Рабочая программа дисциплины (модуля)*

Современные технологии разработки программных систем искусственного интеллекта

Направление подготовки (специальность)

01.03.02 Прикладная математика и информатика

Направленность (профиль)

Прикладная математика и искусственный интеллект

Присваиваемая квалификация (степень)

бакалавр

Форма обучения

очная

Год набора 2024

*Рабочая программа дисциплины (модуля) адаптирована для инклюзивного обучения инвалидов и лиц с ограниченными возможностями здоровья

Челябинск 2023 г.

Содержание

1. ЦЕЛИ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Цели преподавания дисциплины: познакомить бакалавров с основными особенностями и проблемами разработки программного обеспечения; сформировать представление о современных тенденциях развития разработки ПО; изучить методические основы создания современных программных систем искусственного интеллекта; изучить требования предъявляемые к современным технологиям создания программного обеспечения; познакомить с технологиями создания ПО ведущих компаний в области разработки программных продуктов, использующих искусственный интеллект. Задачи изучения дисциплины: - познакомить студентов с современными технологиями разработки программных систем искусственного интеллекта; - познакомить современными подходами к выполнению основных технологических операций; - подготовить к командной работе над программными системами искусственного интеллекта.

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОПОП

Цикл (раздел) ОПОП:	Б1.О.36

2.1 Требования к предварительной подготовке обучающегося:

1. Компьютерные сети:

Знает: [УК-1.1. З-4.] основные методы оценки экономической эффективности применяемого программного и аппаратного обеспечения Умеет: [УК-1.1. У-3.] формировать и использовать критерии оценки эффективности применения программного и аппаратного обеспечения в профессиональной деятельности

2. Основы защиты данных в интеллектуальных системах:

Знает: [УК-1.2. З-1.] цели задачи и предмет, основные понятия информационной безопасности, информационные угрозы, их классификацию, возможные последствия для организаций различных форм собственности и критерии оценки защищённости информационных систем и систем искусственного интеллекта Умеет: [УК-1.2. У-2.] сознавать опасности и угрозы, возникающие в профессиональной деятельности и в социальной сфере, соблюдать основные требования информационной безопасности, в том числе защиты государственной тайны;[УК-1.2. У-3.] работать с информацией с учётом требованийинформационной безопасности

3. Объектно-ориентированное программирование:

нает: [ПК-2.2. З-1.] современные языки программирования, библиотеки и программные платформы для объектно-ориентированного программирования приложений систем интеллекта, методику разработки программ с использованием технологии объектноориентированного программирования, синтаксис языка объектно- ориентированного программирования С++; устройство и принципы построения объектно-ориентированных библиотек Умеет: реализовывать программно и использовать на практике математические алгоритмы, с применением высокоуровневого языка программирования С++, адаптировать и использовать шаблоны объектноориентированного программирования для решения профессиональных задач Имеет практический опыт: разработки компьютерных программ на языке С++, применения объектных технологий разработки программных систем

Компьютерные сети

Основы защиты данных в интеллектуальных системах

Объектно-ориентированное программирование

2.2 Дисциплины и практики, для которых освоение данной дисциплины (модуля) необходимо как предшествующее:

Функциональное и логическое программирование

Анализ требований и проектирование систем искусственного интеллекта

Технологии и модели управления проектами в интеллектуальных системах

Скриптовые языки программирования

3. КОМПЕТЕНЦИИ ОБУЧАЮЩЕГОСЯ, ФОРМИРУЕМЫЕ В РЕЗУЛЬТАТЕ ОСВОЕНИЯ ДИСЦИПЛИНЫ (МОДУЛЯ)

ОПК-4: Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности

Знать:

Знает: принципы работы современных информационных технологий

Уметь:

Умеет: использовать современные информационные технологии для решения задач профессиональной деятельности

Владеть:

Имеет практический опыт: применения современных информационных технологий при проектировании систем искусственного интеллекта

ПК-1: Способен активно участвовать в разработке системного и прикладного программного обеспечения

Знать:

Знает: современные компьютерные технологии разработки программных систем

Владеть:

Имеет практический опыт: участия в разработке прикладного программного обеспечения

УК-91: Способен планировать и организовывать свою деятельность в цифровом пространстве с учётом правовых и этических норм взаимодействия человека и искусственного интеллекта и требований информационной безопасности

Знать:

[УК-1.1. З-3.] современное состояние информационно-коммуникационных технологий в мире и перспективы их развития

ОПК-91: Способен анализировать, разрабатывать, внедрять и выполнять организационно технические и экономические процессы с применением технологий и систем искусственного интеллекта

Знать:

[ОПК-1.1. З-1.] рынок информационных систем и информационно-коммуникационных технологий, автоматизирующих организационнотехнические и экономические процессы

ПК-4: Способен разрабатывать и тестировать программные компоненты решения задач в системах искусственного интеллекта

Знать:

[ПК-2.1. З-1.] основные программные платформы и компоненты систем искусственного интеллекта: механизмы логического вывода (рассуждений), объяснений, приобретения знаний, интеллектуальных интерфейсов, принципы Data Ops и Dev Ops

Уметь:

[ПК-2.1. У-1.] настраивать основные программные платформы и компоненты систем искусственного интеллекта: механизмов логического вывода (рассуждений), объяснений, приобретения знаний, интеллектуальных интерфейсов на особенности проблемной области, участвует в их разработке

В результате освоения дисциплины обучающийся должен

3.1	**Знать:**
3.1.1	1. [ПК-2.1. З-1.] основные программные платформы и компоненты систем искусственного интеллекта: механизмы логического вывода (рассуждений), объяснений, приобретения знаний, интеллектуальных интерфейсов, принципы Data Ops и Dev Ops
3.1.2	2. современные компьютерные технологии разработки программных систем
3.1.3	3. [ОПК-1.1. З-1.] рынок информационных систем и информационно-коммуникационных технологий, автоматизирующих организационнотехнические и экономические процессы
3.1.4	4. принципы работы современных информационных технологий
3.1.5	5. [УК-1.1. З-3.] современное состояние информационно-коммуникационных технологий в мире и перспективы их развития
3.2	**Уметь:**
3.2.1	1. [ПК-2.1. У-1.] настраивать основные программные платформы и компоненты систем искусственного интеллекта: механизмов логического вывода (рассуждений), объяснений, приобретения знаний, интеллектуальных интерфейсов на особенности проблемной области, участвует в их разработке

3.2.2	2. использовать современные информационные технологии для решения задач профессиональной деятельности
3.3	**Владеть:**
3.3.1	1. участия в разработке прикладного программного обеспечения
3.3.2	2. применения современных информационных технологий при проектировании систем искусственного интеллекта

4. ОБЪЕМ ДИСЦИПЛИНЫ (МОДУЛЯ)

Общая трудоемкость	**3 ЗЕТ**
Часов по учебному плану : 108 в том числе : аудиторные занятия : 48 самостоятельная работа : 53,75 : контактная работа: 54,25 ИКР: 0	Виды контроля в семестрах: зачеты 5

5. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

Код занятия	Наименование разделов и тем /вид занятия/	Семестр / Курс	Часов	Литература
	Раздел 1. Жизненный цикл и этапы разработки ПО			
1.1	Программная инженерия. Проблемы разработки ПО. Стадии и процессы жизненного цикла ПО. Модели ЖЦ ПО. /Лек/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.2	Планирование и определение системы. Термины и методики (интервью, мозговой штурм), разработка спецификаций, управление масштабом. Особенности планирования внедрения систем искусственного интеллекта. /Лек/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.3	Проектирование архитектуры системы. Структура системы, модели управления, виды декомпозиции, архитектуры распределенных систем и систем искусственного интеллекта. /Лек/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.4	Объектно-ориентированное проектирование. Диаграммы UML. Выявление классов и выбор операций /Лек/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.5	Принципы проектирования интеллектуальных интерфейсов пользователя, способы взаимодействия, представление информации. /Лек/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.6	Реализация. Основные программные платформы и компоненты систем искусственного интеллекта: механизмы логического вывода (рассуждений), объяснений, приобретения знаний /Лек/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.7	Разработка через тестирование. Рефакторинг. /Лек/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.8	Принципы тестирования. Восходящее и нисходящее тестирование. Тестирование модуля как черного и белого ящика. /Лек/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.9	Системное тестирование. Тестирование систем искусственного интеллекта. Отладка (задача, методы, принципы). /Лек/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.10	Эксплуатация и сопровождение /Лек/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3

1.11	Проведение интервью с заказчиком /Лаб/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.12	Проведение мозгового штурма для определения функций ПО /Лаб/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.13	Подведение итогов мозгового штурма и выбор функций для 1-й версии /Лаб/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.14	Написание спецификации для функциональных и нефункциональных требований /Лаб/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.15	Проектирование архитектуры системы искусственного интеллекта с использованием UML /Лаб/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.16	Разработка диаграммы классов UML для системы искусственного интеллекта /Лаб/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.17	Разработка компонент для системы искусственного интеллекта /Лаб/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.18	Разработка через тестирование /Лаб/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.19	Тестирование модуля как белого ящика /Лаб/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.20	Тестирование модуля как чёрного ящика и тестирование системы /Лаб/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
1.21	Подготовка к лабораторным занятиям /Ср/	5	34,75	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
	Раздел 2. Управление разработкой ПО			
2.1	Классические и гибкие методы управление разработкой ПО. Бригада главного программиста. Экстремальное программирование. Scrum. Принципы Data Ops и Dev Ops. /Лек/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
2.2	CASE-средства. Классификация, примеры CASE-средств и их назначение (IDE, VCS и др.) /Лек/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
2.3	Использование библиотек для логирования, локализация ошибки с помощью отладчика в IDE /Лаб/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
2.4	Работа с системой контроля версий /Лаб/	5	2	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
2.5	Подготовка к лабораторным занятиям /Ср/	5	10	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
	Раздел 3. Иная контрольная работа			
3.1	Индивидуальные консультации, текущий контроль /КурсР/	5	6,25	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3
	Раздел 4. Зачет			

4.1	Подготовка к зачету /Ср/	5	9	Л1.1 Л1.2Л2.1 Л2.2 Л2.3 Л2.4 Э1 Э2 Э3

6. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

6.1. Перечень видов оценочных средств

Текущий контроль: выполнение лабораторных работ
Активность на занятиях, посещаемость, ответы на практических занятиях, участие в олимпиадах по программированию
Промежуточная аттестация - зачет.

6.2. Типовые контрольные задания и иные материалы для текущей аттестации

Смотри приложение

6.3. Типовые контрольные вопросы и задания для промежуточной аттестации

Вопросы к зачету
1. Программное обеспечение. Особенности разработки ПО.

2. Этапы разработки ПО (Software development lifecycle). Временные затраты.

3. Модели ЖЦ ПО (каскадная, инкрементальная, спиральная).

4. Гибкие (agile) методы разработки: Экстремальное программирование (XP)
(особенности, принципы, модель ЖЦ, состав команды)

5. DataOps и DevOps
(особенности, принципы)

6. Планирование системы (термины: проблемы, потребности и функции)
(временные затраты и задачи этапа, перечислить используемые методики).

7. Интервью (определение проблем и потребностей)
(цели, план интервью, итоги)

8. Мозговой штурм (определение функций)
(цели, принципы, подведение итогов)

9. Определение системы
(термины: функциональные и нефункциональные требования, спецификация требований)
(временные затраты и задачи этапа, перечислить используемые методики).

