ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СТАВРОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»

УТВЕРЖДАЮ

Директор/Декан
факультета цифровых технологий
Шлаев Дмитрий Валерьевич

_____

_____

«___» _____ 20___ г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ (ОЦЕНОЧНЫХ МАТЕРИАЛОВ)

Б1.В.06 Разработка программных приложений

09.03.02 Информационные системы и технологии

Инженерия информационных систем

бакалавр

очная

Ставрополь, 2026

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

Процесс изучения дисциплины направлен на формирование следующих компетенций ОП ВО и овладение следующими результатами обучения по дисциплине:

Код и наименование компетенции	Код и наименование индикатора достижения	Перечень планируемых результатов обучения по дисциплине
ПК-1 Выполнение работ по созданию (модификации) и сопровождению ИС, автоматизирующих задачи организационного управления и бизнес-процессы	ПК-1.1 Способен создавать программный код, настраивать и конфигурировать ИС в рамках выполнения работ по созданию (модификации) и сопровождению ИС	**знает** как создавать программный код, настраивать и конфигурировать ИС в рамках выполнения работ по созданию (модификации) и сопровождению ИС
		умеет создавать программный код, настраивать и конфигурировать ИС в рамках выполнения работ по созданию (модификации) и сопровождению ИС
		владеет навыками навыками создания программного кода, настройки и конфигурирования ИС в рамках выполнения работ по созданию и сопровождению ИС
ПК-1 Выполнение работ по созданию (модификации) и сопровождению ИС, автоматизирующих задачи организационного управления и бизнес-процессы	ПК-1.2 Способен развертывать серверную и клиентскую часть ИС у заказчика в рамках выполнения работ по созданию (модификации) и сопровождению ИС	**знает** как развертывать серверную и клиентскую часть ИС у заказчика в рамках выполнения работ по созданию (модификации) и сопровождению ИС
		умеет развертывать серверную и клиентскую часть ИС у заказчика в рамках выполнения работ по созданию (модификации) и сопровождению ИС
		владеет навыками навыками развертывания серверной и клиентской части ИС у заказчика в рамках выполнения работ по созданию и сопровождению ИС
ПК-1 Выполнение работ по созданию (модификации) и сопровождению ИС, автоматизирующих задачи организационного управления и бизнес-процессы	ПК-1.3 Выполняет интеграцию ИС с существующими ИС заказчика в рамках выполнения работ по созданию (модификации) и сопровождению ИС	**знает** как выполнять интеграцию ИС с существующими ИС заказчика
		умеет интегрировать ИС с существующими ИС заказчика в рамках выполнения работ по созданию и сопровождению ИС
		владеет навыками навыками интеграции ИС с существующими ИС заказчика в рамках выполнения работ по созданию и сопровождению ИС

2. Перечень оценочных средств по дисциплине

№	Наименование раздела/темы	Семестр	Код индикаторов достижения компетенций	Оценочное средство проверки результатов достижения индикаторов компетенций
1.	1 раздел. 1			
1.1.	ОСНОВЫ РАЗРАБОТКИ ТРЕБОВАНИЙ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ И АНАЛИЗ ИСПОЛНЕНИЯ ТРЕБОВАНИЙ	6	ПК-1.1, ПК-1.2, ПК-1.3	Тест
1.2.	ТЕХНИЧЕСКИЕ СПЕЦИФИКАЦИИ И ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ	6	ПК-1.1, ПК-1.2, ПК-1.3	Устный опрос
1.3.	ОРГАНИЗАЦИОННОЕ И ТЕХНОЛОГИЧЕСКОЕ ОБЕСПЕЧЕНИЕ КОДИРОВАНИЯ НА ЯЗЫКАХ ПРОГРАММИРОВАНИЯ	6	ПК-1.1, ПК-1.2, ПК-1.3	Тест
1.4.	ИНТЕРФЕЙС ПРОГРАММНЫХ ПРИЛОЖЕНИЙ И ПОЛЬЗОВАТЕЛЬСКИЙ ОПЫТ	6	ПК-1.1, ПК-1.2, ПК-1.3	Устный опрос
	Промежуточная аттестация			Эк

3. Оценочные средства (оценочные материалы)

Примерный перечень оценочных средств для текущего контроля успеваемости и промежуточной аттестации

№ п/п	Наименование оценочного средства	Краткая характеристика оценочного средства	Представление оценочного средства в фонде (Оценочные материалы)
Текущий контроль			
	Для оценки знаний		
1	Устный опрос	Средство контроля знаний студентов, способствующее установлению непосредственного контакта между преподавателем и студентом, в процессе которого преподаватель получает широкие возможности для изучения индивидуальных особенностей усвоения студентами учебного материала.	Перечень вопросов для устного опроса

