовки, времени и других условий.

Основной формой самостоятельной работы студента является изучение конспекта лекций, их дополнение, рекомендованной литературы, активное участие на практических и семинарских занятиях.

Работа с книгой.

При работе с книгой необходимо подобрать литературу, научиться правильно ее читать, вести записи. Для подбора литературы в библиотеке используются алфавитный и систематический каталоги.

Важно помнить, что рациональные навыки работы с книгой - это всегда большая экономия времени и сил.

Правильный подбор учебников рекомендуется преподавателем, читающим лекционный курс. Необходимая литература может быть также указана в методических разработках по данному курсу.

Изучая материал по учебнику, следует переходить к следующему вопросу только после правильного уяснения предыдущего, описывая на бумаге все выкладки и вычисления (в том числе те, которые в учебнике опущены или на лекции даны для самостоятельного вывода).

При изучении любой дисциплины большую и важную роль играет самостоятельная индивидуальная работа.

Особое внимание следует обратить на определение основных понятий курса. Студент должен подробно разбирать примеры, которые поясняют такие определения, и уметь строить аналогичные примеры самостоятельно. Нужно добиваться точного представления

о том, что изучаешь. Полезно составлять опорные конспекты. При изучении материала по учебнику полезно в тетради (на специально отведенных полях) дополнять конспект лекций. Там же следует отмечать вопросы, выделенные студентом для консультации с преподавателем.

Выводы, полученные в результате изучения, рекомендуется в конспекте выделять, чтобы они при перечитывании записей лучше запоминались.

Опыт показывает, что многим студентам помогает составление листа опорных сигналов, содержащего важнейшие и наиболее часто употребляемые формулы и понятия. Такой лист помогает запомнить формулы, основные положения лекции, а также может служить постоянным справочником для студента.

Различают два вида чтения; первичное и вторичное. Первичное - это внимательное, неторопливое чтение, при котором можно остановиться на трудных местах. После него не должно остаться ни одного непонятного слова. Содержание не всегда может быть понятно после первичного чтения.

Задача вторичного чтения - полное усвоение смысла целого (по счету это чтение может быть и не вторым, а третьим или четвертым).

Правила самостоятельной работы с литературой.

Как уже отмечалось, самостоятельная работа с учебниками и книгами (а также самостоятельное теоретическое исследование проблем, обозначенных преподавателем на лекциях) – это важнейшее условие формирования у себя научного способа познания. Основные советы здесь можно свести к следующим:

- Составить перечень книг, с которыми Вам следует познакомиться;
- Сам такой перечень должен быть систематизированным (что необходимо для семинаров, что для экзаменов, что пригодится для написания курсовых и дипломных работ, а что Вас интересует за рамками официальной учебной деятельности, то есть что может расширить Вашу общую культуру...).
- Обязательно выписывать все выходные данные по каждой книге (при написании курсовых и дипломных работ это позволит очень сэкономить время).
- Разобраться для себя, какие книги (или какие главы книг) следует прочитать более внимательно, а какие – просто просмотреть.
- При составлении перечней литературы следует посоветоваться с преподавателями и научными руководителями (или даже с более подготовленными и эрудированными сокурсниками), которые помогут Вам лучше сориентироваться, на что стоит обратить большее внимание, а на что вообще не стоит тратить время.
- Естественно, все прочитанные книги, учебники и статьи следует конспектировать, но это не означает, что надо конспектировать все подряд: можно выписывать кратко основные идеи автора и иногда приводить наиболее яркие и показательные цитаты (с указанием страниц).
- Если книга – Ваша собственная, то допускается делать на полях книги краткие пометки или же в конце книги, на пустых страницах просто сделать свой предметный указатель, где отмечаются наиболее интересные для Вас мысли и обязательно указываются страницы в тексте автора (это очень хороший совет, позволяющий экономить время и быстро находить избранные места в самых разных книгах).
- Если Вы раньше мало работали с научной литературой, то следует выработать в себе способность воспринимать сложные тексты; для этого лучший прием – научиться читать медленно, когда Вам понятно каждое прочитанное слово (а если слово незнакомое, то либо

с помощью словаря, либо с помощью преподавателя обязательно его узнать), и это может занять немалое время (у кого-то – до нескольких недель и даже месяцев); опыт показывает, что после этого студент каким-то чудом начинает буквально заглатывать книги и чуть ли не видеть сквозь обложку, стоящая это работа или нет.

• Либо читайте, либо перелистывайте материал, но не пытайтесь читать быстро... Если текст меня интересует, то чтение, размышление и даже фантазирование по этому поводу сливаются в единый процесс, в то время как вынужденное скорочтение не только не способствует качеству чтения, но и не приносит чувства удовлетворения, которое мы получаем, размышляя о прочитанном, – советует Г. Селье (Селье, 1987. – С. 325-326).

• Есть еще один эффективный способ оптимизировать знакомство с научной литературой – следует увлечься какой-то идеей и все книги просматривать с точки зрения данной идеи. В этом случае студент (или молодой ученый) будет как бы искать аргументы за или против интересующей его идеи, и одновременно он будет как бы общаться с авторами этих книг по поводу своих идей и размышлений... Проблема лишь в том, как найти свою идею.