10. Проектирование архитектуры системы (Соммервиль главы 10, 11)
Структура системы, модели управления, виды декомпозиции,
архитектуры распределенных систем

11. Детальное проектирование системы.
Объектно-ориентированное проектирование. Основные виды диаграмм UML.
Диаграмма классов, классы и связи.

12. Проектирование интеллектуального интерфейса пользовате
Принципы, способы взаимодействия, представление информации.

13. Реализация (кодирование).
(временные затраты и задачи этапа,
основные программные платформы и компоненты систем искусственного интеллекта: механизмы логического вывода (рассуждений), объяснений, приобретения знаний, разработка через тестирование).

14. Тестирование.
Принципы тестирования. Тестирование и реализация.

Восходящее и нисходящее тестирование (термины: драйверы и заглушки), преимущества и недостатки.

15. Тестирование модуля как черного ящика (цели, используемые методики)

16. Тестирование модуля как белого ящика (цели, используемые методики)

17. Системное тестирование (цели, используемые методики)

18. Эксплуатация и сопровождение
(временные затраты и задачи этапа, перечислить используемые методики).

19. Отладка (задача, методы, принципы)

20. CASE-средства.
Что такое CASE-средство. Примеры CASE-средств и их назначение (IDE, VCS и др.)

6.4. Критерии оценивания

Текущий контроль: выполнение лабораторных работ - оценивается каждая от 0 до 10 баллов. - максиму 120 баллов - 60% от общего кол-ва баллов.
Активность на занятиях, посещаемость 100% посещение (допускаются пропуски уважительной причине) - 3 балла 85 -99% посещение - 2 балла по 0,1 балла за ответы на практических
занятиях. Участие в олимпиадах по программированию - не более 6 баллов - максимум 15 баллов - 7,5 % от общего объема баллов

Зачет:
Это контрольное мероприятие проводится в форме собеседования. Задаются два вопроса по пройденным темам. В первую очередь предлагаются вопросы по темам, которые были оценены на "неудовлетворительно" по текущему контролю. Каждый ответ оценивается от 0 до 5 баллов в зависимости от полноты ответа, знания терминов. Шкала оценивания Полный, правильный ответ - 5 баллов Одна неточность, неправильный термин - 4 балла Частичный ответ - 3 балла В ответе есть некоторые правильные определения - 2 балла Нет ответа - 0 баллов Оценка ставится как сумма баллов за оба ответа.
32,5 % от общего объема баллов

Если студент набирает:
от 0 до 60 баллов - оценка "не зачтено"
от 61 до 100 баллов - оценка "зачтено"

7. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

7.1. Рекомендуемая литература

7.1.1. Основная литература

	Авторы, составители	Заглавие	Издательство, год	Ресурс
Л1.1	Иванов Д., Новиков Ф.	Моделирование на UML (http://e.lanbook.com/books/element.php?pl1_id=40879)	Санкт-Петербург : НИУ ИТМО, 2010	ЭБС
Л1.2	Зубкова Т. М.	Технология разработки программного обеспечения: учебное пособие (https://e.lanbook.com/book/206882)	Санкт-Петербург : Лань, 2022	ЭБС

7.1.2. Дополнительная литература

	Авторы, составители	Заглавие	Издательство, год	Ресурс
Л2.1	Джесутасан Р., Будро Д.	Реинжиниринг бизнеса: как грамотно внедрить автоматизацию и искусственный интеллект: практическое пособие (https://znanium.com/catalog/document?id=352155)	Москва : ООО "Альпина Паблишер", 2019	ЭБС
Л2.2	Ехлаков Ю. П.	Управление программными проектами. Стандарты, модели (https://e.lanbook.com/book/175498)	Санкт-Петербург : Лань, 2021	ЭБС

	Авторы, составители	Заглавие	Издательство, год	Ресурс
Л2.3	Остроух А. В., Суркова Н. Е.	Системы искусственного интеллекта: монография (https://e.lanbook.com/book/310199)	Санкт-Петербург : Лань, 2023	ЭБС
Л2.4	Бурков А.	Инженерия машинного обучения (https://e.lanbook.com/book/314834)	Москва : ДМК Пресс, 2022	ЭБС

7.2. Перечень ресурсов информационно-телекоммуникационной сети "Интернет"

Э1	Электроннобиблиотечная система издательства Лань https://e.lanbook.com
Э2	Электроннобиблиотечная система издательства Юрайт https://urait.ru/
Э3	Электронно библиотечная система znanium https://znanium.com/

7.3 Перечень информационных технологий

7.3.1 Программное обеспечение

OpenOffice
Adobe Reader
NetBeans
Notepad++
Python
Visual Studio

7.3.2 Профессиональные базы данных и информационно-справочные системы

8. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

Компьютеры, проектор

Доска маркеры (мел)

9. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ (МОДУЛЯ)

10. СПЕЦИАЛЬНЫЕ УСЛОВИЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ ОБУЧАЮЩИМИСЯ С ИНВАЛИДНОСТЬЮ И ОГРАНИЧЕННЫМИ ВОЗМОЖНОСТЯМИ ЗДОРОВЬЯ

Освоение дисциплины инвалидами и лицами с ограниченными возможностями здоровья осуществляется с использованием специальных технических средств и голо информационных технологий, предоставляемых Ресурсным учебно-методическим центром по обучению инвалидов и лиц с ограниченными возможностями здоровья ЧелГУ по запросу обучающегося.

1. Мобильные специальные технические средства для лиц с нарушениями зрения: портативный компьютер с вводом/выводом шрифтом Брайля с синтезатором речи «ElBraile-W14J G2»; ноутбуки с программной экранного доступа NVDA; электронные увеличители для удаленного просмотра; видеоувеличители портативные; тифлоплеер; цифровые диктофоны.

2. Мобильные специальные технические средства для лиц с нарушениями слуха: система свободного звукового поля со встроенной совместимостью с FM-устройствами; радиоклассы «Сонет-РСМ» с передатчиком, заушным индуктором и индукционной петлей; система информационная для слабослышащих переносная «Исток» А2 со встроенным плеером – звуковым информатором; документ-камера; программируемые слуховые аппараты индивидуального пользования.

3. Ассистивные информационные технологии: программное обеспечение экранного доступа с синтезом речи NVDA; программы экранного увеличения; программы речевого синтеза для компьютеров и ноутбуков; программы речевого синтеза для мобильных устройств; экранная клавиатура; экранная лупа.

При необходимости для обучающихся с нарушениями зрения на рабочих местах для проведения практических или лабораторных занятий устанавливается специальное программное обеспечение (программа речевой навигации NVDA, речевые синтезаторы, экранные лупы).

В учебные аудитории обеспечивается беспрепятственный доступ для обучающихся инвалидов и обучающихся с ограниченными возможностями здоровья. В каждой аудитории, где обучаются инвалиды и лица с ограниченными возможностями здоровья, предусматривается соответствующее количество мест для обучающихся с учетом нарушений их здоровья.

Для освоения дисциплины инвалидам и лицам с ограниченными возможностями здоровья предоставляется доступ к

печатным источникам, имеющимся в научной библиотеке ЧелГУ, с помощью специальных технических средств; доступ к электронным источникам, представленным в форме электронного документа в фонде научной библиотеки ЧелГУ или электронно-библиотечных системах, с помощью специальных технических и программных средств (рабочее место для незрячего пользователя с программным обеспечением экранного доступа с синтезом речи NVDA, рабочее место с компьютерным роллером и клавиатурой Clevy с большими кнопками и с разделяющей клавиши накладкой).

Учебно-методические материалы для обучающихся из числа инвалидов и лиц с ограниченными возможностями здоровья предоставляются в формах, адаптированных к ограничениям их здоровья и восприятия информации:

Для лиц с нарушениями зрения:

- в печатной форме увеличенным шрифтом,
- в форме электронного документа,
- в форме аудиофайла,
- в печатной форме шрифтом Брайля.

Для лиц с нарушениями слуха:

- в печатной форме,
- в форме электронного документа.

Для лиц с нарушениями опорно-двигательного аппарата:

- в печатной форме,
- в форме электронного документа,
- в форме аудиофайла.

Данный перечень может быть конкретизирован в зависимости от контингента обучающихся.

Для инвалидов и лиц с ограниченными возможностями здоровья освоение дисциплины может быть частично или полностью осуществлено с использованием дистанционных образовательных технологий (Moodle, Adobe Connect Pro и пр.).

В освоении дисциплины инвалидами и лицами с ограниченными возможностями здоровья используется индивидуальная работа. Под индивидуальной работой подразумевается две формы взаимодействия с преподавателем: индивидуальная учебная работа (консультации), т.е. дополнительное разъяснение учебного материала и углубленное изучение материала с теми обучающимися, которые в этом заинтересованы, и индивидуальная воспитательная работа. Индивидуальные консультации направлены на индивидуализацию обучения и установлению воспитательного контакта между преподавателем и обучающимся инвалидом или обучающимся с ограниченными возможностями здоровья.

При проведении процедуры оценивания результатов обучения инвалидов и лиц с ограниченными возможностями здоровья по дисциплине обеспечивается выполнение следующих дополнительных требований в зависимости от индивидуальных особенностей, обучающихся:

а) инструкция по порядку проведения процедуры оценивания предоставляется в доступной форме (устно, в письменной форме, в письменной форме шрифтом Брайля, устно с использованием услуг сурдопереводчика);

б) доступная форма предоставления заданий оценочных средств (в печатной форме, в печатной форме увеличенным шрифтом, в печатной форме шрифтом Брайля, в форме электронного документа, задания зачитываются ассистентом, задания предоставляются с использованием сурдоперевода);

в) доступная форма предоставления ответов на задания (письменно на бумаге, набор ответов на компьютере, письменно шрифтом Брайля, с использованием услуг ассистента, устно).

При проведении процедуры оценивания результатов обучения инвалидов и лиц с ограниченными возможностями здоровья предусматривается использование технических средств, необходимых им в связи с их индивидуальными особенностями. Эти средства могут быть предоставлены ЧелГУ или могут использоваться собственные технические средства. При необходимости инвалидам и лицам с ограниченными возможностями здоровья предоставляется дополнительное время для подготовки ответа на задания, процедура оценивания результатов обучения по дисциплине может проводиться в несколько этапов.

Проведение процедуры оценивания результатов обучения инвалидов и лиц с ограниченными возможностями здоровья допускается с использованием дистанционных образовательных технологий.

№ 1 Лабораторная

Предметная область выбирается на 3 следующие лабораторные работы
Примеры:

- Обучение иностранному языку
- Доска электронных объявлений на остановке
- Проведение игр в страйкбол
- Сайт для профессионального фотографа (съёмки портфолио, мероприятий)

Разделение на группы
В каждой группе:
1-2 заказчика, являющиеся "экспертами" в выбранной предметной области
2 аналитика

Для заказчика
- Выбрать тему, в которой вы разбираетесь (свое хобби, работа родителей, из реальной жизни, в которой участвовали как клиент)
- Описывать проблемы и потребности, но не решения
- Придумать должность

Интервью является способом выявления проблем и потребностей.
Нужно различать проблему и потребность
Пример проблемы
Товар не продается
Товар не продается, так как у него низкое качество
Пример потребности
Нужно товар с приемлимым соотношением цена/качество

Для аналитика
Проведение интервью
- Важно задавать контекстно-свободные вопросы, которые не навязывают конкретного решения, а позволяют выслушать заказчика, узнать о его проблемах.
- Рекомендуется подготовить набор вопросов заранее (см. план интервью).
- Кратко записывайте ответы в блокнот (_не_ на диктофон!)
- Не страшно, если отклонитесь от сценария.
- Не перебивайте, пока заказчик красочно описывает текущие проблемы, уточняющие вопросы можно задать потом.