2	Тест	Система стандартизированных заданий, позволяющая автоматизировать процедуру измерения уровня знаний и умений обучающегося.	Фонд тестовых заданий
	Для оценки умений		
	Для оценки навыков		
	Промежуточная аттестация		
3	Экзамен	Средство контроля усвоения учебного материала и формирования компетенций, организованное в виде беседы по билетам с целью проверки степени и качества усвоения изучаемого материала, определить необходимость введения изменений в содержание и методы обучения.	Комплект экзаменационных билетов

4. Примерный фонд оценочных средств для проведения текущего контроля и промежуточной аттестации обучающихся по дисциплине (модулю) "Разработка программных приложений"

Примерные оценочные материалы для текущего контроля успеваемости

ПК-1.1 Способен создавать программный код, настраивать и конфигурировать ИС в рамках выполнения работ по созданию (модификации) и сопровождению ИС

ЗНАНИЯ

1. Какое ключевое преимущество клиент–серверной архитектуры подчёркивается в учебнике?
A) Отсутствие задержек сети
B) Централизация бизнес-логики и данных на сервере
C) Полная независимость клиента от сервера
D) Ненужность масштабирования
Ответ: B.

2. Что включает этап сопровождения ПО?
A) Только добавление новых функций
B) Только исправление ошибок
C) Исправление ошибок, добавление функций, оптимизация и адаптация
D) Только оптимизацию производительности
Ответ: C.

3. Выберите ДВЕ верные выгоды от использования систем управления версиями (VCS):
A) Возможность отката к предыдущим состояниям проекта
B) Автоматическое исправление конфликтов без участия разработчика
C) Параллельная работа команды с объединением изменений после проверки
D) Запрет на интеграцию с конвейерами CI/CD
Ответ: A, C.

4. Какие ДВА понятия относятся к базовым концепциям Git?
A) Ветки (branches)
B) Оптимизатор SQL-запросов
C) Пулл-реквесты (Pull Requests)
D) Сборщик мусора JVM
Ответ: A, C.

5. Что характерно для Waterfall (каскадной) модели? Выберите ДВА варианта.

A) Проектирование предшествует реализации, ошибки на этом этапе критичны
B) Лёгкая адаптация архитектуры в ходе разработки
C) Разработка стартует до завершения проектирования
D) Возврат на прошлые этапы возможен, но дорог
Ответ: A, D.
УМЕНИЯ
6. Расположите шаги типового рабочего процесса с Git для проекта на Rust:
A) Создание новой ветки
B) Внесение изменений и коммит
C) Отправка изменений в удалённый репозиторий (push)
D) Клонирование удалённого репозитория
Правильный порядок: D → A → B → C
7. Соотнесите термин (1–4) и определение (A–D):
1. Функциональные требования
2. Нефункциональные требования
3. «Хлебные крошки»
4. Эвристическая оценка
A) Навигационный паттерн, показывающий путь и уровень вложенности
B) Описывают, что должна делать система
C) Быстрый экспертный аудит интерфейса по набору правил
D) Определяют требуемые качества: производительность, надёжность, безопасность, UX
Ответ: 1–B, 2–D, 3–A, 4–C.
8. Верно ли, что результатом выполнения программы на RUST будет 2?

```
fn main() {
  let s = String::from("hi");
  let len = s.len();
  println!("{}", len);
}
```

Ответ: Верно

9. Верно ли, что результатом выполнения программы на RUST будет 6?

```
fn main() {
  let mut x = 5;
  x = x + 1;
  println!("{}", x);
}
```

Ответ: Верно

10. Верно ли, что результатом выполнения программы на RUST будет 4?

```
fn main() {
  let s = String::from("ok");
  let r1 = &s;
  let r2 = &s;
  println!("{}", r1.len() + r2.len());
}
```

Ответ: Верно

НАВЫКИ
11. В ответе впишите целое число, которое будет получено в результате выполнения программы:

```
fn fact(n: u64) -> u64 {
  (1..=n).product()
}
fn main() {
  println!("{}", fact(5));
}
```

Ответ: 120

12. В ответе впишите целое число, которое будет получено в результате выполнения программы:

```
fn main() {
  let s = "abracadabra";
  let vowels = ['a','e','i','o','u'];
  let count = s.chars().filter(|c| vowels.contains(c)).count();
  println!("{}", count);
}
```