Чтение научного текста является частью познавательной деятельности. Ее цель – извлечение из текста необходимой информации. От того на сколько осознанна читающим собственная внутренняя установка при обращении к печатному слову (найти нужные сведения, усвоить информацию полностью или частично, критически проанализировать материал и т.п.) во многом зависит эффективность осуществляемого действия.

Выделяют четыре основные установки в чтении научного текста:

1. информационно-поисковый (задача – найти, выделить искомую информацию)
2. усваивающая (усилия читателя направлены на то, чтобы как можно полнее осознать и запомнить как сами сведения излагаемые автором, так и всю логику его рассуждений)
3. аналитико-критическая (читатель стремится критически осмыслить материал, проанализировав его, определив свое отношение к нему)
4. творческая (создает у читателя готовность в том или ином виде – как отправной пункт для своих рассуждений, как образ для действия по аналогии и т.п. – использовать суждения автора, ход его мыслей, результат наблюдения, разработанную методику, дополнить их, подвергнуть новой проверке).

Методические рекомендации по составлению конспекта:

1. Внимательно прочитайте текст. Уточните в справочной литературе непонятные слова. При записи не забудьте вынести справочные данные на поля конспекта;
2. Выделите главное, составьте план;
3. Кратко сформулируйте основные положения текста, отметьте аргументацию автора;
4. Законспектируйте материал, четко следуя пунктам плана. При конспектировании старайтесь выразить мысль своими словами. Записи следует вести четко, ясно.
5. Грамотно записывайте цитаты. Цитируя, учитывайте лаконичность, значимость мысли.

В тексте конспекта желательно приводить не только тезисные положения, но и их доказательства. При оформлении конспекта необходимо стремиться к емкости каждого предложения. Мысли автора книги следует излагать кратко, заботясь о стиле и выразительности написанного. Число дополнительных элементов конспекта должно быть логически обоснованным, записи должны распределяться в определенной

последовательности, отвечающей логической структуре произведения. Для уточнения и дополнения необходимо оставлять поля.

Овладение навыками конспектирования требует от студента целеустремленности, повседневной самостоятельной работы.

Практические занятия.

Для того чтобы практические занятия приносили максимальную пользу, необходимо помнить, что упражнение и решение задач проводятся по вычитанному на лекциях материалу и связаны, как правило, с детальным разбором отдельных вопросов лекционного курса. Следует подчеркнуть, что только после усвоения лекционного материала с определенной точки зрения (а именно с той, с которой он излагается на лекциях) он будет закрепляться на практических занятиях как в результате обсуждения и анализа лекционного материала, так и с помощью решения проблемных ситуаций, задач. При этих условиях студент не только хорошо усвоит материал, но и научится применять его на практике, а также получит дополнительный стимул (и это очень важно) для активной проработки лекции.

При самостоятельном решении задач нужно обосновывать каждый этап решения, исходя из теоретических положений курса. Если студент видит несколько путей решения проблемы (задачи), то нужно сравнить их и выбрать самый рациональный. Полезно до начала вычислений составить краткий план решения проблемы (задачи). Решение проблемных задач или примеров следует излагать подробно, вычисления располагать в строгом порядке, отделяя вспомогательные вычисления от основных. Решения при необходимости нужно сопровождать комментариями, схемами, чертежами и рисунками.

Следует помнить, что решение каждой учебной задачи должно доводиться до окончательного логического ответа, которого требует условие, и по возможности с выводом. Полученный ответ следует проверить способами, вытекающими из существа данной задачи. Полезно также (если возможно) решать несколькими способами и сравнить полученные результаты. Решение задач данного типа нужно продолжать до приобретения твердых навыков в их решении.

Самопроверка.

После изучения определенной темы по записям в конспекте и учебнику, а также решения достаточного количества соответствующих задач на практических занятиях и самостоятельно студенту рекомендуется, используя лист опорных сигналов, воспроизвести по памяти определения, выводы формул, формулировки основных положений и доказательств.

В случае необходимости нужно еще раз внимательно разобраться в материале.

Иногда недостаточность усвоения того или иного вопроса выясняется только при изучении дальнейшего материала. В этом случае надо вернуться назад и повторить плохо усвоенный материал. Важный критерий усвоения теоретического материала - умение решать задачи или пройти тестирование по пройденному материалу. Однако следует помнить, что правильное решение задачи может получиться в результате применения механически заученных формул без понимания сущности теоретических положений.

Консультации

Если в процессе самостоятельной работы над изучением теоретического материала или при решении задач у студента возникают вопросы, разрешить которые самостоятельно не удастся, необходимо обратиться к преподавателю для получения у него разъяснений или указаний. В своих вопросах студент должен четко выразить, в чем он испытывает

затруднения, характер этого затруднения. За консультацией следует обращаться и в случае, если возникнут сомнения в правильности ответов на вопросы самопроверки.