Далее приводится примерный план интервью Лэффенгуэлл Д. Принципы работы с требованиями к программному обеспечению. Унифицированный подход, глава 9

===План интервью===
I Определение профиля заказчика
Имя, компания, должность (может быть выяснено заранее)
Каковы ваши обязанности?
Кому необходимы результаты вашей деятельности?
Что считается успехом в вашей деятельности?
Какие проблемы влияют на успешность вашей деятельности?
Какие тенденции, если такие существуют, делают вашу работу проще или сложнее?

II. Оценка проблемы
Для каких проблем (прикладного типа) вы ощущаете нехватку хороших решений?
Назовите их. (Не забывайте спрашивать: "А ещё?")
Для каждой проблемы выясняйте следующее.
- Почему существует эта проблема?
- Как она решается в настоящее время?
- Как заказчик (пользователь) хотел бы ее решать?

III. Понимание пользовательской среды
Кто такие пользователи?
Какое у них образование?
Каковы их навыки в компьютерной области?
Имеют ли пользователи опыт работы с данным типом приложений?
Какая платформа используется? Каковы ваши планы относительно будущих платформ?
Используются ли дополнительные приложения, которые имеют отношение к данному приложению?
Если да, то пусть о них немного расскажут.
Каковы ожидания заказчика относительно удобства продукта?
Сколько времени необходимо для обучения?
В каком виде должна быть представлена справочная информация для пользователя
(в интерактивном или в виде печатной копии)?

IV. Резюме (перечисляются основные пункты, чтобы проверить, все ли правильно вы поняли)

Итак, вы сказали мне ... (перечислите описанные заказчиком проблемы своими словами)
Адекватно ли этот список представляет проблемы, которые имеются при существую¬щем решении?
Какие ещё проблемы (если такие существуют) вы испытываете?

V. Предположения аналитика относительно проблемы заказчика (проверенные или непроверенные предположения, те проблемы, которые не были упомянуты)
Какие проблемы, если они есть, связаны с (перечислите все потребности или дополнительные проблемы, которые, по-вашему, может испытывать заказчик или пользователь)
Для каждой из указанных проблем выясните следующее.
- Является ли она реальной?
- Каковы ее причины?
- Как она решается в настоящее время?
- Как бы заказчик (пользователь) хотел ее решать?
- Насколько важно для заказчика (пользователя) решение этой проблемы в сравнении с другими, упомянутыми им?

VI. Оценка предлагаемого вами решения (если это уместно)
(Охарактеризуйте основные возможности предлагаемого вами решения, а потом задайте пользователю следующие вопросы.)
Что, если вы сможете ...
Как вы расцениваете важность этого?

VII. Оценка возможности
Кто в организации нуждается в данном приложении?
Сколько пользователей указанных типов будет использовать его?
Насколько значимо для вас успешное решение?

VIII. Оценка необходимого уровня надежности и производительности, а также потребности в сопровождении
Каковы ваши ожидания относительно надежности?
Какой, по-вашему, должна быть производительность?
Будете ли вы заниматься поддержкой продукта или этим будут заниматься другие?
Испытываете ли вы потребности в поддержке?
Что вы думаете о доступе для сопровождения и обслуживания?
Каковы требования относительно безопасности?
Какие требования относительно установки и конфигурации?
Существуют ли специальные требования по лицензированию?

Как будет распределено программное обеспечение?
Есть ли требования на маркировку и упаковку?

IX. Другие требования
Существуют ли законодательные требования, требования информационной среды,
инструкции или другие стандарты, которых необходимо придерживаться?
Нет ли других требований, о которых нам следовало бы знать?

X. Окончание
Существуют ли другие вопросы, которые мне следовало бы вам задать?
Если мне еще понадобится задать вам несколько вопросов, могу ли я вам позвонить?
Будете ли вы принимать участие в обсуждении требований?

=Заключение аналитика=
После интервью, пока его данные еще свежи в вашей памяти, зафиксируйте три потребности или проблемы с наивысшими приоритетами, выявленные вами в беседе с данным заказчиком (пользователем).

№2 Лабораторная

Продолжение работы над предметной областью из ЛР1.

Роли аннулируются, все участники группы из 3-4 человек могут рассматривать проблему с любой точки зрения, как эксперта в этой области, так и разработчика ПО.

Каждый участник должен записывать свои идеи своими словами на листке бумаги и озвучивать их, если в этот момент не говорит кто-то другой.

Для большой группы участников рекомендуется записывать идеи на небольших листах толстым маркером и прикреплять на доску. Нельзя, чтобы идеи записывал на доске секретарь, так как может получиться затор в творческом процессе.
Правила:
- Не допускается критика или дебаты
- Дайте свободу фантазии
- Генерируйте как можно больше идей
- Переделывайте и комбинируйте чужие идеи
Нельзя говорить "это дурацкая идея", "это невозможно", "это уже было" или "почти то же самое", но можно и нужно хвалить идеи для стимуляции творческого процесса.
Наличие "сумасбродных" идей является индикатором качества процесса. Возможно, кто-то найдет рациональное зерно в такой идее и предложит более реалистичный упрощенный вариант.
Генерация идей происходит пока перерывы в появлении идей не станут слишком большими.

Количество идей не ограничивается 5, хотя баллы начисляются только за первые 5 идей, связанных с компьютерами и ПО, поэтому можно сгенерировать 10-20 и более, и не только из области компьютеров или реалистичные, так как все идеи будут использованы на ЛР 3.

Список идей отдается преподавателю для оценки, на каждом листке должна быть написана фамилия, инициалы и группа. Листки возвращаются на следующем занятии для выполнения ЛР3.

№3 Лабораторная

Порядок выполнения

Группе из 3-4 студентов возвращаются списке функций, предложенных ими на предыдущей лабораторной работе.

Сначала отсекаются непродуктивные идеи.
Ведущий зачитывает идею, при необходимости уточняет определение у автора.
Если хотя бы один участник высказывается за идею, то она остается на доске.
Похожие идеи могут объединяться и прикрепляться на одну кнопку.
Можно разделять идеи на доске по темам.

Список идей вносится в таблицу (Excel/Calc) и каждый участник должен отнести
идею к одной из трех групп:
- критическая, без этой функции система никому не нужна (9 баллов);
- важная, система без этой функции сильно потеряет,
  пользователи будут приобретать (использовать) продукт, но могут перейти
  на конкурирующий продукт, если он предоставит подобную функцию (3 балла);
- полезная, система будет немного удобнее в использовании (1 балл).

К каждой группе нужно отнести ровно 1/3 идей.
Голосование проводится тайно, для этого каждый участник выставляет баллы в собственной (подписанной его ФИО) колонке и после голосования прячет её, чтобы его выбор не повлиял не решения других участников.
Для подведения итогов колонки открываются, результаты суммируются и список сортируется в порядке уменьшения суммы баллов.

Для определения набора функций для первой версии необходимо учесть
трудоемкость, риск и ограничения времени (бюджета) на разработку.

Вместо конкретных значений можно использовать качественные оценки:
Трудоемкость
- низкая (несколько часов, дней для 1 разработчика)
- средняя (1-2 недели для 1 разработчика)
- высокая (работа для нескольких разработчиков в течении недель)

Риск
- низкий (используются известные технологии, высокая точность оценки трудоемкости)
- средний (необходимо изучение новых технологий, трудоемкость может быть выше в 1.5-2 раза)
- высокий (об технологиях и способах решения неизвестно, трудоемкость может быть выше оценки в несколько раз)

В первую версию включаем функции из верхней части упорядоченного списка функций,
чтобы сумма трудоемкости не превышала заданную с учетом риска и количества разработчиков. Срок разработки (2-3 недели) умножается на количество разработчиков (3-4 человека).
Функции с высоким риском включаются только, если они являются критическими для системы. Из менее важных функции можно выбрать с низкой трудоемкостью и/или низким риском так, чтобы они дополняли более важные функции, а также приближали суммарную трудоемкость к заданному лимита для первой версии.

Таблица, пересылаемая в качестве ответа, должна содержать наименования функций, результаты голосования всех участников и сумму баллов, оценки трудоемкости и риска. Выбранные функции должны быть выделены цветом в таблице.

№4 Лабораторная

Задание выполняется индивидуально, каждый участник выбирает одну из функций, вошедшую в список на предыдущей лабораторной работы, и одно из 5 перечисленных ниже нефункциональных требований

Функциональные требования
Пример из лекции

Функция	Спецификация
Функция 63. Система обнаружения неполадок будет предоставлять информацию об обнаруженных дефектах, чтобы помочь пользователю оценить состояние проекта	SR63.1. Информация будет предоставляться в виде отчета-гистограммы, где по оси X откладывается время, а по оси Y — количество обнаруженных дефектов SR63.2. Пользователь может задавать временной период в днях, неделях или месяцах SR63.3. Пример отчета об обнаруженных дефектах представлен на прилагаемом рисунке (здесь рисунок)

Количество пунктов в спецификации не ограничивается 3, а определяется критерием полноты спецификации. Также каждый пункт должен определять один тестовый набор или проверяемое вручную условие. SR63.2 в примере это не три отдельных условия, так как выбор реализуется комбо-боксом, в котором при проверке должны выявлены быть указанные три варианта.

Нефункциональные требования (выбирается один из пунктов и указываются только характеристики, соответствующие предметной области, которым вы можете дать оценку).
Нефункциональные требования задаются не для одной функции, а для системы в целом или одной из подсистем (например, веб-сервер)

1. Практичность

Как правило, указывается следующее.

- Время, необходимое для обучения рядовых пользователей и пользователей с большими полномочиями, чтобы они научились эффективно выполнять определенные действия.

- Время выполнения типичных задач; или же практичность новой системы, сравнивается с практичностью известных систем, которые пользователь знает и любит.

- Требования соответствия общепринятым стандартам практичности, таким как CUA IBM или опубликованные компанией Microsoft стандарты GUI для системы Windows 98.

2. Надежность

К характеристикам надежности относятся.

- *Доступность.* Указывается, какой процент времени система доступна (xx.xx%), определяются часы использования и доступа для обслуживания, операции при ухудшении параметров системы и т.д.

- *Среднее время между отказами (mean time between failures, MTBF). Обычно* выражается в часах, но может указываться в днях, месяцах и годах.

- *Среднее время восстановления (mean time to repair, MTTR)* Сколько времени система может находиться в нерабочем состоянии после сбоя.

- *Точность.* С помощью некоего известного стандарта указывается требуемая точность (разрешающая способность) выводимой системой информации.

- *Максимально допустимый коэффициент ошибок и дефектов.* Как правило, выражается как число ошибок, приходящееся на K.LOS (тысячу строк кода), или число ошибок, приходящихся на отдельную функцию.

- *Доля ошибок или дефектов различных типов.* Обычно ошибки разбиваются на следующие категории: незначительные, серьезные и критические. Требования должны определять, что понимается под "критической" ошибкой (такой, как полная потеря данных или невозможность использовать определенную часть функциональных возможностей системы).

3. Производительность

Здесь описываются характеристики производительности системы. Следует указать время ответа для различных ситуаций. Если требуется, указываются названия соответствующих вариантов использования.