Ответ: 5

13. В ответе впишите целое число, которое будет получено в результате выполнения программы:

```
fn main() {
  let sum: i32 = (1..=10).filter(|n| n % 3 == 0).sum();
  println!("{}", sum);
}
```

Ответ: 18

14. Верно ли, что результатом выполнения программы на RUST будет 1?

```
fn main() {
  let s = String::from("a");
  let t = s;
  println!("{}", s.len());
}
```

Ответ: Неверно (ошибка после перемещения владения)

15. Верно ли, что результатом выполнения программы на RUST будет 3?

```
fn main() {
  let a = [1, 2, 3];
  println!("{}", a[3]);
}
```

Ответ: Неверно (паника из-за выхода за пределы массива)

ПК-1.2 Способен развертывать и администрировать серверную и клиентскую часть ИС у заказчика в рамках выполнения работ по созданию (модификации) и сопровождению ИС

ЗНАНИЯ

1. Что является ключевым преимуществом клиент-серверной архитектуры при развёртывании и администрировании?

A) Полная независимость клиента от сервера

B) Централизация бизнес-логики и данных на сервере

C) Отсутствие задержек сети

D) Необходимость ручного обновления клиентов

Ответ: B.

2. Какая практика прямо предотвращает SQL-инъекции при работе серверной части с БД в Rust (diesel)?

A) Подготовленные (параметризованные) запросы

B) Конкатенация SQL-строк

C) Отключение внешних ключей

D) Логирование паролей для отладки

Ответ: A.

3. Выберите ДВА шага, которые учебник относит к типичному CI/CD-конвейеру для проектов на Rust:

A) cargo fmt/cargo clippy как обязательные проверки

B) cargo audit/cargo deny для контроля уязвимостей/лицензий
C) Ручная правка README перед каждым деплоем
D) Обязательная «ручная» миграция ОС на сервере перед сборкой
Ответ: A, B.

4. Какие ДВА утверждения отражают рекомендуемые практики наблюдаемости/логирования при эксплуатации ИС?
A) Структурированные логи без секретов
B) Корреляционные идентификаторы запросов сквозь все слои
C) Хранение паролей в логах для быстрого доступа
D) Отключение метрик в продакшене
Ответ: A, B.

5. Что характерно для микросервисной архитектуры в эксплуатации? (выберите ДВА)
A) Требует инструментов оркестрации (например, Kubernetes) и мониторинга
B) Все сервисы деплоятся только единым артефактом
C) Отказ одного сервиса валит всю систему
D) Сервисы разворачиваются и масштабируются независимо
Ответ: A, D

УМЕНИЯ

6. Расположите шаги типового конвейера поставки сервиса на Rust:
A) Сборка и публикация артефактов/образов
B) Линт и сборка (fmt, clippy, build)
C) Развёртывание в целевое окружение
D) Тестирование (cargo test, доктесты)
Правильный порядок: B → D → A → C.

7. Соотнесите термин (1–4) и определение (A–D):
1. Контейнеризация
2. Оркестрация
3. Наблюдаемость
4. План отката (rollback)
A) Управление множеством контейнеров/сервисов в продакшене
B) Упаковка приложения и зависимостей в образ
C) Заранее подготовленная процедура возврата версии
D) Видимость состояния через логи, метрики и трассировки
Ответ: 1–B, 2–A, 3–D, 4–C.

8. Верно ли, что если PORT не задана, программа выведет 8080?
```
use std::env;
fn main() {
  let port = env::var("PORT").unwrap_or("8080".to_string());
  println!("{}", port);
}
```
Ответ: Верно

9. Верно ли, что программа выведет true?
```
use std::net::IpAddr;
fn main() {
  let ip: IpAddr = "0.0.0.0".parse().unwrap();
  println!("{}", ip.is_ipv4());
}
```
Ответ: Верно

10. Верно ли, что программа выведет 2?
```
fn main() {
  let levels = vec!["INFO", "ERROR", "DEBUG", "ERROR"];
  let count = levels.iter().filter(|&&s| s == "ERROR").count();
```

```
  println!("{}", count);
}
```
Ответ: Верно

НАВЫКИ

11. Впишите число, которое напечатает программа:
```
fn main() {
  let cfg = "HOST=0.0.0.0;PORT=9090";
  let port: u16 = cfg
      .split(';')
      .find(|s| s.starts_with("PORT="))
      .and_then(|s| s[5..].parse().ok())
      .unwrap();
  println!("{}", port);
}
```
Ответ: 9090