Подготовка к экзаменам и зачетам.

Изучение многих общепрофессиональных и специальных дисциплин завершается экзаменом. Подготовка к экзамену способствует закреплению, углублению и обобщению знаний, получаемых, в процессе обучения, а также применению их к решению практических задач. Готовясь к экзамену, студент ликвидирует имеющиеся пробелы в знаниях, углубляет, систематизирует и упорядочивает свои знания. На экзамене студент демонстрирует то, что он приобрел в процессе обучения по конкретной учебной дисциплине.

Экзаменационная сессия - это серия экзаменов, установленных учебным планом. Между экзаменами интервал 3-4 дня. Не следует думать, что 3-4 дня достаточно для успешной подготовки к экзаменам.

В эти 3-4 дня нужно систематизировать уже имеющиеся знания. На консультации перед экзаменом студентов познакомят с основными требованиями, ответят на возникшие у них вопросы. Поэтому посещение консультаций обязательно.

Требования к организации подготовки к экзаменам те же, что и при занятиях в течение семестра, но соблюдаться они должны более строго. Во-первых, очень важно соблюдение режима дня; сон не менее 8 часов в сутки, занятия заканчиваются не позднее, чем за 2-3 часа до сна. Оптимальное время занятий, особенно по математике - утренние и дневные часы. В перерывах между занятиями рекомендуются прогулки на свежем воздухе, неустойчивые занятия спортом. Во-вторых, наличие хороших собственных конспектов лекций. Даже в том случае, если была пропущена какая-либо лекция, необходимо во время ее восстановить (переписать ее на кафедре), обдумать, снять возникшие вопросы для того, чтобы запоминание материала было осознанным.

В-третьих, при подготовке к экзаменам у студента должен быть хороший учебник или конспект литературы, прочитанной по указанию преподавателя в течение семестра. Здесь можно эффективно использовать листы опорных сигналов.

Вначале следует просмотреть весь материал по сдаваемой дисциплине, отметить для себя трудные вопросы. Обязательно в них разобраться. В заключение еще раз целесообразно повторить основные положения, используя при этом листы опорных сигналов.

Систематическая подготовка к занятиям в течение семестра позволит использовать время экзаменационной сессии для систематизации знаний.

Правила подготовки к зачетам и экзаменам:

- Лучше сразу сориентироваться во всем материале и обязательно расположить весь материал согласно экзаменационным вопросам (или вопросам, обсуждаемым на семинарах), эта работа может занять много времени, но все остальное – это уже технические детали (главное – это ориентировка в материале!).

- Сама подготовка связана не только с запоминанием. Подготовка также предполагает и переосмысление материала, и даже рассмотрение альтернативных идей.

- Готовить шпаргалки полезно, но пользоваться ими рискованно. Главный смысл подготовки шпаргалок – это систематизация и оптимизация знаний по данному предмету, что само по себе прекрасно – это очень сложная и важная для студента работа, более сложная и важная, чем простое поглощение массы учебной информации. Если студент самостоятельно подготовил такие шпаргалки, то, скорее всего, он и экзамены сдавать будет

более уверенно, так как у него уже сформирована общая ориентировка в сложном материале.

- Как это ни парадоксально, но использование шпаргалок часто позволяет отвечающему студенту лучше демонстрировать свои познания (точнее – ориентировку в знаниях, что намного важнее знания запомненного и тут же забытого после сдачи экзамена).

- Сначала студент должен продемонстрировать, что он усвоил все, что требуется по программе обучения (или по программе данного преподавателя), и лишь после этого он вправе высказать иные, желательно аргументированные точки зрения.

Правила написания научных текстов (рефератов, курсовых и дипломных работ):

- Важно разобраться сначала, какова истинная цель Вашего научного текста - это поможет Вам разумно распределить свои силы, время и.

- Важно разобраться, кто будет читателем Вашей работы.

- Писать серьезные работы следует тогда, когда есть о чем писать и когда есть настроение поделиться своими рассуждениями.

- Как создать у себя подходящее творческое настроение для работы над научным текстом (как найти вдохновение)? Во-первых, должна быть идея, а для этого нужно научиться либо относиться к разным явлениям и фактам несколько критически (своя идея – как иная точка зрения), либо научиться увлекаться какими-то известными идеями, которые нуждаются в доработке (идея – как оптимистическая позиция и направленность на дальнейшее совершенствование уже известного). Во-вторых, важно уметь отвлекаться от окружающей суеты (многие талантливые люди просто пропадают в этой суете), для чего важно уметь выделять важнейшие приоритеты в своей учебно-исследовательской деятельности. В-третьих, научиться организовывать свое время, ведь, как известно, свободное (от всяких глупостей) время – важнейшее условие настоящего творчества, для него наконец-то появляется время. Иногда именно на организацию такого времени уходит немалая часть сил и талантов.

- Писать следует ясно и понятно, стараясь основные положения формулировать четко и недвусмысленно (чтобы и самому понятно было), а также стремясь структурировать свой текст.