- Время ответа для транзакции (среднее, максимальное)

- Пропускная способность (транзакций в секунду)

- Емкость (число пользователей или транзакций, которые может обслужить система)

- Режимы снижения производительности (допустимые режимы работы при ухудшении параметров системы)

- Использование ресурсов (память, диск, каналы связи)

4. Возможность сопровождения

Указываются требования, способствующие улучшению возможности сопровождения и обслуживания создаваемой системы, в том числе стандарты кодирования, определенные соглашения, библиотеки классов, доступ для обслуживания и вспомогательные обслуживающие программы.

5. Ограничения проектирования

Возможные источники: экономические, политические, технические, системные, эксплуатационные, на разработку.

№5 Лабораторная

Задание выполняется в группе из 2 человек

Варианты зданий для проектирования архитектуры

1. Проектирование системы интернет-бронирования гостиницы.
2. Проектирование системы интернет-заказов товаров магазина электроники.
3. Проектирование системы предоставления и запроса вакансий для бюро по трудоустройству.
4. Проектирование системы интернет-заказов у поставщиков автозапчастей.
5. Проектирование системы записи и учета прохождения курсов повышения квалификации.
6. Проектирование электронной системы учета оценок студентов
7. Проектирование электронной системы записи на прием пациентов частной клиники.
8. Проектирование системы бронирования для проката автомобилей.
9. Проектирование системы учета рекламы в эфире телеканала.
10. Проектирование системы продажи и бронирования билетов кинотеатра через интернет.
11. Проектирование информационной системы турфирмы.
12. Проектирование системы покупки и бронирования билетов на поезд.
13. Проектирование системы учета телефонных разговоров сотрудников.
14. Проектирование интернет-кабинета клиента банка.
15. Проектирование информационной системы агенства недвижимости.
16. Проектирование интернет-системы записи и учета скидок клиентов салона красоты.
17. Проектирование интернет-системы заказа и доставки пиццы.
18. Проектирование информационной системы детского сада.

19. Проектирование системы курсов дистанционного обучения.

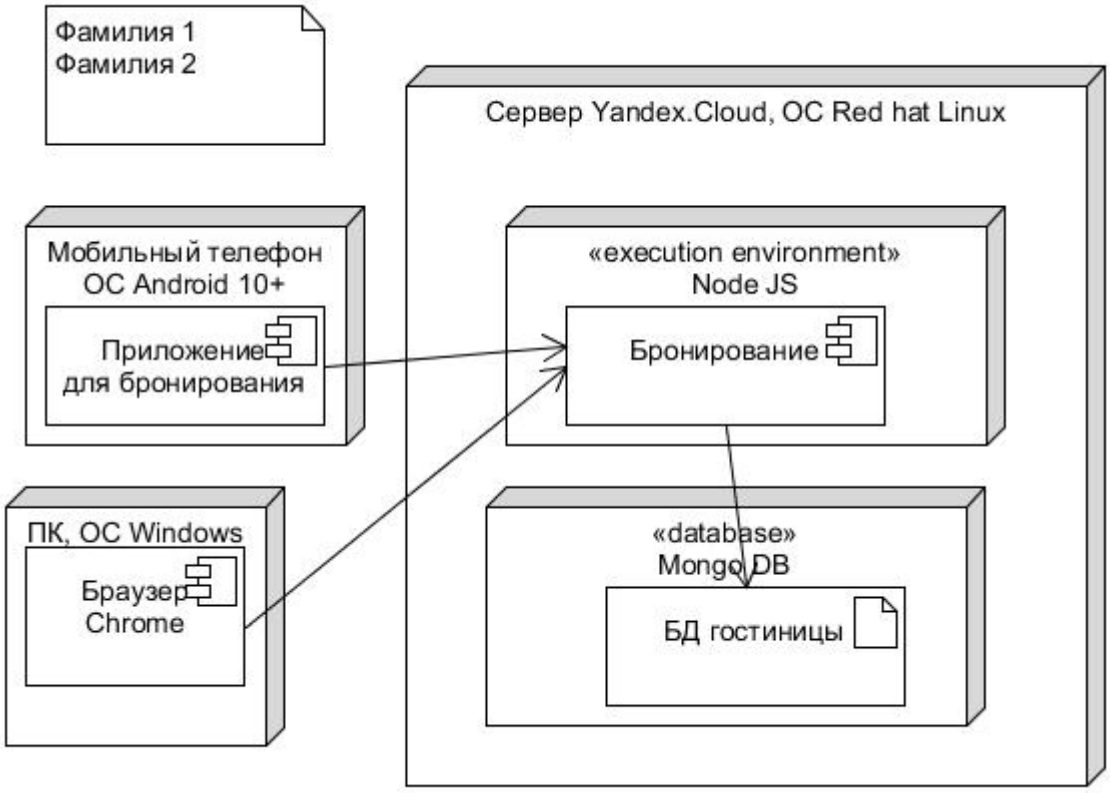
20. Проектирование системы футбольных ставок.

21. Проектирование системы бронирования столиков и заказа блюд меню ресторана по интернету.

22. Проектирование системы для грузоперевозок.

Нарисовать диаграмму компонентов/размещения UML

Фамилии разработчиков и номер варианта написать в элементе «комментарий»



№6 Лабораторная

Задание выполняется в группе из 2 человек
Пример выполнения:

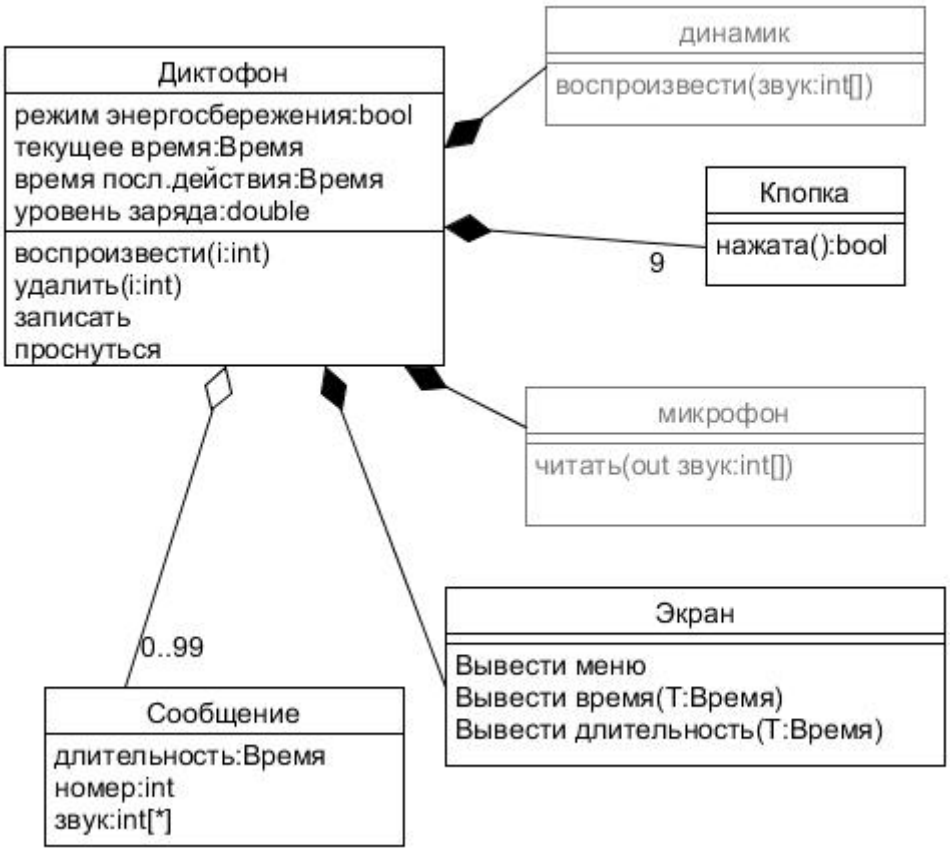
Цифровой диктофон – это бытовое электронное устройство, предназначенное для записи и воспроизведения речи. Звуковые сообщения записываются через встроенный микрофон и сохраняются в памяти устройства. Сообщения воспроизводятся через встроенный громкоговоритель.
Диктофон хранит до 99 звуковых сообщений. Длина каждого сообщения ограничена размером свободной памяти. Диктофон осуществляет прямой (по номеру сообщения) доступ к любому сообщению из

памяти. Пользователь имеет возможность воспроизводить сообщения, хранящиеся в памяти диктофона, стирать их, записывать новые. Интерфейс с пользователем осуществляется при помощи экранного меню и управляющих кнопок на корпусе диктофона. При помощи 4 кнопок-стрелок осуществляется навигация по пунктам меню. Кнопки «OK», «Отмена» служат для подтверждения или отмены пользователем выбора той или иной опции меню (структуру меню нужно разработать самостоятельно). Имеются также кнопки «Воспроизведение», «Пауза» и «Запись» для работы со звуковыми сообщениями.

Во время записи сообщения на экране отображается время, в течение которого ведется запись, при воспроизведении – длительность воспроизведенной части сообщения.

Если диктофон не используется, через 30 секунд он автоматически переходит в режим сбережения энергии. В этом режиме никакие операции над звуковыми сообщениями не возможны. Энергия расходуется только на сохранение памяти диктофона в неизменном состоянии. Переход из режима сбережения энергии в обычный режим осуществляется при нажатии пользователем любой кнопки.

В диктофоне имеется датчик уровня заряда батарей. При падении уровня заряда ниже установленного предела диктофон автоматически переходит в режим сбережения энергии. Переход в обычный режим становится возможным только после восстановления нормального уровня заряда батарей.



Вариант 1

Домофон регулирует доступ в подъезд многоквартирного дома. В подъезде

имеется дверь с замком. С наружной стороны двери установлена внешняя панель домофона, на которой находятся кнопки для связи с каждой квартирой, микрофон и динамик. В каждой квартире находится внутренняя панель домофона с кнопками: «СВЯЗЬ», «БЛОКИРОВКА» и «ОТКРЫТЬ». Кроме того, на внутренней панели имеется микрофон и динамик.

Жильцы могут открывать дверь ключом. Посетитель может нажать кнопку квартиры на внешней панели. При этом в квартире раздается звонок (если подача звонка в квартиру не заблокирована). Услышав звонок, жилец квартиры нажимает на кнопку «СВЯЗЬ» внутренней панели домофона, после чего домофон устанавливает звуковое сообщение между жильцом и посетителем. Звуки, произносимые посетителем в микрофон, установленный на внешней панели, воспроизводятся в динамике, установленном в квартире. Звуки из микрофона в квартире, передаются в динамик на внешней панели. После сеанса связи жилец может нажать на кнопку «ОТКРЫТЬ», чтобы замок на двери в подъезд открылся, и посетитель смог войти. По истечении минуты замок должен снова заблокировать вход в подъезд.

Жилец, который желает, чтобы его не беспокоили, может отключить подачу звонка в свою квартиру, нажав на кнопку «БЛОКИРОВКА». Повторное нажатие на эту кнопку вновь включает подачу звонка.

Вариант 2

В торговом автомате имеется пять лотков для хранения и выдачи товаров. Загрузка товаров на лотки осуществляется обслуживающим персоналом. Автомат следит за наличием товара. Если какой-либо товар распродан, автомат отправляет сообщение об этом на станцию обслуживания и информирует покупателей (зажигается красная лампочка рядом с лотком данного товара). Автомат принимает к оплате бумажные купюры и монеты. Специальный индикатор высвечивает текущую сумму денег, принятых автоматом к оплате. После ввода денег клиент нажимает на кнопку выдачи товара. Выдача товара производится только в том случае, если введенная сумма денег точно соответствует цене товара. Товар выдается поштучно.