12. Впишите число, которое напечатает программа:
```
fn main() {
  let s = "error,info,info,debug,error,trace";
  let out = s.split(',').filter(|&x| x == "error").count();
  println!("{}", out);
}
```
Ответ: 2

13. Впишите слово, которое напечатает программа:
```
fn main() {
  let url = "https://example.org/api/v1/health";
  let last = url.rsplit('/').next().unwrap();
  println!("{}", last);
}
```
Ответ: health

14. Верно ли, что программа выведет true?
```
fn main() {
  let line = "user=admin;password=secret";
  let has_secret = line.contains("password=");
  println!("{}", has_secret);
}
```
Ответ: Верно

15. Верно ли, что программа успешно скомпилируется и выведет error?
```
fn main() {
  let levels = ["info","warn","error"];
  let first = levels[3];
  println!("{}", first);
}
```
Ответ: Неверно (паника из-за выхода за границы массива во время выполнения)

ПК-1.3 Способен интегрировать ИС с существующими ИС заказчика в рамках выполнения работ по созданию (модификации) и сопровождению ИС

ЗНАНИЯ

1. Что является базовым требованием безопасности при интеграции с внешним API?

A) Отключить TLS ради скорости

B) Использовать HTTPS и механизм аутентификации (ключ/токен)

C) Передавать секреты в URL

D) Хранить токены в логах
Ответ: B.
2. Что помогает избежать «ломающих» изменений при эволюции внешнего сервиса?
A) Игнорирование изменений в ответах API
B) Жёсткое кодирование структуры ответа
C) Явное версионирование API и отслеживание обновлений
D) Полный отказ от внешних API
Ответ: C.
3. Выберите ДВА подхода для повышения надёжности интеграции с внешним API:
A) Повторные попытки с backoff
B) Кэширование ответов
C) Передача паролей в открытом виде
D) Жёсткое падение при первом сетевом сбое
Ответ: A, B.
4. Какие два фактора часто вызывают несовместимость при интеграции?
A) Разные форматы/версии API (JSON, XML, Protobuf)
B) Единый формат по умолчанию
C) Стабильные неизменяемые схемы
D) Изменение структуры ответов внешнего сервиса
Ответ: A, D
5. Какие ДВА преимущества событийной интеграции через брокер сообщений?
A) Слабая связность компонентов
B) Обязательная синхронность вызовов
C) Буферизация событий в очереди при временной недоступности потребителя
D) Жёсткая зависимость от порядка вызовов RPC
Ответ: A, C

УМЕНИЯ
6. Расположите шаги типовой интеграции с внешним API:
A) Реализовать повторные попытки при сбоях
B) Настроить безопасность (HTTPS, токен/ключ)
C) Явно указать используемую версию API
D) Добавить кэширование ответов
Правильный порядок: B → C → A → D
7. Соотнесите термин (1–4) и определение (A–D):
1. API-токен/ключ
2. Версионирование API
3. Кэширование ответов
4. Мок-сервер для интеграционных тестов
A) Явное указание версии интерфейса для контролируемых переходов
B) Симуляция внешнего сервиса для тестов интеграции
C) Секрет для аутентификации запросов
D) Снижение числа запросов к внешнему API путём хранения результатов
Ответ: 1–C, 2–A, 3–D, 4–B
8. Верно ли, что после компиляции программа выведет https://api.example.com/v2/users?

```
fn main() {
  let base = "https://api.example.com";
  let v = "v2";
  let path = "users";
  let url = format!("{}/{}/{}", base, v, path);
  println!("{}", url);
}
```

Ответ: Верно

9. Верно ли, что после компиляции программа выведет true?

```
fn main() {
```

```
  let mut headers = String::from("Authorization: Bearer ");
  headers.push_str("ABC123");
  println!("{}", headers.contains("Bearer ABC123"));
}
```
Ответ: Верно

10. Верно ли, что после компиляции программа выведет xyz?
```
fn main() {
  let qs = "a=1&b=2&token=xyz";
  let token = qs.split('&')
      .find(|p| p.starts_with("token="))
      .unwrap();
  println!("{}", &token[6..]);
} Ответ: Верно
```

НАВЫКИ

11. В ответе впишите совокупность текстовых символов и чисел, которые напечатает программа:
```
fn main() {
  let h = "Authorization: Bearer TOK123";
  let tok = h.split_whitespace().last().unwrap();
  println!("{}", tok);
}
```
Ответ: TOK123

12. В ответе впишите число, которое напечатает программа:
```
fn main() {
  let qs = "a=1&b=2&b=3";
  let count = qs.split('&').filter(|p| p.starts_with("b=")).count();
  println!("{}", count);
}
```
Ответ: 2