При нажатии на кнопку «Возврат» клиенту возвращаются все принятые от него к оплате деньги. Возврат денег не производится после выдачи товара.

Автомат получает со станции обслуживания данные о товарах и хранит их в своей памяти. Данные включают в себя цену, наименование товара, номер лотка, на котором находится товар и количество товара на лотке.

Вариант 3

Табло расположены на каждой станции метро. Они работают под управлением единого пункта управления (ПУ) информационной службы метро. Табло отображает текущее время (часы, минуты, секунды) и время, прошедшее с момента отправления последнего поезда (минуты, секунды). Момент прибытия и отправления поезда определяется при помощи датчиков, устанавливаемых на путях. Все табло метро синхронизованы, текущее время отсчитывается и устанавливается из центральной службы времени, находящейся на ПУ.

На табло высвечивается конечная станция назначения прибывающего поезда. Эти данные содержатся в расписании движения поездов, которое хранится в памяти табло и периодически обновляется с ПУ.
В «бегущей строке» табло отображается рекламная информация. Память табло хранит до 10 рекламных сообщений. Сообщения отображаются друг за другом с небольшими паузами, циклически. Содержание рекламных сообщений поступает с ПУ.
Дополнительная функция табло – по запросу с ПУ оно пересылает данные о нарушениях расписания (преждевременных отправлениях поездов или опозданиях).
Пояснение: в задании требуется разработать модель ПО только для табло, но не для пункта управления информационной службы.

Вариант 4

Пункт проката содержит каталог кассет, имеющихся в наличии в данный момент времени. Система поддерживает работу каталога, позволяя служащим проката добавлять новые наименования кассет, удалять старые и редактировать данные о кассетах.
Клиент, обратившийся в пункт, выбирает кассету по каталогу, вносит залог и забирает ее на определенный срок. Срок проката, измеряемый в сутках, оговаривается при выдаче кассеты. Стоимость проката вычисляется системой исходя из тарифа за сутки и срока проката. Клиент возвращает кассету и оплачивает прокат. Если кассета не повреждена, клиенту возвращается залог. Служащий пункта проката регистрирует сдачу кассеты клиенту и ее возврат в системе. Если клиент повредил кассету, то кассета удаляется из каталога, а залог остается в кассе проката.
При необходимости служащий может запросить у системы следующие данные:
– имеется ли в наличии кассета с данным названием;
– когда будет возвращена какая-либо кассета из тех, что сданы в прокат;
– является ли данный клиент постоянным клиентом пункта проката (пользовался ли прокатом 5 или более раз).
Постоянным клиентам предоставляются скидки, а также от них принимаются заявки на пополнение ассортимента кассет. Заявки регистрируются в системе. По ним готовится итоговый отчет, руководствуясь которым, служащие пункта проката обновляют ассортимент кассет.

Вариант 5

Стиральная машина предназначена для автоматической стирки белья. Машина включает в себя следующие устройства: бак для белья, клапаны для забора и слива воды, мотор, устройство подогрева воды, термометр, таймер, дверца для доступа в бак, несколько емкостей для различных моющих средств, панель управления с кнопками и индикатором. В памяти машины хранятся 5 программ стирки, заданные изготовителем. Пользователи не могут вносить в них изменения. Каждая программа определяет температуру воды, длительность стирки, используемые моющие средства (номер емкости и время подачи), скорость вращения бака во время стирки и отжима.

Для использования машины необходимо открыть дверцу, поместить белье в бак, поместить моющие средства в емкости, закрыть дверцу, выбрать программу стирки и нажать на кнопку «Пуск». Перед тем как приступить к стирке машина открывает клапан для забора воды, набирает необходимое количество воды, после чего закрывает клапан. Далее, машина действует по выбранной пользователем программе:
1) Подогревает, если необходимо, воду до нужной температуры.
2) Включает таймер и запускает вращение бака для стирки.
3) По таймеру подает в бак моющие средства, предусмотренные программой.
4) По окончании стирки сливает воду и запускает отжим.
Во время работы машины на индикаторе высвечивается время, прошедшее с момента запуска (минуты и секунды), текущий режим работы (стирка или отжим), номер текущей программы стирки. В целях безопасности дверца бака блокируется до окончания стирки. Машина не воспринимает нажатий на кнопки, за исключением одной – пользователь имеет возможность в любой момент нажать на кнопку «Останов», чтобы принудительно остановить стирку и слить воду.

Вариант 6

Холодильник состоит из нескольких холодильных камер для хранения продуктов. В каждой холодильной камере имеется регулятор температуры, мотор, термометр, индикатор, таймер, датчик открытия двери камеры и устройство для подачи звуковых сигналов.
При помощи терморегулятора устанавливается максимально допустимая температура в данной камере. Мотор предназначен для поддержания низкой температуры. Термометр постоянно измеряет температуру внутри камеры, а индикатор температуры, расположенный на дверце, постоянно высвечивает ее значение. При повышении температуры выше предела, определяемого текущим положением регулятора, включается мотор. При снижении температуры ниже некоторого другого значения, связанного с первым, мотор отключается.
Доступ в камеру осуществляется через дверцу. Если дверь холодильной камеры открыта в течение слишком долгого времени, подается звуковой сигнал. Звуковой сигнал также подается в любых нештатных ситуациях (например, при поломке мотора).
Холодильник ведет электронный журнал, в котором отмечаются все происходящие события:
– изменение положения терморегулятора камеры;
– включение и отключение мотора;
– доступ в камеру;
– внештатные ситуации.
Вариантом задания предусмотрена разработка схемы базы данных для хранения журнала событий холодильника. Содержимое журнала может быть передано в компьютер, подсоединенный к специальному гнезду на корпусе холодильника.

Вариант 7

В бакалейной лавке для каждого товара фиксируется место хранения (определенная полка), количество товара и его поставщик. Система поддержки заказа и учета товаров должна обеспечивать добавление

информации о новом товаре, изменение или удаление информации об имеющемся товаре, хранение (добавление, изменение и удаление) информации о поставщиках, включающей в себя название фирмы, ее адрес и телефон. При помощи системы составляются заказы поставщикам. Каждый заказ может содержать несколько позиций, в каждой позиции указываются наименование товара и его количество в заказе. Система учета по требованию пользователя формирует и выдает на печать следующую справочную информацию:
– список всех товаров;
– список товаров, имеющихся в наличии;
– список товаров, количество которых необходимо пополнить;
– список товаров, поставляемых данным поставщиком.

Вариант 8

Система поддержки управления библиотекой должна обеспечивать операции (добавление, удаление и изменение) над данными о читателях.
В регистрационном списке читателей хранятся следующие сведения: фамилия, имя и отчество читателя; номер его читательского билета и дата выдачи билета. Наряду с регистрационным списком системой должен поддерживаться каталог библиотеки, где хранится информация о книгах: название, список авторов, библиотечный шифр, год и место издания, название издательства, общее количество экземпляров книги в библиотеке и количество экземпляров, доступных в текущий момент. Система обеспечивает добавление, удаление и изменение данных каталога, а также поиск книг в каталоге на основании введенного шифра или названия книги. В системе осуществляется регистрация взятых и возвращенных читателем книг. Про каждую выданную книгу хранится запись о том, кому и когда была выдана книга, и когда она будет возвращена. При возврате книги в записи делается соответствующая пометка, а сама запись не удаляется из системы. Система должна выдавать следующую справочную информацию:
– какие книги были выданы за данный промежуток времени;
– какие книги были возвращены за данный промежуток времени;
– какие книги находятся у данного читателя;
– имеется ли в наличии некоторая книга.

Вариант 9

Интернет-магазин позволяет делать покупки с доставкой на дом. Клиенты магазина при помощи программы-браузера имеют доступ к каталогу продаваемых товаров, поддержку которого осуществляет Интернет-магазин. В каталоге товары распределены по разделам. О каждом товаре доступна полная информация (название, вес, цена, изображение, дата изготовления и срок годности) Для удобства клиентов предусмотрена система поиска товаров в каталоге. Заполнение каталога информацией происходит автоматически в начале рабочего дня, информация берется из системы автоматизации торговли.
При отборе клиентами товаров поддерживается виртуальная «торговая корзина». Любое наименование товара может быть добавлено в «корзину» или изъято в любой момент по желанию покупателя с последующим пересчетом общей стоимости покупки. Текущее содержимое «корзины» постоянно показывается клиенту.

По окончании выбора товаров производится оформление заказа и регистрация покупателя. Клиент указывает в регистрационной форме свою фамилию, имя и отчество, адрес доставки заказа и телефон, по которому с ним можно связаться для подтверждения сделанного заказа. Заказы передаются для обработки в систему автоматизации торговли. Проверка наличия товаров на складе и их резервирование Интернет-магазином не производятся.

Вариант 10

WWW-конференция представляет собой хранилище сообщений в сети Интернет, доступ к которому осуществляется при помощи браузера. Для каждого сообщения конференции хранятся значения следующих полей: номер сообщения, автор, тема, текст сообщения, дата добавления сообщения, ссылка на родительское сообщение. Начальной страницей конференции является иерархический список сообщений. Верхний уровень иерархии составляют сообщения, открывающие новые темы, а подуровни составляют сообщения, полученные в ответ на сообщения верхнего уровня. Сообщение-ответ всегда имеет ссылку на исходное сообщение.

В списке отображаются только темы сообщений, их авторы и даты добавления. Просматривая список, пользователь выбирает сообщение и по гиперссылке открывает страницу с текстом сообщения. Помимо текста на этой странице отображается список (иерархический) сообщений являющихся ответами, ответами на ответы и т.д. Для удобства пользователей необходимо предусмотреть поиск сообщений по автору или по ключевым словам в теме или тексте сообщения.

Сообщения добавляются в конференцию зарегистрированными пользователями, которые при отправке сообщения должны указать своё имя и пароль. Регистрирует новых пользователей модератор конференции – её ведущий. При регистрации пользователь заполняет специальную форму, содержимое которой затем пересылается модератору и запоминается в базе пользователей. Модератор решает, регистрировать пользователя или нет, и отправляет свой ответ.

При добавлении сообщений пользователь имеет возможность начать новую тему или ответить на ранее добавленные сообщения. После добавления сообщения оно доступно для чтения всем пользователям (даже незарегистрированным), и список сообщений обновляется.

Модератор имеет право по тем или иным причинам удалять сообщения любых авторов. Он также может наказывать пользователей, нарушающих правила поведения в конференции, лишая на некоторое время пользователя возможности добавлять и редактировать сообщения.

Вариант 11

В каталоге хранится следующая информация о ресурсах: название ресурса, уникальный локатор ресурса (URL), раздел каталога, в котором содержится ресурс, список ключевых слов, краткое описание, дата последнего обновления, контактная информация.

Доступ пользователей к каталогу осуществляется при помощи браузера. Пользователи каталога могут добавлять новые ресурсы, информация о которых не была внесена ранее. Ресурсы в каталоге классифицируются по разделам. Полный список ресурсов каждого

раздела должен быть доступен пользователям. Пользователям каталога должны быть предоставлены возможности по поиску ресурсов. Поиск осуществляется по ключевым словам. Если пользователь не доволен результатами поиска, он может уточнить запрос (осуществить поиск среди результатов предыдущего поиска). Должна быть возможность выдавать результаты поиска в разной форме (вывод всей информации о ресурсах или частичной). Пользователь может отсортировать список ресурсов по релевантности (соответствию ключевым словам из запроса) или по дате обновления.
Поскольку содержание ресурсов Интернет со временем изменяется необходимо следить за датой последнего обновления, периодически опрашивая Web-сайты, URL которых хранятся в каталоге.

Вариант 12

Системы для поддержки генеалогических деревьев хранит сведения о персонах (Ф.И.О., пол, дата рождения, дата смерти, биография) и о родственных связях между ними. Связи бывают только трех видов: «мужья-жены», «дети-родители» и «братья-сестры». Система обеспечивает возможность добавления данных о новых персонах и родственных связях, изменение введенных данных и удаление ненужных данных.
Система следит за непротиворечивостью вводимых данных. Например, недопустимо, чтобы человек был собственным предком или потомком.
Пользователи системы могут осуществлять поиск полезной информации по дереву:
– находить для указанного члена семьи его детей;
– находить для указанного члена семьи его родителей;
– находить для указанной персоны братьев и сестер, если таковые есть;
– получать список всех предков персоны;
– получать список всех потомков персоны;
– получать список всех родственников персоны;
– прослеживать цепочку родственных связей от одной персоны до другой (например, если Петр является шурином Ивана, то на запрос о родственных связях между Петром и Иваном выдается такой результат: «Петр – брат Ольги, Ольга – жена Ивана»).

Вариант 13

Система обеспечивает составление расписания некоторого учебного заведения, внесение в расписание изменений, выдачу полного расписания и дополнительной информации (например, по итоговому расписанию составляется расписание указанной группы на заданный день или неделю).
В расписании фиксируются время и место проведения занятия, предмет и преподаватель, проводящий занятие, а также номер группы, для которой это занятие проводится. Расписание не должно содержать коллизий (например, разные занятия не должны пересекаться друг с другом по месту и времени их проведения, один преподаватель не может вести одновременно два разных занятия, в одно и то же время у одной и той же группы не может быть два различных занятия и т. д.).

№7 Лабораторная

Задание выполняется в группе из 2 человек

Вариант 1

Выбор тарифного плана данного оператора сотовой связи

Вариант 2

Выбор специальности обучения в университете

Вариант 3

Выбор блюд в ресторане

Вариант 4

Рекомендательная система для фильмов

№8 Лабораторная

Задание выполняется индивидуально

Разработать функцию, согласно выбранному варианту, методом TDD. История разработки кода сохраняется после добавления каждого теста в папке history как файл solve#.cpp

Начальные файлы

solve.hpp:

```
int solve(параметры);
```

solve.cpp:

```
#include "solve.hpp"
int solve(параметры) {
// решение
}
```

solve_unittest.cpp:

```
#include "solve.hpp"
#include <gtest/gtest.h>

TEST(TestSet, Test1) {
  EXPECT_EQ(solve(аргументы1), результат1);
  EXPECT_EQ(solve(аргументы2), результат2);
}
```

Для проверки полноты тестов можно отправить решение на сайт ipc.susu.ru, добавив main

```
int main() {
  cin>>параметры;
  cout<<solve(параметры)<<"\n";
}
```

Варианты заданий:

Вариант 1 (1999)

На дороге, ведущей из Киева в Чернигов, находится камень. Функции предаются четыре целых числа — расстояние от села Калиновки до Киева A, расстояние от реки Смородины до Киева B ($1 \le A < B \le 1000$), расстояние от камня до села Калиновки X и расстояние от камня до реки Смородины Y ($1 \le X, Y \le 1000$). Гарантируется, что числа A, B, X, Y соответствуют какому-то возможному расположению камня на дороге.

Функция возвращает одно целое число — расстояние от камня с надписью до Киева. При нарушении ограничений функция порождает исключительную ситуацию.

Вариант 2 (2318)

Петру необходимо попасть с этажа A на этаж B. Для вызова лифта на всех этажах офисного здания, кроме первого и последнего, есть две кнопки – для перемещения вниз и перемещения вверх. В тот момент, когда Петр нажал нужную кнопку вызова, лифт находился на этаже C и вез одного пассажира на этаж D. Если лифт проезжает мимо этажа, на котором нажата кнопка вызова, и лифт движется в подходящем направлении, то лифт останавливается, чтобы посадить дополнительного пассажира. Лифт перемещается между соседними этажами за одну единицу времени, также одну единицу времени занимает остановка лифта на этаже для высадки или посадки пассажиров.

Напишите функцию, вычисляющую, через сколько времени Петр доберется до этажа B, при условии, что никто больше не будет вызвать лифт.

Параметрами функции являются четыре целых числа A, B, C и D, разделенных одним пробелом ($1 \le A, B, C, D \le 20$, $A \ne B$, $C \ne D$, $A \ne C$). При нарушении ограничений функция порождает исключительную ситуацию.

Вариант 3 (1985)

Есть четыре рейки, из которых он хочет сделать прямоугольную рамку. При этом не обязательно, чтобы все рейки были использованы полностью или частично. Можно разрезать любую рейку на 2 или более частей, но **нельзя** соединять рейки или их части для использования полученной склейки в качестве какой-либо стороны рамки. Необходимо, чтобы рамка имела наибольший периметр.

Напишите функцию, вычисляющую максимальный периметр прямоугольной рамки, которую можно сделать из четырех реек заданных размеров.

Параметрами функции являются четыре целых числа a, b, c, d в неубывающем порядке ($1 \le a \le b \le c \le d \le 1000$) — размеры реек. При нарушении ограничений функция порождает исключительную ситуацию.

Вариант 4 (1596)

Скоро у мальчика Пети будет день рождения. Петя коллекционирует почтовые марки, поэтому его друзья решили подарить ему на день рождения A марок. В местном почтовом отделении марки продаются только в упаковках. Каждая упаковка содержит B марок и стоит C рублей. Какую минимальную сумму денег необходимо иметь друзьям Пети, чтобы сделать подарок из A марок?

Примечания:

- Друзьям Пети не обязательно покупать ровно A марок. Они могут приобрести большее количество марок, и часть из них оставить себе.

- Число A может быть равно нулю. Это означает, что друзья Пети решили не дарить Пете ни одной марки.
- Число B может быть равно нулю. Это означает, что в упаковках марок, которые продаются на почте, на самом деле нет ни одной марки.
- Число C может быть равно нулю. Это означает, что упаковки с марками выдаются на почте бесплатно.

Параметрами функции являются три целых числа A, B и C ($0 \le A, B, C \le 10000$).
Функция возвращает целое число, равное минимальной сумме денег в рублях, которую необходимо иметь друзьям Пети, чтобы сделать подарок из A марок. В случае, когда сделать подарок из A марок невозможно и нарушении ограничений, функция порождает исключительную ситуацию.

Вариант 5 (2375)

Том покрасил часть забора, начиная с доски под номером A до доски под номером B включительно. Затем Том взял новое ведро с извёсткой и покрасил часть забора, начиная с доски под номером C до доски под номером D включительно, при этом часть забора могла быть покрашена дважды. Кроме того Том мог красить доски забора как справа налево, так и слева направо.
Напишите функцию, вычисляющую общее количество покрашенных досок забора.
Параметрами функции являются четыре целых числа A, B, C и D ($1 \le A, B \le 10^9$, $1 \le C, D \le 10^9$) – диапазоны номеров досок забора, покрашенных Томом. При нарушении ограничений функция порождает исключительную ситуацию.

Вариант 6 (2429)

В магазине продаются два вида печенья. Первый вид печенья упакован в коробки по A штук и стоит B центов за коробку, второй вид печенья упакован в коробки по C штук и стоит D центов за коробку. Аня собирается угостить печеньем Nгостей и хочет приобрести столько коробок печенья одного вида, чтобы каждому гостю досталось по одному печенью. Например, для 22 гостей можно купить либо 3 коробки за 11 центов по 10 печений, либо 2 коробки за 15 центов по 12 печений. В первом случае Аня потратит 33 цента, во втором случае — 30 центов.
Напишите функцию, определяющую, какой вид печенья выгоднее купить.
Параметрами функции являются пять целых чисел A, B, C, D и N ($1 \le A, B, C, D, N \le 10^9$)
— информация о количестве печенья в коробке и стоимости для каждого вида печенья и количество гостей.
Функция возвращает пару из строки и целого числа. Первый элемент пары "`FIRST`", если выгоднее купить печенье первого вида, или сообщение "`SECOND`", если выгоднее купить печенье второго вида, или сообщение "`ANY`", если стоимость приобретения N или более штук печенья для обоих видов одинакова. Первый элемент пары — стоимость покупки. При нарушении ограничений функция порождает исключительную ситуацию.

№9 Лабораторная

Задание выполняется индивидуально

Выполняется в MinIDE или Visual Studio C++ 2017 Enterprise Edition
Результат функции в этой лабораторной работе игнорируется, но на практике покрытие кода используется для оценки качества тестов, разработанных в ходе TDD

Вариант 1

```
#include "gtest/gtest.h"
#include <string>
using namespace std;
void minroman(const char st[], char r[]);
TEST(MINROMAN,WHITE_BOX)
{ char res[100];
  minroman("IVXCL",res);
}

#include <cstring>
using namespace std;
void minroman(const char st[], char s[]) {

      int p[ 255 ] = { 0 };
    for( int i = 0; st[ i ] != 0; ++i )
       p[ ( int )st[ i ] ]++;
    s[0]=0;
    if( p[ 'C' ] == 1 ) {
           strcat(s,"XC");
           p[ 'C' ]--;
           p[ 'X' ]--;
       }
    if( p[ 'L' ] == 1 ) {
       if( p[ 'X' ] == 1 || ( p[ 'X' ] == 2 && p[ 'I' ] == 1 && p[ 'V' ]
== 0 ) )
       {
                strcat(s,"X");
                p[ 'X' ]--;
              }
              strcat(s,"L");
              p[ 'L' ]--;
    }

    while( p[ 'X' ] > 1 )
       { strcat(s,"X");
         p[ 'X' ]--;
       }
    if( p[ 'X' ] == 1 && p[ 'V' ] == 0 && p[ 'I' ] == 1 )
    { strcat(s,"IX");
         p[ 'X' ]--;
         p[ 'I' ]--;
       }
    if( p[ 'X' ] == 1 )
       { strcat(s,"X");
         p[ 'X' ]--;
       }

    if( p[ 'I' ] == 1 && p[ 'V' ] == 1 )
       { strcat(s,"I");
         p[ 'I' ]--;
       }
    if( p[ 'V' ] == 1 )
       { strcat(s,"V");
         p[ 'V' ]--;
       }


    while( p[ 'I' ] > 0 )
       { strcat(s,"I");
```

```
        p[ 'I' ]--;
      }

}
```

Вариант 2

```
#include "gtest/gtest.h"
bool check(int day, int month, int year);
TEST(CHECK,WHITE_BOX)
{ check(1,1,1);
}

int const days_count[12] = {31,28,31,30,31,30,31,31,30,31,30,31};

int is_vis(int year) {
    return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
}

bool check(int day, int month, int year) {
    if (day > 31 || day <= 0)
        return 0;

    if (month > 12 || month <= 0)
        return 0;

    if (month == 2) {
        if (is_vis(2000 + year + ((year == 0) ? (100) : (0)) )) {
            if (day > 29)
                return 0;


            return 1;
        }
        else {
            if (day > 28)
                return 0;


            return 1;
        }
    }
    else {
        if (day > days_count[month-1])
            return 0;


        return 1;
    }
}
```

№10 Лабораторная

Задание выполняется индивидуально

Проверка разработанных тестов выполняется на реальных программах, содержащих ошибки, исходный код которых недоступен.