13. В ответе впишите число, которое напечатает программа:
```
fn main() {
  let base = "https://srv.local";
  let ver = "v1";
  let ep = "health";
  let url = format!("{}/{}/{}", base, ver, ep);
  println!("{}", url.len());
}
```
Ответ: 27

14. Верно ли, что программа успешно скомпилируется и выведет XYZ?
```
fn main() {
  let token = String::from("XYZ");
  let header = token;
  println!("{}", token);
}
```
Ответ: Неверно (владение token перемещено в header, повторное использование недопустимо)

15. Верно ли, что программа успешно скомпилируется и выведет true?
```
fn main() {
  let payload = r#"{"ok":true}"#;
  let ok = payload.contains("\"ok\":true");
  println!("{}", ok);
```

}
Ответ: Верно

Примерные оценочные материалы
для проведения промежуточной аттестации (зачет, экзамен)
по итогам освоения дисциплины (модуля)

Вопросы для экзамена
1. Технология программирования и основные этапы ее развития
2. Проблемы разработки сложных программных приложений
3. Блочно-иерархический подход к созданию сложных систем
4. Жизненный цикл
5. Ускорение разработки программного обеспечения. Технология RAD
6. Понятие технологичности программного обеспечения
7. Нисходящая и восходящая разработка программного обеспечения
8. Эффективность и технологичность
9. Программирование «С защитой от ошибок»
10. Сквозной структурный контроль
11. Классификация программных продуктов по функциональному назначению
12. Основные эксплуатационные требования к программным продуктам
13. Предпроектные исследования предметной области
14. Разработка технического задания
15. Принципиальные решения начальных этапов проектирования
16. Спецификации программного обеспечения при структурном подходе
17. Диаграммы переходов состояний
18. Функциональные диаграммы
19. Диаграммы потоков данных
20. Модели
21. Разработка структурной и функциональной схем
22. Исследование метода пошаговой детализации для проектирования структуры программного обеспечения
23. Проектирование структур данных
24. Проектирование программного обеспечения, основанное на декомпозиции данных
25. Case-технологии
26. UML – стандартный язык описания разработки программных продуктов с использованием объектного подхода
27. Определение «вариантов использования»
28. Построение концептуальной модели предметной области
29. Описание поведения. Системные события и операции
30. Разработка структуры программного обеспечения при объектном подходе
31. Определение отношений между объектами
32. Проектирование классов
33. Компоновка программных компонентов
34. Типы пользовательских интерфейсов и этапы их разработки
35. Психофизические особенности человека, учитываемые при создании пользовательского интерфейса
36. Модели пользовательского интерфейса
37. Виды контроля качества
38. Структурное тестирование
39. Функциональное тестирование
40. Комплексное тестирование
41. Оценочное тестирование
42. Классификация ошибок
43. Методы отладки
44. Виды программных документов
45. Пояснительная записка
46. Руководство пользователя
47. Руководство системного программиста

Темы письменных работ (эссе, рефераты, курсовые работы и др.)

Примерные темы курсовых работ

1. Разработка приложения для торгового предприятия
2. Разработка приложения для кредитного отдела банка
3. Разработка приложения для гостиницы
4. Разработка приложения для авторемонтной мастерской
5. Разработка приложения для автосалона
6. Разработка приложения для агентства недвижимости
7. Разработка приложения для склада
8. Разработка приложения для учета расходов семьи
9. Разработка приложения для организации делопроизводства
10. Разработка приложения для рекламного агентства
11. Разработка приложения для службы поддержки
12. Разработка приложения для кадровой службы организации
13. Разработка приложения для туристического агентства
14. Разработка приложения компьютерного магазина
15. Разработка приложения для страховой компании
16. Разработка приложения для кафедры вуза
17. Разработка приложения для мебельного магазина
18. Разработка приложения для книжного магазина
19. Разработка приложения для учета договоров в организации
20. Разработка приложения для строительной компании
21. Разработка приложения по взаимодействию с клиентами организации
22. Разработка приложения для взаимодействия с заказчиками в организации
23. Разработка приложения для салона красоты
24. Разработка приложения сервисного центра по ремонту компьютерной техники
25. Разработка приложения для транспортной компании
26. Разработка приложения для менеджера по продажам
27. Разработка приложения для учета компьютерной техники и программного обеспечения в организации
28. Разработка приложения для организации документооборота
29. Разработка приложения для управляющей компании ЖКХ
30. Разработка приложения для салона сотовой связи