Необходимо написать набор тестов, используя методы тестирования программы как черного ящика (эквивалентное разбиение, граничные значения, предположение об ошибке), который выявляет ошибки во всех программах или большей их части.
Так как программы разрабатывались под определенные ограничения, то ошибочные классы эквивалентности не должны использоваться для тестирования.

Подзадача 1 (для всех вариантов)

Напишите тесты для следующей задачи:

Гном Ворчун родился 29 февраля, поэтому свой день рождения он отмечает только в високосном году. Год является високосным, если он кратен 4 и при этом не кратен 100, либо кратен 400. Год не является високосным, если он не кратен 4 либо кратен 4, но при этом кратен 100 и не кратен 400.

Напишите программу, которая поможет Ворчуну определить, получит ли он подарки в заданном году.

Ввод содержит одно целое число N ($1 \le N < 10^9$) – номер года.

Вывести сообщение "`Yes`", если в год N подарки будут, или сообщение "`No`", в противном случае.

Пример ввода

```
2007
```

Пример вывода

```
No
```

Тесты задаются в форме:

```
*INPUT
Ввод для теста 1
*OUTPUT
Вывод для теста 1
*INPUT
Ввод для теста 2
*OUTPUT
Вывод для теста 2
...
```

Ввод для тестов должен соответствовать условиям задачи. Решение считается принятым, если тесты обнаруживают ошибки во всех программах из некоторого набора. Количество тестов не должно превышать 20. Тесты после 20-го игнорируются. Ошибка PE при проверке означает, что выведенные тесты содержат ошибки, т.е. не соответствуют условиям задачи или ввод выводу.

Пример вывода

```
*INPUT
2007
*OUTPUT
No
*INPUT
2008
*OUTPUT
Yes
```

Подзадача 2 вариант 1

Напишите тесты для следующей задачи:
Напишите программу, выполняющую превращение набора английских слов в идентификатор по следующим правилам.

- Если слово содержит три или менее буквы, то оно полностью включается в идентификатор.
- Иначе удаляются все гласные буквы в слове, кроме гласной, являющейся первой буквой в слове. В идентификатор включаются три первых буквы результата (если осталось меньше трех, то все оставшиеся). К гласным относятся буквы a, e, i, o, u, y.
- Первая буква всех слов, начиная со второго, становится прописной.
- Порядок слов не меняется.

В первой строке ввода содержится непустой набор слов, разделенных пробелами. Длина строки не превышает 100 символов. Слова содержат только строчные латинские буквы.
Вывести получившийся идентификатор.

Пример ввода

```
list of integer numbers
```

Вывод для примера

```
lstOfIntNmb
```

Тесты задаются в форме:

```
*INPUT
Ввод для теста 1
*OUTPUT
Вывод для теста 1
*INPUT
Ввод для теста 2
*OUTPUT
Вывод для теста 2
...
```

Ввод для тестов должен соответствовать условиям задачи. Решение считается принятым, если тесты обнаруживают ошибки во всех программах из некоторого набора. Количество тестов не должно превышать 20. Тесты после 20-го игнорируются. Ошибка PE при проверке означает, что выведенные тесты содержат ошибки, т.е. не соответствуют условиям задачи или ввод выводу.

Пример теста

```
*INPUT
list of integer numbers
*OUTPUT
lstOfIntNmb
*INPUT
ident
*OUTPUT
idn
```

Подзадача 2 вариант 2

Напишите тесты для следующей задачи:
Найти наименьшее целое число Y, большее или равное заданному числу X, запись которого в некоторой системе счисления по основанию B ($2 \le B \le 36$) состоит из одинаковых цифр.

Например, для числа 1234 таким числом будет 1256, которое в 12-ричной системе можно записать, используя только одну цифру 8: 88812.
В первой строке входного файла содержится целое X ($1 \le X \le 10^9$) в десятичной системе счисления.
В выходной файл вывести основание системы счисления B и через пробел запись искомого числа Y в этой системе счисления. Если существует несколько вариантов искомого Y
в системах счисления с различными основаниями, то вывести результат, используя наименьшее из оснований. Для цифр от 10 до 35 использовать прописные латинские буквы A..Z соответственно.

Пример ввода

```
1234
```

Пример вывода

```
12 888
```

Тесты задаются в форме:

```
*INPUT
Ввод для теста 1
*OUTPUT
Вывод для теста 1
*INPUT
Ввод для теста 2
*OUTPUT
Вывод для теста 2
...
```

Ввод для тестов должен соответствовать условиям задачи. Решение считается принятым, если тесты обнаруживают ошибки во всех программах из некоторого набора. Количество тестов не должно превышать 20. Тесты после 20-го игнорируются. Ошибка PE при проверке означает, что выведенные тесты содержат ошибки, т.е. не соответствуют условиям задачи или ввод выводу.
Подсказка: Определите сначала для каждой системы счисления какие десятичные числа в ней можно записать с помощью одинаковых цифр. Всего таких чисел несколько десятков.

Пример вывода

```
*INPUT
1234
*OUTPUT
12 888
*INPUT
2
*OUTPUT
3 2
```

Подзадача 2 вариант 3

Напишите тесты для следующей задачи:

Гном Чихун получил свое прозвище из-за частого чиханья. Иногда он может чихнуть 2-3 раза посреди какого-нибудь длинного слова гномьего языка. Если чих получится в середине слога, то слово станет непонятным и его придется произносить заново. Поэтому для Чихуна важно знать, как слово делится на отдельные слоги. Основой слога в гномьем языке является непрерывная последовательность гласных букв, к которой слева и справа может примыкать 0 или более согласных. Если слог не является первым слогом слова, то к нему относится только согласная, стоящая непосредственно перед группой гласных. Остальные согласные перед слогом относятся

к предыдущему слогу слова. Гласными буквами являются буквы 'a', 'e', 'i', 'o' и 'u', согласными – все остальные. Буква 'h' не является самостоятельной согласной, она указывает на приглушение согласной, стоящей перед ней, или на то, что гласные произносятся раздельно (относятся к разным слогам). Буква 'h' не может стоять перед 'h', но может быть первой буквой слова. (На самом деле в гномьем языке гораздо больше букв, чем в латинском алфавите, но их нет даже в кодировке Unicode, поэтому задачу придется решать для транскрипции гномьих слов латинскими буквами).
Напишите программу, которая найдет для заданного слова места, где Чихун может чихнуть.
Ввод содержит одно слово длиной не более 50 букв, состоящее из строчных латинских букв. Слово содержит как минимум одну гласную букву.
Вывести слово из входного файла, указав места разбиения слова на слоги с помощью символа '–'.

Пример ввода 1

```
bundshaatur
```

Пример вывода 1

```
bund-shaa-tur
```

Пример ввода 2

```
aha
```

Пример вывода 2

```
a-ha
```

Вывод программы должен содержать тесты в форме:

```
*INPUT
Ввод для теста 1
*OUTPUT
Вывод для теста 1
*INPUT
Ввод для теста 2
*OUTPUT
Вывод для теста 2
...
```

Ввод для тестов должен соответствовать условиям задачи. Решение считается принятым, если тесты обнаруживают ошибки во всех программах из некоторого набора. Количество тестов не должно превышать 20. Тесты после 20-го игнорируются. Ошибка PE при проверке означает, что выведенные тесты содержат ошибки, т.е. не соответствуют условиям задачи или ввод выводу.

Пример вывода

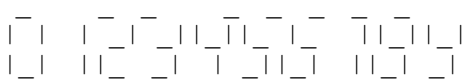
```
*INPUT
bundshaatur
*OUTPUT
bund-shaa-tur
*INPUT
aha
*OUTPUT
a-ha
```

Подзадача 2 вариант 4

Напишите тесты для следующей задачи:

Невилл высыпал спички на стол, и они сложились в число, цифры которого не сулили ничего хорошего. Чтобы изменить прогноз, Невиллу нужно превратить получившееся число в другое, переложив всего одну спичку. Новое число должно быть самым большим из возможных, чтобы максимально снизить риск неблагоприятного события.

Цифры из волшебных спичек имеют вид как на калькуляторах муглов:



Напишите программу, которая вводит число, сложившееся из спичек, и выводит число, получившееся после перекладывания одной спички.

В первой строке входного файла содержится одно целое число N ($1 \le N < 10^9$).

В выходной файл вывести новое число K ($K > 0$, $K \ne N$, в случае нескольких вариантов – наибольшее из них) или сообщение "FAIL", если изменить число перекладыванием одной спички невозможно.

Пример ввода

```
194
```

Вывод для примера

```
754
```

Тесты задаются в форме:

```
*INPUT
Ввод для теста 1
*OUTPUT
Вывод для теста 1
*INPUT
Ввод для теста 2
*OUTPUT
Вывод для теста 2
...
```

Ввод для тестов должен соответствовать условиям задачи. Решение считается принятым, если тесты обнаруживают ошибки во всех программах из некоторого набора. Количество тестов не должно превышать 50. Тесты после 50-го игнорируются. Ошибка PE при проверке означает, что выведенные тесты содержат ошибки, т.е. не соответствуют условиям задачи или ввод выводу.

Пример вывода

```
*INPUT
194
*OUTPUT
754
*INPUT
8
*OUTPUT
FAIL
```

№11 Лабораторная

Задание выполняется индивидуально

Имеется тест, выявляющий ошибку в программе. Используя отладчик и/или логирование определите ошибочный оператор. Напишите причину ошибку, не исправляя эти операторы.

Вариант 1 (144)

```
#include<iostream>
using namespace std;
int main()
{
  int n,k,a,c,kol[3]={0};
  char s[3][20100];
  cin>>n>>k;
  for(int i=0;i<k;++i)
  {
    cin>>a>>c;
    if(c==8)
    {
      if(kol[a-1]!=0)
      {
    kol[a-1]--;
      }
    }
    else if(c==13)
    {
      s[a-1][kol[a-1]++]='/';
      s[a-1][kol[a-1]++]='/';
    }
    else
    {
      s[a-1][kol[a-1]++]=c;
    }
  }
  for(int i=0;i<n;++i)
  {
    s[i][kol[i]]='\0';
    cout<<s[i]<<endl;
  }
  return 0;
}
```

Тест, выявивший ошибку

```
2 7
2 48
2 13
2 49
2 8
2 8
2 8
2 50
```

Ожидаемый ответ

```
0//2
```

Вариант 2 (144)

```
#include<iostream>
using namespace std;

  int n,k,a,c,kol[100]={0};
  char s[100][20100];

int main()
{

  cin>>n>>k;
  for(int i=0;i<k;++i)
  {
    cin>>a>>c;
    if(c==8)
    {
      if(kol[a-1]!=0 && s[a-1][kol[a-1]-1]!='/')
      {
    kol[a-1]--;
      }
    }
    else if(c==13)
    {
      s[a-1][kol[a-1]++]='/';
      s[a-1][kol[a-1]++]='/';
    }
    else
    {
      s[a-1][kol[a-1]++]=c;
    }
  }
  for(int i=0;i<n;++i)
  {
    s[i][kol[i]]='\0';

    cout<<s[i]<<endl;
  }
  return 0;
}
```

Тест, выявивший ошибку

```
1 8
1 65
1 8
1 66
1 13
1 47
1 8
1 8
1 68
```

Ожидаемый ответ

B//D

Вариант 3 (1190)

```
#include<iostream>
#include <vector>
#include <set>
#include <map>
#include <cstring>
#include <cstdlib>
#include <cstdio>
typedef long long ll;
```

```
using namespace std;
char mat[20][20][80];
int size[20];
int style[20];
int allStolb;
char st[80];
int main(){
    gets(st);
    allStolb=strlen(st);
    for (int i=0;i<allStolb;++i)
        style[i]=st[i]=='<'?0:st[i]=='='?1:2;
    int strok=0;
    while (1){
        gets(st);
        if (strcmp(st,"*")==0)
            break;
        int uk=0;
        int nowLen=0;
        for (int i=0;st[i]!=0;++i)
            if (st[i]!='&')
                mat[uk][strok][nowLen++]=st[i];
            else{
                mat[uk][strok][nowLen]=0;
                if (strlen(mat[uk][strok])>size[uk])
                    size[uk]=strlen(mat[uk][strok]);
                uk++;
                nowLen=0;
            }
        mat[uk][strok][nowLen]=0;
        if (strlen(mat[uk][strok])>size[uk])
            size[uk]=strlen(mat[uk][strok]);
        strok++;
    }
    int tableSize=1+allStolb;
    for (int i=0;i<allStolb;++i)
        tableSize+=2+size[i];
    cout << "@";
    for (int i=0;i<tableSize-2;++i)
        cout << "-";
    cout << "@" << endl;
    for (int j=0;j<strok;++j){
        cout << "|";
        for (int i=0;i<allStolb;++i){
            if (style[i]==0){
                cout << " ";
                cout << mat[i][j];
                for (int z=0;z<size[i]-strlen(mat[i][j]);++z)
                    cout << " ";
                cout << " |";
            }else
            if (style[i]==1){
                cout << " ";
                for (int z=0;z<(size[i]-strlen(mat[i][j]))/2;++z)
                    cout << " ";
                cout << mat[i][j];
                for (int
z=0;z<size[i]-strlen(mat[i][j])-(size[i]-strlen(mat[i][j]))/2;++z)
                    cout << " ";
                cout << " |";
            }else
            if (style[i]==2){
```

```
                cout << " ";
                for (int z=0;z<size[i]-strlen(mat[i][j]);++z)
                    cout << " ";
                cout << mat[i][j];
                cout << " |";
            }
        }
        cout << endl;
        if (j==0){
            cout << "|";
            for (int i=0;i<allStolb;++i){
                for (int j=0;j<size[i]+2;++j)
                    cout << "-";
                if (i<allStolb-1)
                    cout << "+";
            }
            cout << "|" << endl;
        }
    }
    cout << "@";
    for (int i=0;i<tableSize-2;++i)
        cout << "-";
    cout << "@" << endl;
    return 0;
}
```

Тест, выявивший ошибку

```
>=<
PHANTASMAGORIA&CANTO I&"The Trystyng"
One&winter night,&at half-past nine,
Cold, tired,&and cross,&and muddy,
I had come home,&too late&to dine,
And supper,&with cigars&and wine,
Was&waiting&in the study.
There was&a strangeness&in the room,
And&Something&white and wavy
Was standing&near me&in the gloom -
I took it&for the&carpet-broom
Left by&that careless&slavey.
But presently&the Thing&began
To shiver&and&to sneeze:
On which&I said&"Come, come, my man!
That's a most&inconsiderate&plan.
Less&noise there,&if you please!"
"I've caught a cold,"&the Thing&replies,
"Out there&upon the&landing."
I turned&to look&in some surprise,
And there,&before&my very eyes,
A little&Ghost&was standing!
*
```

Ожидаемый ответ

```
@-------------------------------------------------------------@
|       PHANTASMAGORIA |    CANTO I    | "The Trystyng"      |
|----------------------+----------------+---------------------|
|                One | winter night, | at half-past nine,   |
|         Cold, tired, |  and cross,   | and muddy,          |
|     I had come home, |   too late    | to dine,            |
|          And supper, |  with cigars  | and wine,           |
|                Was |    waiting    | in the study.       |
|            There was | a strangeness | in the room,        |
|                And |   Something   | white and wavy      |
|         Was standing |    near me    | in the gloom -      |
```

```
|           I took it |    for the    | carpet-broom        |
|             Left by | that careless | slavey.             |
|        But presently |   the Thing   | began               |
|           To shiver |     and       | to sneeze:          |
|            On which |    I said     | "Come, come, my man! |
|        That's a most | inconsiderate | plan.               |
|                Less | noise there,  | if you please!"     |
| "I've caught a cold," |   the Thing   | replies,            |
|           "Out there |   upon the    | landing."           |
|            I turned |    to look    | in some surprise,   |
|           And there, |    before     | my very eyes,       |
|             A little |     Ghost     | was standing!       |
@--------------------------------------------------------------@
```

Вариант 4 (1667)

```
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>

using namespace std;

int main(){
    int n;
    string str="";
    vector<vector<int> > a;
    cin>>n;
    vector<string> res(n*2+1);
    for(int i=0; i<n; i++){
        vector<int> buf;
        int m,x;
        cin>>m;
        for(int j=0; j<m; j++){
            cin>>x;
            buf.push_back(x);
        }
        a.push_back(buf);
    }

    for(int i=0; i<n; i++){
        str+='|';
        for(int j=0; j<a[i].size(); j++){
            int x=a[i][j];
            for(int k=0; k<x; k++)
                str+=' ';
            str+='|';
        }
        res[2*i+1]=str;
        str.erase();
    }

    str+='+';
    for(int i=0; i<a[0].size(); i++){
        int x=a[0][i];
        for(int j=0; j<x; j++)
            str+='-';
        str+='+';
    }
    res[0]=str;
    str.erase();
```

```
    for(int i=0; i<n-1; i++){
        int aa=a[i].size(),bb=a[i+1].size();
        int z=max(aa,bb);
        str+='+';
        for(int j=0; j<z; j++){
            int x,y,p,q;
            if(j < a[i].size())
                p=a[i][j];
            else
                p=-1;

            if(j < a[i+1].size())
                q=a[i+1][j];
            else
                q=-1;

            if(p == -1){
                x=q;
                y=0;
            }
            else{
                if(q == -1){
                    x=p;
                    y=0;
                }
                else{
                    x=min(p,q);
                    y=max(p,q);
                }
            }
            bool check=false;
            for(int k=0; k<x; k++){
                str+='-';
                if(k+1 == x && !check && y != 0 && x != y){
                    x=y;
                    check=true;
                    str+='+';
                    k++;
                }
            }
            str+='+';
        }
        res[2*(i+1)]=str;
        str.erase();
    }

    str+='+';
    for(int i=0; i<a[n-1].size(); i++){
        int x=a[n-1][i];
        for(int j=0; j<x; j++)
            str+='-';
        str+='+';
    }
    res[res.size()-1]=str;

    for(int i=0; i<res.size(); i++){
        cout<<res[i]<<"\n";
    }

    return 0;
}
```

Тест, выявивший ошибку

```
3
2 1 2
2 2 2
2 3 1
```

Ожидаемый ответ

```
+-+--+
| |  |
+-++-++
|  |  |
+--++-+
|   | |
+---+-+
```

№12 Лабораторная

Задание выполняется в группе от 2 до 4 человек (ограничения githib)

1. Зарегистироваться на githib (все)
2. Создать начальные файлы проекта (один из группы)

SciTE.properties:

```
build.goal=all
```

program.cpp:

```
#include "graphics.h"
#include "picture.hpp"
int main()
{ initwindow(800,600);
  house(); // дом
  sun(); // солнце
  man(); // человек
  fence(); // забор
  getch();
  closegraph();
}
```

picture.hpp:

```
void   house(); // 1 участник house.cpp
void   sun(); // 2 участник sun.cpp
void   man(); // 3 участник man.cpp
void   fence(); // 4 участник fence.cpp
```

.gitignore

```
*.o
*.exe
project.api
```

Временные файлы, файлы .exe в репозиторий не записываем!
Через Settings/Manage access предоставить доступ к проекту остальным участникам группы

3. Установить (все)
H:\SOFT\LNGS\githubdesktop\githubdesktopSetup-x64.exe
Выйти
4. Запустить (только в на компьютерах кафедры)
H:\SOFT\LNGS\githubdesktop\config.bat
5. Запустить через ярлык githubdesktop
6. Скачать проект
7. Create new branch (название как функция)

8. Добавить файл (имя функции).cpp:
#include "graphics.h"
#include "picture.hpp"
void (имя функции) () {
... рисование части картинки
}
9. Выполнять цикл Pull request/Merge/Fetch по слиянию всех веток в master
10. Проверить результат
11. Исправить ошибки (размеры, позиция) для получения согласованной картинки в соответствующих ветках
12. Отправить исправления в master

Вопросы к зачету

1. Программное обеспечение. Особенности разработки ПО.

2. Этапы разработки ПО (Software development lifecycle). Временные затраты.

3. Модели ЖЦ ПО (каскадная, инкрементальная, спиральная).

4. Гибкие (agile) методы разработки: Экстремальное программирование (XP)
(особенности, принципы, модель ЖЦ, состав команды)

5. DataOps и DevOps
(особенности, принципы)

6. Планирование системы (термины: проблемы, потребности и функции)
(временные затраты и задачи этапа, перечислить используемые методики).

7. Интервью (определение проблем и потребностей)
(цели, план интервью, итоги)

8. Мозговой штурм (определение функций)
(цели, принципы, подведение итогов)

9. Определение системы
(термины: функциональные и нефункциональные требования, спецификация требований)
(временные затраты и задачи этапа, перечислить используемые методики).

10. Проектирование архитектуры системы (Соммервиль главы 10, 11)
Структура системы, модели управления, виды декомпозиции,
архитектуры распределенных систем

11. Детальное проектирование системы.
Объектно-ориентированное проектирование. Основные виды диаграмм UML.
Диаграмма классов, классы и связи.

12. Проектирование интеллектуального интерфейса пользовате

Принципы, способы взаимодействия, представление информации.

13. Реализация (кодирование).
(временные затраты и задачи этапа,
основные программные платформы и компоненты систем искусственного интеллекта: механизмы логического вывода (рассуждений), объяснений, приобретения знаний, разработка через тестирование).

14. Тестирование.
Принципы тестирования. Тестирование и реализация.
Восходящее и нисходящее тестирование (термины: драйверы и заглушки), преимущества и недостатки.

15. Тестирование модуля как черного ящика (цели, используемые методики)

16. Тестирование модуля как белого ящика (цели, используемые методики)

17. Системное тестирование (цели, используемые методики)

18. Эксплуатация и сопровождение
(временные затраты и задачи этапа, перечислить используемые методики).

19. Отладка (задача, методы, принципы)

20. CASE-средства.
Что такое CASE-средство. Примеры CASE-средств и их назначение (IDE, VCS и др.)

01.03.02_Прикладная матем_ и искусств интеллект_Современные технологии разработки программных систем искусственного интеллекта_2024_о

Проректор по учебной работе	утверждено	В.Е. Федоров

Ученым советом математического факультета

Протокол заседания № 3 от 19.10.2023

Председатель Ученого совета математического факультета	согласовано	Е.А. Сбродова

Заседанием кафедры теории управления и оптимизации

Протокол заседания № 4 от 12.10.2023

Заведующий кафедрой	согласовано	И. В. Изместьев
Автор (составитель)		Н.Д. Ливанов

Структура рабочей программы соответствует приказу ректора ФГБОУ ВО «ЧелГУ» от «13» апреля 2021 г. № 247-1